EXPLORING WAYS TO VISUALIZE NEWS OVER GEOGRAPHICAL MAPS

by

Mohamad Hussein Salimian Rizi

Submitted in partial fulfilment of the requirements

for the degree of Master of Computer Science

at

Dalhousie University

Halifax, Nova Scotia

August 2012

DALHOUSIE UNIVERSITY

FACULTY OF COMPUTER SCIENCE

The undersigned hereby certify that they have read and recommend to the Faculty of Graduate Studies for acceptance a thesis entitled “EXPLORING WAYS TO VISUALIZE NEWS OVER GEOGRAPHICAL MAPS” by Mohamad Hussein Salimian Rizi in partial fulfilment of the requirements for the degree of Master of Computer Science.

Dated: 16 August 2012

Supervisor: ____________________________________

Readers: ____________________________________

____

DALHOUSIE UNIVERSITY

DATE: August 16, 2012

AUTHOR: Mohamad Hussein Salimian Rizi

TITLE: EXPLORING WAYS TO VISUALIZE NEWS OVER GEOGRAPHICAL MAPS

DEPARTMENT OR SCHOOL: Faculty of Computer Science

DEGREE: M.C.Sc. CONVOCATION: October YEAR: 2012

Signature of Author

“There are no rules of architecture for a castle in the clouds.”

G. K. Chesterton

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Online news sites are some of the most useful and popular information retrieval systems in use today. Thousands of articles in different languages and on a variety of subjects are posted every day and updated every hour. Most articles are uninteresting and unimportant to a particular individual, and individuals may not want to review entire websites for stories of interest. Systems have been developed that provide summaries from online news websites but finding a means to rapidly scan stories of potential interest remains an open problem. In this thesis, we introduce a novel visualization system that uses geographical location combined with image collages and tag clouds to provide a tool for rapidly reviewing news stories. Tag clouds are arrangements of tags with the most important tags allocated a bigger font size or otherwise more prominent visual properties; and image collages provide a compact, effective and attractive representation for photos on one page. Bringing these media representations together over a geographic map offers a new style of interaction for online news browsing.

The usability of our application was evaluated with two user studies. We aimed to determine how best to configure our visualization to communicate more information in less time to users than traditional feed-based news aggregators. We were particularly interested in knowing whether users interpret text/image size and placement as indications of a news item's prominence. We also wanted to establish whether users understand the semantic relationship between zoom level on the map and the regional relevance (municipal, provincial, national) of news items displayed at that zoom level. The results of user feedback and data analysis (e.g., eye tracking logs) were examined to improve the usability of the system. Data analysis from the second user study suggests that, in general, the system is highly effective in helping users achieve an immediate and effortless bird's-eye-view of news summaries within a large geographic region. However, users had varying opinions about the level of detail in the user interface (e.g., the number of images).

LIST OF ABBREVIATIONS USED

ROI	Region of Interest
Dal	Dalhousie University
CSS	Cascading Style Sheets
API	Application programming interface
TF-IDF	Term Frequency–Inverse document frequency
TF	Term Frequency
TTF	Table Term Frequency
TF*PDF	term frequency–proportional document frequency
PDF	proportional document frequency
NER	Named-Entity Recognition
NLP	Natural Language Processing
GNIS	Geographic Names Information System
ST	Shift- Truncation
STC	Shift-Truncation-Scale
SCT	Shift-Scale- Truncation

TSC	Truncation -Shift-Scale
CBIR	Context of content-Based Image Retrieval
IOR	Incorporation of inhibition of return
GVF	Gradient Vector Flow
CoFiN2	Coarse-to-Fine Normal Neighborhoods strategy
DFT	Discrete Fourier Transform
MCMC	Markov chain Monte Carlo
WGS	World Geodetic System
MM	Medium images-Medium texts
LH	Low images-High texts
HL	High images-Low texts

ACKNOWLEDGEMENTS

I would like to thank Dr. Stephen Brooks and Dr. Derek Reilly for the experience, creativity, useful advice and helpful input throughout the development of this work. I would like to especially thank Dr. Stephen Brooks for his kind supports during the past two years when he never let me feel confused and alone.

I would like to thanks to my lovely mother, and brother who cheered me onwards at all times.

CHAPTER 1

INTRODUCTION

Every day hundreds of news websites post a wide spectrum of news stories to their web portals on an hourly basis. Millions of people check these news websites daily to access stories in their local areas and around the world. However, it remains a challenge to browse through the thousands of news stories returned from a search query from within news aggregation websites. Surfing between different news websites also takes time and can be tedious. We have therefore developed Geo-Collage maps, which combine image cuttings and tags on geographic maps, as a novel visualization system for reviewing news in different geographic regions.

Our geographic visualization tool extracts images from different databases and uses tags to form mixed image and text based geospatial tag clouds, where tag clouds are arrangements of tags with the most important tags enjoying a bigger font size or otherwise more prominent visual properties. Geo-tag clouds distribute tag clouds into a geo-spatial arrangement that conveys textual and visual information.

This research consists of three main parts: image processing, tag cloud geo-tags and two user studies, which evaluate our approach

Collages can be generated with image processing methods to provide a compact, effective and attractive representation for photos on one page. It can act as an attractive visualization of picture albums, which might be desirable as memorabilia for friends who attended an event [1]. A collage layout allows extra information to be expressed such as creating a path through time, telling a story, or remembering an event. Adding a layer of image collage to geo-tag maps provides a new and effective visualization tool for different categories of information such as tourism, science events (conferences, workshops, etc.), sports and news. Figure 1 presents a sample collage image. In this work we focus on news stories; however, we believe the approach could apply more broadly. For example, it might be used for sport events such as Olympics and soccer world cups or for ecotourism purposes.

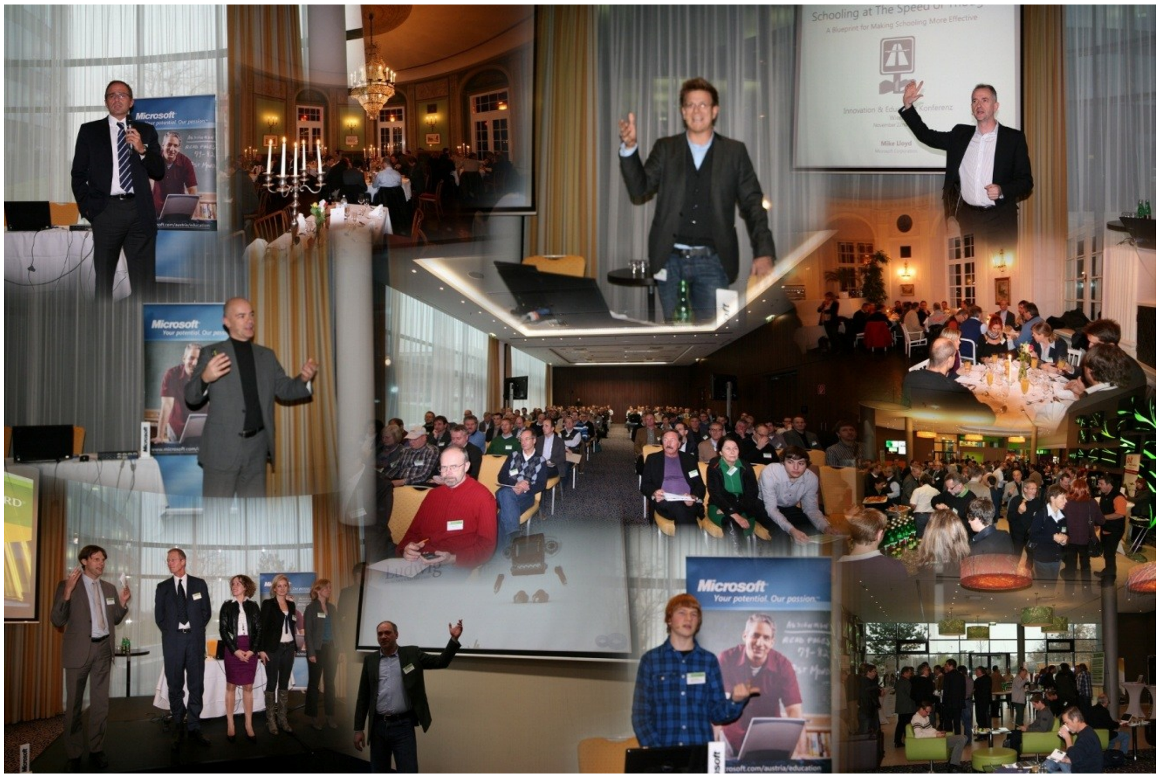


Figure 1: Collage samples; a collage layout contains extra information like creating a path through time, telling a story, or remembering an event.

Chapter 3 discusses the different types of image processing used in the project, including extracting regions of interest (ROI), ROI placement and image summarization.

The second part of this work involves interactively combining tag and image clouds with maps. We refer to these new tools for visualizing data as *geo-tags* and *tag maps*. The central idea is to distribute packed tag and image clouds into a geo-spatial arrangement that conveys textual and visual information while providing overviews and filtering by text and geography. Figure 2 shows an interesting sample of a tourist map that was created manually which should give the reader visual understanding of what our system is trying to achieve. This image was a key inspiration for this work.

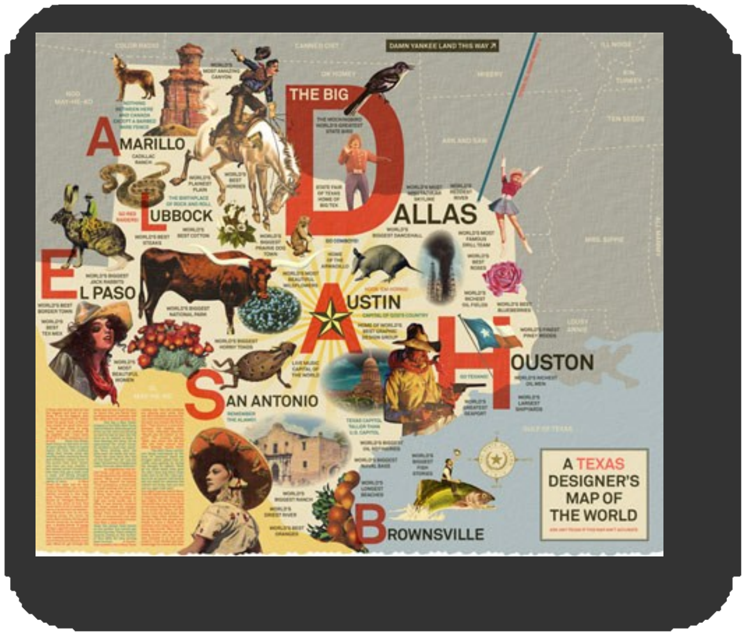


Figure 2: A static, manually created map that contains images and text (retrieved from [2]).

1.1 Thesis overview

Two common spatial queries about news are "Where did story X happen?" "What is happening in location Y?" [3] News software, like NewsStand [3], has been developed to answer these questions. A good news application benefits from an efficient method for extracting keywords and geographic information from online news sources, especially given the transient, time-sensitive nature of news stories.

For this we used keyword extraction techniques for extracting keywords from news stories; however we did not develop a comprehensive keyword extraction implementation since it is not our primary area of contribution. Chapter 4 discusses the relevance of keyword extraction for tag clouds and in particular geo-tags. We review TF-IDF model [4] and the TF*PDF model (Term Frequency * Proportional Document Frequency) [5] algorithms as bases for future work on keyword extraction.

We evaluated the usability of our application with two user studies, which we outline here and present in more detail in chapter 5. We aimed to find out how best to configure our visualization to communicate news-relevant geospatial information to users in a more effective manner than traditional feed-based news aggregators. We aimed to identify proper sizes, numbers and orientations of images and tags that are more attractive and useful for users. We also wanted to study how combinations of tags and images help users to find out about different news stories in different geographic areas, In particular, if they understand the relationship between zoom level on the map and the regional relevance (provincial, national) of news items displayed at that zoom level.

We discuss the details of the user studies and their results in chapter 5, and finally we end our discussions in chapter 6 by summarizing all key contributions and observations, and highlighting useful suggestions for future work.

1.2 Contributions

We end this chapter by stating the contributions made in this thesis. Firstly, we developed a new visualization tool with the capability of presenting image ROIs and text on an interactive map. During our work, we did not find any similar software or applications to our system.

For the text processing part, we developed a ROI placement algorithm for the system. Our placement algorithm uses geographic coordinates and predefined masks of geographic areas. After reading the geographic coordinates of the news from a database, the algorithm converts them to pixel and tile coordinates. Tiles are fixed size images that by putting together they build the map. The placement algorithm first tries to place the ROIs at the coordinates of the event. If it is not successful to use the exact coordinates of an event, i.e. the location on the map was marked before with another ROI, the algorithm will try a new coordinates close to the previous coordinates. In addition, if the algorithm failed in placing the ROI because of the size of the ROI, it will change the size of the ROI and repeat the process again.

In the image processing part, we developed a new ROI extraction method that combines morphological reconstruction, Canny edge detection, active contours and Fourier transform techniques. Briefly, we apply a high frequency filter and Canny edge detection on an image to sharpen the edges, then we use the active contour and morphological reconstruction to extract a logical mask of the ROI. Finally, we apply the mask to the image to extract the ROI.

In the next step, we focused on a method for blending the ROIs into the map. Although we attempted to use the Poisson image editing [6] technique to create a smooth and visual attractive transition between ROIs and background, this algorithm was slow for large images and it made blurry results on small images. We therefore developed a simple new algorithm to blend the ROIs into the map. We called it an averaging method. This method simply combined the value of the pixels in ROIs and background image with different percentage based on the distance of pixels from the center of the ROIs. Our algorithm uses either Poisson image editing or our an averaging method for blending images into the map, depending on their size. In addition, the system during the process of extracting the ROI can mark an image for use with the average method. For example, if the result of high frequency filters was a black image without any sharpen edges, it will be marked for using the average algorithm.

CHAPTER 2

PRIOR WORK / LITERATURE REVIEW

This chapter provides context for other parts of the thesis. First, we discuss image processing techniques related to our work. We then consider related work involving tag clouds and geo-tags, and end with a review of user studies similar to our own.

2.1 Image processing

A part of our work is focused on the extraction of imagery from geo-referenced documents, and the subsequent processing of such imagery to extract regions of interest (ROIs) from images to use in our news visualization application. This section reviews some of image summarization techniques, algorithms and software; the last part of this section discusses existing methods of extracting ROIs within images.

2.1.1 Image summarization techniques and algorithms

Graph cuts have been proposed as an efficient solution for a wide variety of computer vision and graphics problems including image smoothing, region segmentation of objects in N-D images, stereo correspondence, texture synthesis, multi-view recognition, and any vision or graphics application which can be formulated as an energy minimization problem or label assignment problem. We attempted to develop a graph cut algorithm in our implementation for use as a technique for blending images onto a map; however, we later switched to Poisson image editing.

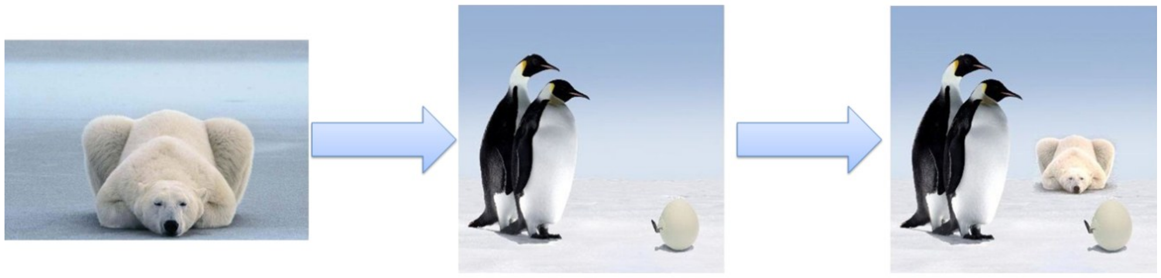


Figure 3: Result of running graph cut and min-cut algorithm on the images of polar bear and penguins.

Graph cut formulates the image as a connected graph with weighted edges. These edges are weighed based on the difference between neighboring pixels. The graph is then treated as a max-flow/min-cut problem where the sources (vertices) are pixels only from the first image and the sinks (vertices) are only pixels from the second image. This technique can be used to render an image within another image; in our case, we could use it to render extracted ROIs onto the map. Figure 3 shows a sample result of running the graph cut and min-cut algorithms on the images of a polar bear and penguins.

Assigning a label to a pixel is a common task in many computer vision problems. A common limitation is that the labels should preserve sharp discontinuities. These tasks are naturally stated in terms of energy minimization [7]. Graph cut algorithms are more efficient than known strongly polynomial time max-flow algorithms, which were designed based on pre-flow push or shortest augmenting path paradigms [8].

Greig [9] was the first to use the theory of graph cuts in computer vision. He obtained the maximum posteriori estimate of a binary image by maximizing the flow through an associated image network, involving the introduction of a source and sink.

Label assignment in general is an NP hard problem [10], but there are some labeling algorithms which can use binary labeling repeatedly to find a good solution. Problems such as texture synthesis and segmentation use the advantage of binary labeling in order to determine the minimum capacity cut in a flow graph [9]. Most of the current research aims to map computer vision problems onto appropriately constructed graphs and describe an energy function that can be minimized by graph cuts [8][11][12]. Minimizing the energy function is difficult because it requires minimizing a non-convex function in an N-D space. This means that without graph cuts we are left with some extremely slow algorithms with an exponential time.

Generally, the idea of using graph cuts is about finding a graph for the energy function to be minimized such that the minimum cut on the graph also minimizes the energy (locally or globally) [10]. The max flow algorithms can compute the minimum cut in an efficient way. These methods have been successfully used for image restoration, stereo and motion, image synthesis, image segmentation, voxel occupancy, multi camera scene reconstruction, and medical imaging. Finding and constructing a good graph for an energy function is a difficult problem itself.

Using binary-valued variables is a common way to calculate energy functions to solve the graph cut problem. Assume G = ($\mathcal{V}$, $\mathcal{E}$) is a directed and non-negative weighted graph where $\mathcal{V}$ shows vertices and $\mathcal{E}$ shows edges. Each edge connects two vertices (terminals), these vertices are called source s and sink t. A s-t cut $C = S, T$ is a partition of the vertices in $\mathcal{V}$ into two disjoint sets S and T such that $s \in S$ and $t \in T$. The cost of the cut can be calculated with equation (1):

$$c(S,T) = \sum_{u \in S, v \in T, (u,v) \in \mathcal{E}} c(u,v) \qquad (1)$$

According to the theorem of Ford and Fulkerson[1], the minimum s-t cut is equal to computing the maximum flow from a source to a sink. There are different polynomial time algorithms for solving this problem. A cut $C = S,T$ is labeling f mapping from the set of the vertices $V - \{s,t\}\ to\ \{0,1\}$, where $f(v) = 0$ means that $v \in S$ and $f(v) = 1$ means that $v \in T$. A cut is viewed as a binary labeling, while the minimum $s - t$ cut problem involves two terminals. If a cut involves more than two terminals then we are facing an NP-hard problem [10].

An expansion move algorithm is an effective algorithm for minimizing the energy function. The expansion move algorithm tries to finds the lowest energy move from the current labeling and replaces it with new labeling if it has a lower energy [13]. A standard form of energy function is:

$$E(f) = \sum_{p \epsilon P} D_p(f_p) + \sum_{p,q \epsilon N} V_{p,q}(f_p, f_q) \qquad (2)$$

N is a system on pixels. $D_p(f_p)$ measures the cost of assigning the label f_p to the pixel p. $V_{p,q}(f_p, f_q)$ measures the cost of assigning the labels f_p, f_q to the adjacent pixels p,q.. The standard 2-camera stereo problem and multi-camera scene reconstruction are two good examples of using this algorithm.

[1] http://en.wikipedia.org/wiki/Ford%E2%80%93Fulkerson_algorithm

Boykov and Kolmogorov [8] have developed a min-cut/max-flow algorithm (BK) which is known as one of the best flow algorithms. Chetan in [7] proposes a new algorithm. He claims that in comparison to the other algorithms like BK, his algorithm acts faster on most of the BVZ (2D dataset) and the other datasets from UWO[2]. In addition, he believes his algorithm is three times faster than the time reported for GPU runs.

2.2 Folksonomy, tag clouds and geo-tags

Our system requires the use of tagging to represent news articles over maps, so we also need to examine existing work in this area.

Tagging is a way for creating metadata, for use with media sharing and social bookmarking and citation websites. Tagging-based systems enable users to add tags – freely chosen keywords– to web resources for organizing their resources for themselves and/or others [14]. "A folksonomy is a system of classification derived from the practice and method of collaboratively creating and managing tags to annotate and categorize content"[3]. A Folksonomy can also be defined as collaborative tagging, social classification, social indexing, and social tagging. Tagging, as one of the characteristics of Web 2.0 services provides the opportunity for users to collectively classify and find information. Websites with different content and use such as shopping, education, weblogs and news include tag clouds as a way to visualize tags in a tag cloud.

---

[2] http://vision.csd.uwo.ca/maxflow-data/

[3] http://en.wikipedia.org/wiki/Folksonomy

Halpin [15] examined the dynamics of collaborative tagging systems (e.g. tag clouds) to show that the frequency distribution of tags for a tagging system (with many tags and many users) can be described by the power law distribution. Results of his study show that consensus around stable distributions and shared vocabularies with or without a central controlled vocabulary do emerge. In addition, categorizing the contents made the information more searchable. In other research Halpin shows that in large folksonomies, common structures also emerge on the level of categorizations [16].

Tag clouds can be also generated directly from content. In our case, tag clouds and interactive maps in combination represent geographically referenced text, which is known as a 'tag map'. Figure 4 briefly shows some advantages of tag maps and tag clouds. Tag maps are a kind of tag cloud in which the position of words is based on geographical locations, and the size of the tag represents the importance of the subject at a specific location [17].

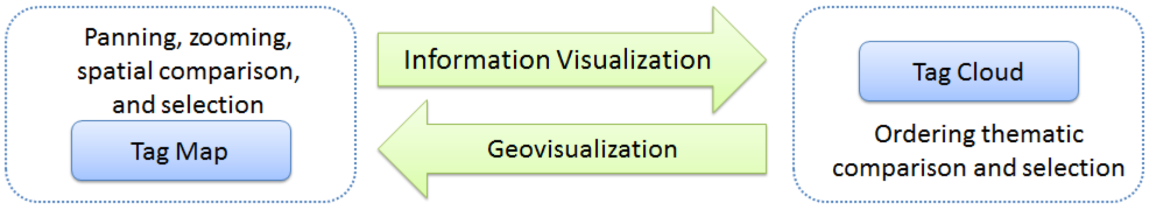

Panning, zooming, spatial comparison, and selection
Tag Map
Information Visualization
Geovisualization
Tag Cloud
Ordering thematic comparison and selection

Figure 4: Interactive tag maps and tag clouds for selecting data.

T.Eda [18] says tag maps are used to explore a large structured spatio-temporal data set by providing overviews and filtering by text and geography. Tag maps enable users to use zoom-in and zoom-out interfaces on maps to explore tags in different regions.

Interactive tag map and tag cloud techniques were informally evaluated by Lohmann et al. through a discussion of successes and limitations encountered [14]. They argue that tag cloud and tag map designers have to balance font size, color, shape, location, etc. to produce an optimized layout; but a lack of knowledge about user perception and acceptance of different layouts remain obstacles to the creation of efficient and useful tag map systems.

2.2.1 Review of some tag cloud algorithms

Standard tag clouds use different font sizes, font weights, and an alphabetical order. The number of times that a subject has been used for tagging usually determines the importance of that tag. Nguyen [17] discusses other ways to accomplish this task—constraining by time or by space for example, especially for tag maps. Next, we review some novel algorithms for creating tag cloud layers.

Hassan & Victor [19] present a new way of tag selection for tag clouds, and use clustering algorithms for visual layout to improve the browsing experience. Tag similarity is the foundation of Hassan's algorithm. They used the semantic relationship between tags to measure tag similarity. This can be achieved by using relative co-occurrence of tags (Jaccard coefficient). If A and B are the sets of resources described by two tags relative co-occurrence is defined as:

$$\mathrm{RC}(\mathrm{A}, \mathrm{B}) = \frac{|\mathrm{A} \cap \mathrm{B}|}{|\mathrm{A} \cup \mathrm{B}|} \tag{3}$$

This similarity based on co-occurrence is also called tag overlapping. Hassan's algorithm uses a tag cloud layout that was designed based on clustering techniques. Figure 5 shows a sample result of Hassan's algorithm.

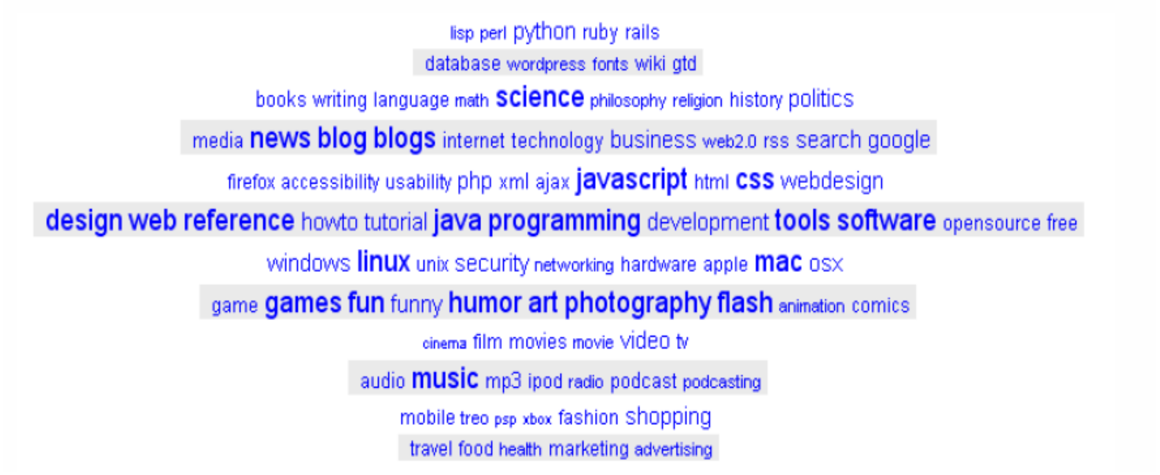
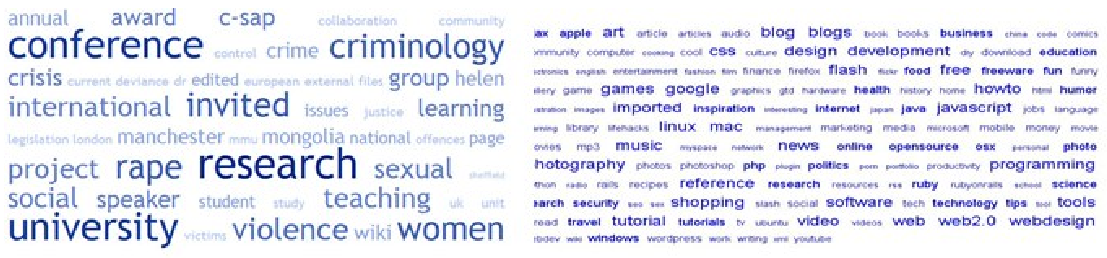

lisp perl python ruby rails
database wordpress fonts wiki gtd
books writing language math science philosophy religion history politics
media news blog blogs internet technology business web2.0 rss search google
firefox accessibility usability php xml ajax javascript html css webdesign
design web reference howto tutorial java programming development tools software opensource free
windows linux unix security networking hardware apple mac osx
game games fun funny humor art photography flash animation comics
cinema film movies movie video tv
audio music mp3 ipod radio podcast podcasting
mobile treo psp xbox fashion shopping
travel food health marketing advertising

Figure 5: Sample result of Hassan & Víctor algorithm.

The PubCloud layout [20] is designed to visualize queries from a database of biomedical literature. The PubCloud algorithm first generates a tag list from the database in reply to the requested query. It calculates frequencies of each tag according to the number of occurrences of each tag. The resulting frequency is represented by font size. In addition, PubCloud calculates recency by using the average publication dates; the results of recency show in different colors (ranging from bright red for the most recent to dark grey for the oldest). As you can see in Figure 6 (a), the PubCloud method produces a lot of unused white spaces. Kuo [20] conducted a user study with 20 participants. He believes tag clouds themselves are not appropriate tools for web summarization; however, they can help other techniques as a summarizing descriptive information tool. Kuo's results indicate that tag clouds not only do not help the user find relational concepts, but can slow them down when retrieving an specific item. Kaser & Lemire [21] proposed an algorithm to improve the display of tag clouds, which consists of in-line HTML and nested tables. (See Figure 6 (b)). An inline HTML tag cloud is a text block that is styled by HTML elements like span, font, br, etc. Cascading style sheets (CSS) is one of the best ways to define the HTML style. Kaser [21] used the length of tags in his algorithm to calculate the size (height or width) of lines and white spaces.


annual award c-sap collaboration community
conference control crime criminology
crisis current deviance dr edited european external files group helen
international invited issues justice learning
legislation london manchester mmu mongolia national offences page
project rape research sexual sheffield
social speaker student study teaching uk unit
university victims violence wiki women
apple art article articles audio blog blogs book books business china code comics
computer cooking cool css culture design development diy download education
english entertainment fashion film finance firefox flash flickr food free freeware fun funny
game games google graphics gtd hardware health history home howto html humor
images imported inspiration interesting internet japan java javascript jobs language
library lifehacks linux mac management marketing media microsoft mobile money movie
mp3 music myspace network news online opensource osx personal photo
photos photoshop php plugin politics porn portfolio productivity programming
radio rails recipes reference research resources rss ruby rubyonrails school science
security seo sex shopping slash social software tech technology tips tool tools
read travel tutorial tutorials tv ubuntu video videos web web2.0 webdesign
wiki windows wordpress work writing xml youtube

(a) (b)

Figure 6: Existing tag layouts: (a) PubCloud layout, (b) in nested HTML tables

He defined some criteria for his algorithm with arbitrary placement:

1. Tags can be reordered and placed arbitrarily but they should not overlap or rotate.

2. Related tags should be in close proximity (for this, the relation between tags should be known).

3. Tag-cloud width has an upper bound.

4. Tag-cloud height should be small.

5. Tags may be deformed slightly.

Large clumps of white space are not desired.

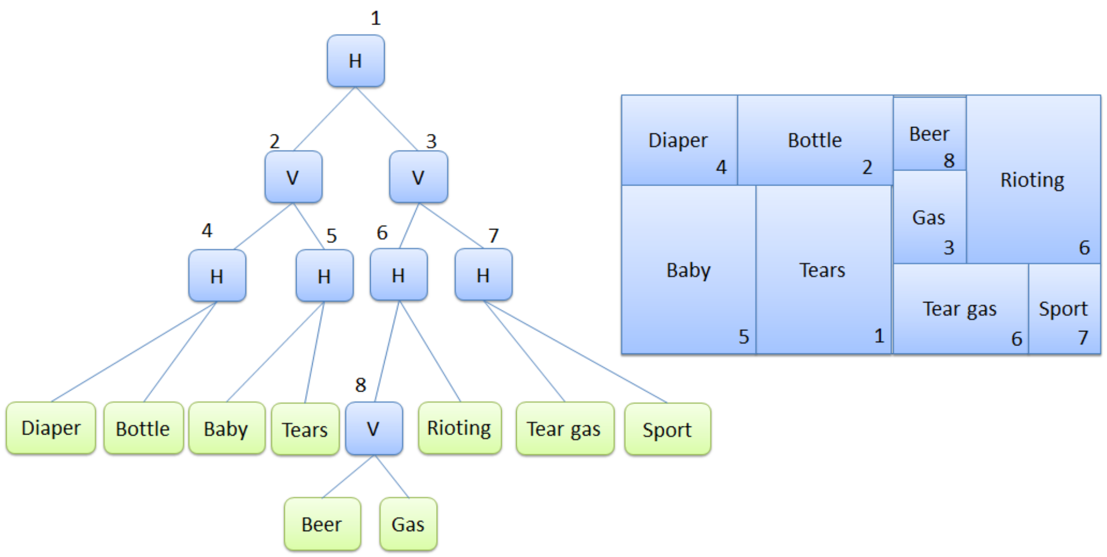


Figure 7: Left Slicing tree, Right associated floor plan.

Kaser & Lemire [21] used a graph structure to create a clustering of semantically related tags [29]. In a graph, vertices represent tags and weighted edges show related tags and their weights indicate the strength of relation between tags. A reasonable measure of spatial non-proximity is:

$$\sum_{p,q} w(p,q)d(p,q) \qquad (4)$$

Where p and q are placed tags linked with strength $w(p,q)$ and separated spatially by distance $d(p,q)$. Small values indicate better clustering. The Kaser & Lemire [21] cloud layer algorithm uses Min-cut placement [29] for fast arbitrary tag placement. Min-cut placement recursively uses bi-partitioning (splitting tags to right and left groups) to create a slicing tree graph. Figure 7 shows a sample Slicing tree and its associated floor plan.

Trees are well-known graph problems, given n tags with $m \in \Omega(n)$ relationships; min-cut placement can run in $O(m \log n)$.

Seifert [22] believes a tag cloud algorithm should be able to produce tag layers in arbitrary convex polygons with the capability of dynamically adapting the tag font size. This means his algorithms can put the maximum number of tags in a layer. In addition, putting tag layers in different convex polygons opens the way for new design ideas. We used Seifert's [22] dividing method idea for the algorithm in our tag cloud application. See chapter 4 for more details.

Tags and keywords are contributed from different databases; some of these sources can be more valuable and trusted in comparison to other sources. For example, extracted news keywords from a government official website are more valuable and trusted than extracted news keywords from a local news agency. This means that information about who contributed and labeled the content with keywords are as important as the keywords themselves.

Shaw [23], by inspiring the idea mentioned in the previous paragraph, tried to show tag clouds in a graph structure; in his design, nodes represent tags and edges between nodes show the relation between tags. However, his layout suffers from lot of unused white space with overlapping tags (See Figure 6 (c)). Shaw used the Kullback-Leibler divergence to calculate the distance between two probability distributions:

$$distance(P, Q) = \sum_{i=0}^{D} P_i \log \frac{P_i}{Q_i} \qquad (5)$$

Where P and Q are probability distributions. Shaw's statistics are defined as follows [23]:

- Number of users, tags, and links collected.
- Average tags per user, tags per link, and links per user.
- Variance of tags per user, tags per link, and links per user.

Bielenberg and Zacher [24] chose a circular layout for visualizing tag clouds. They use the font-size and the position of tags from the center of the orbits to show the relevance of the tags. They used a hierarchical clustering algorithm [25] to cluster resources and tags based on their similarity:

$$\mathrm{S_{AB}} = \frac{\mathrm{CT_{AB}}}{(\mathrm{OT_{AB}} - \mathrm{CT_{AB}})} \tag{6}$$

Where S is a similarity value for each pair of resources AB, CT is the number of correlated tags and OT is the overall number of tags. Similarity values are saved in an $N \times N$ distance matrix to use as the clustering process input.

2.3 Review of user studies on tag clouds

Most user studies evaluate different tag cloud interfaces by making a comparison between them. For instance, Halvey and Keane [26] compared tag clouds with un-weighted horizontal and vertical lists. Results of their user study indicate that alphabetization allows users easier and faster search of tags:

- Font size and position of tags play an important role for conducting a fast and easy search.

- Users scan lists and clouds rather than read them and tags in the upper left corner of the cloud were found faster than other positions.

Sinclair and Cardew-Hall [27] conducted an experiment to explore whether or not tag clouds can help users find information. They asked participants to answer a set of questions using either a tag cloud or a traditional search interface. Their results indicate, for an information seeking task about a specific subject, that users prefer to use a search interface, but for an information seeking task that asks for a general subject they prefer the tag cloud. Other results are:

- The tag cloud is good for browsing or non-specific information discovery and reduces the cost of a query.
- The tag cloud can present a visual summary of the database.
- The tag cloud takes less cognitive load than formulating specific query terms.

Rivadeneira et al. [28] conducted two experiments on tag clouds. The first experiment used tagclouds with Font Size (High, Medium and Low), Quadrants (Upper-Left, Lower-Left, Upper-Right, Lower-Right) and Proximity-to-the-largest-font (Adjacent, Non-Adjacent) as independent variables, and thirteen participants. The result of this experiment shows a strong effect of font size, while proximity-to-the-largest-tag does not have a strong effect. In addition, results show that tags in the left upper corner are referred to frequently by users than tags in other locations. In the second experiment, Rivadeneira examined the effects of font size and tag cloud layout on impression formation and recognition. He used tag layers with different font size and layouts

(Sequential with Alphabetical Sorting, Sequential with Frequency Sorting, Spatial Layout (Feinberg's algorithm) and Single Column List with Frequency Sorting). This experiment showed a significant effect of font size, but tag cloud layouts have no impact on recognition.

Lohmann [14] conducted an experiment to evaluate user performance with three different tag cloud layouts. Below, we briefly describe the experiment and its results. The authors identify several key decisions made in their experimental design:

- Using a common number of tags and font size variations across tag cloud layouts.
- Using tag cloud layouts that differ only in tag arrangement not in other visual properties.
- Using eye tracking to measure the actual attention areas and perception patterns of tag clouds.

Three search tasks were chosen:

- Finding a specific tag
- Finding the most popular tags
- Finding tags that belong to a certain topic.

And finally authors used three types of tag clouds:

- Sequential layout: It uses inline HTML elements to represent the tags. The most popular sequential layout is a rectangular tag arrangement with an alphabetic order.

- Circular layout: It uses the font-size and the position of tags from the center of the orbits to show the relevance of the tags.

- Clustered layout: It clusters the tags semantically therefore similar tags will be placed near each other.

In addition, they created a fourth tag cloud with no variation in the tags' font sizes as a reference layout [14].

Lohmann concluded from his user studies that an optimal solution for arranging weighed terms is depended on the specific user goals and the intention of designer, and there is not a best solution that works for every design. This study inspired us to design our own arrangement system tailored for our application.

CHAPTER 3

TAG CLOUD AND GEO-TAGS

Tag clouds are boxes containing lists of tags with the most popular tags having bigger font size or other more detectable visual properties. A combination of tag clouds and a map with a zoom-and-pan interface provides a new tool for visualizing data. Geo-tagging is called the process of adding geographical identification to different media such as interactive maps. This idea of geo tagged data backs to 1977 Milgram's work [29]. We use the word geo-tag to refer to combination of tags and geographic maps. Tag clouds are text-based visual representations of a set of tags which usually depict tag importance by font size [22]. Interfaces use visual properties, such as color, shape, font type and font size in tag cloud layouts to produce more effective and attractive tag cloud layers.

In this chapter, we review the tag cloud aspect of our application for using tags on an interactive map, and identify challenges we faced while developing the application.

3.1 System overview and features

Google Maps has provided APIs for its services that let users access Google Maps in their own websites and applications. Using the Google maps API, our system distributes packed tag clouds into a geo-spatial arrangement that conveys textual information that is placed both in space and time. We propose that the idea can be developed in different ways:

- Hierarchical Geo-tag Clouds: a map with a Zoom in and Zoom out interface can allow tag clouds in different layers. As one zooms into a geographic area, tags may break-up into smaller, more detailed words.

- Mixed Media Tag Clouds: the inclusion of images can make tag clouds more visually engaging; in addition, more information can be displayed to the audience in a shorter time.

3.2 General issues in tag cloud design

Below we list the general issues in designing a tag cloud layer. Later in this chapter we review possible solutions for each of them in our system.

(i) Analysis of the geographical region:

   a. Geographic regions can have different shapes.
   b. Geographic regions can have different size, for example a city, province, country can be the geographic target for the application.
   c. Geographic regions can have different levels of importance for the application. For instance, some cities are more important than other less populated areas.

(ii) Tag placement and tag location:

   a. Having a mechanism for the association of the tags with the positions, for example a tag layer that shows the news story on a map, should put tag in related location.
   b. Tags can be freely placed, but they must be inside the geographical region.

c. Overlaps between tags, or crossing them into unrelated regions, can make tags un-readable and ineffective; and in some cases, it can give users the wrong perception of the place.

(iii) Tags visual effects

a. Flexibility in size, color, orientation, and transparency.

b. Tags should be readable and make the application visually attractive.

(iv) Tag density

a. The number of tags is not fixed and it depends on the actual shape and size of the region. In addition, depending on the subject, tags can have different distributions in different regions.

b. Finding a mechanism for handling a large number of tags.

Below we briefly review each issue, and discuss our solutions and suggestions to address them in our system.

3.2.1 Working with geographical regions

Geographical maps with zooming and panning features have a wide variety of geographic representation for different applications. For example, in Figure 8, you can see a geographic map of animal distributions (left) and a geographic map of forest distributions (right) in the U.S. This wide range of styles with different shapes and sizes becomes a challenge for tag maps to produce an appropriate tag layer over a map A tag map application should be able to create layers in simple and complex shapes, and process them for different numbers of tags and ROIs with different size and length to find a location to put tags and ROIs on the map in a visually attractive way.

Figure 8: Geographic maps with different applications[4], left animal distribution in US, and right forest distribution in US.

We are using Google Maps as a powerful platform for our application. Google Maps provides an API[5] with many features for map developers. One of these features lets users have their own overlay on top of the map. To use this feature, it is enough to send arbitrary geographic coordinates to the Google Map servers to draw the overlay on top of its maps. For example, to define an overlay on top of the province of Ontario, we have to send the geographic coordinates of political borders of Ontario to Google Map servers. Figure 9 shows a visible gray overlay (40% transparency) on top of the province of Ontario.

[4] http://www.infomercantile.com/blog/labels/wildlife.html

[5] https://developers.google.com/maps/documentation/, https://developers.google.com/maps/documentation/flash/

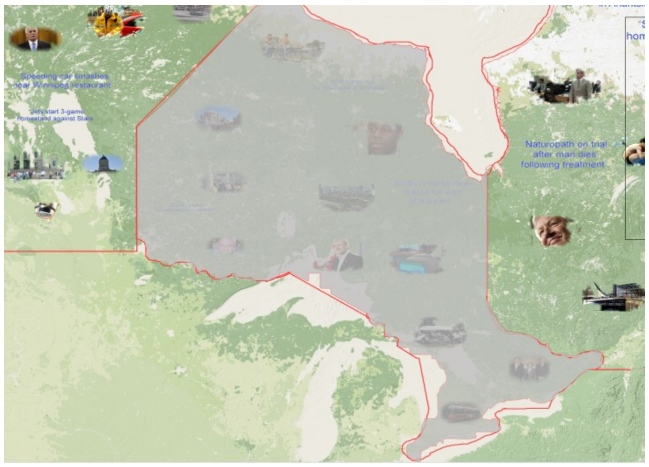

Speeding car smashes near Winnipeg restaurant
Naturopath on trial after man dies following treatment

Figure 9: Sample overlay on top of the province of Ontario.

Our application can have a wide variety of uses in different subjects, such as tourism, eco-tourism, sports and education. However, we have chosen to focus on visualizing news over the geographic maps. This means we need to have overlays on political regions, such as cities, provinces, and countries. In our first tag cloud application, we visualized news over an U.S map and later we moved to Canada. To provide political border coordinate for Google map servers, we created a database from Geopolitics information provided by U.S[6] and Canadian[7] governments. However, geographic coordinates were not provided for all cities and we had to estimate the geographic locations for some cities.

[6] http://www.census.gov/geo/www/cob/

[7] http://geodepot.statcan.gc.ca/2006/040120011618150421032019/02152114040118250609120519_05-eng.jsp?fileType=DBF&format=G&language=E&geo=PR&fileName=&Agreed=I+AGREE&NextFile=Return+to+file+selection

3.2.2 Tag placement

One of the most important requirements of tag map applications is having an effective mechanism for associating tags with geographic positions. As we mentioned in Chapter 1, there are many algorithms for creating tag maps, but most of them suffer from unreadable tags because of tag overlapping and wasted unused white spaces because of weak tag arrangements. A good application should let tags to be freely placed anywhere inside the overlay without overlapping with other tags or hiding map features.

As the shape of geographical regions is irregular, we cannot use usual tag placement techniques. In most of the tag placement techniques, after sorting tags based on their defined weights or alphabetic order, tags will be linearly placed one after another or symmetrically arranged around a simple shape region such as a rectangle or square. In our case, we usually have complex geographic areas, and tag location is heavily dependent on the available space in those regions. To develop an algorithm for tag placement in arbitrary shapes, we needed to find a way to calculate the area and centroid of a polygon. Below we briefly explain Bourke's method[8].

[8] http://paulbourke.net/geometry/polyarea/

Bourke’s method

A polygon is defined by its vertices and edges (see Figure 10). In a polygon with N vertices, with (x_i, y_i) for $i = 1\ to\ N$ the area is given by:

$$A = \frac{1}{2}\sum_{i=1}^{N-1}(x_i y_{i+1} - x_{i+1} y_i) \qquad (7)$$

Consider a scenario where there is a big area in a chosen geographical region that we prefer to remove from calculation. For example, there may be a big lake inside a province and we do not want to count it as a space for placing tags. In this case, we have a polygon with holes. For a polygon with holes, the holes are defined by ordering the vertices of the enclosing polygon in the opposite direction to those of the holes. Therefore, hole areas have an opposite sign to the bounding polygon area. By adding areas to each other, we will have the area of the polygon with the hole. If vertices are ordered counter clockwise the sign will be positive, otherwise it will be negative. This technique will not work for self-intersecting polygons (see Figure 10, right). However to solve this problem we treat each enclosed region as a single polygon and then we combined the result:

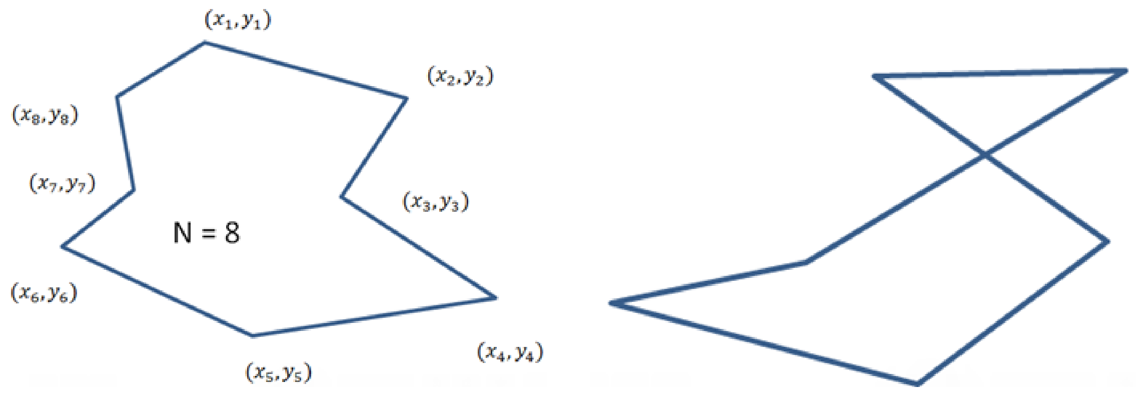


Figure 10: Left: an arbitrary polygon. Right : a self-intersecting polygon.

The centroid (center of gravity or mass) is helpful for distributing tags on the surface of a polygon. To calculate the centroid, we have:

$$C_X = \frac{1}{6A}\sum_{i=1}^{N-1}(x_i + x_{i+1})\,(x_i y_{i+1} - x_{i+1} y_i) \qquad (8)$$

$$C_Y = \frac{1}{6A}\sum_{i=1}^{N-1}(y_i + y_{i+1})\,(x_i y_{i+1} - x_{i+1} y_i) \qquad (9)$$

Finding a location for a tag

Having an effective method for finding suitable positions in a region to place tags, is an important component of tag cloud algorithms. There are different strategies to solve the tag placement problem. In one strategy, a geographic region can continuously break into smaller areas. Each subarea can be a place for a tag. With this approach we will have an area labeling problem with two important challenges [17]:

- How to subdivide the region?
- A region should subdivide into how many sub regions?

Another strategy is to use the topological form of the region to find the tag position, for example using the region's geometrical skeleton. This strategy changes the problem to a point labeling problem for specifying the appropriate positions along the skeleton. A purely random strategy is another option, but repeating the process after each fail result can take excessive computation time.

3.2.3 Tag visual effects and tag density

Tag cloud algorithms usually do not use a fixed number of tags. The number of tags usually depends on the size and shape of the region or the exploration process. A strong tag cloud application should have an effective mechanism to handle different numbers of tags, especially when the number of tags is greater than the available positions. A common design of tag clouds usually supports less than 100 tags in a layer (e.g. 30 tags [22], 93 tags [21], 76 tags [30], or about 100 tags [14]). Tag cloud research [31][26] shows people usually do not see a tag cloud as a rich source of information, and they often scan tag clouds for the highlighted tags instead of reading all of them to find information.

We think that the system can use two possible solutions. In the first solution the algorithm can render the first important tag on to the map and then look for a suitable place for the next one. This will continue until the algorithm cannot find any more locations for the remaining tags. The second solution can include pre-processing on the region for estimating possible locations, and post-processing for possible remaining locations. Particle-based labeling [32] can be a useful method for this approach.

To have acceptable tag visual properties and density in a geographic region, the shape and size of the geographic region, tag density and visual properties should be considered as a part of the placement algorithm.

3.3 Keyword extraction

Our system uses tagging to show news articles and their keywords over interactive maps, therefore we believe that, it is needed to briefly review some of keywords extraction techniques, for future work on this part.

Many text-mining applications, like search engines and text categorization systems operate based on keyword extraction. In this section, we concentrate on methods for extracting keywords from online news articles to produce a short summary of them. Most of the news articles contain geographic content, which can help readers to find related news in different geographic areas. Extracting this geographical information, in addition to keywords extraction, can be used for geo-tagging software. We review two methods for keywords extraction and a method for geographic location extraction from news articles that we used in our application. We note, however that keyword extraction is not an area of novel contributed for this thesis, but the keyword extraction algorithms that we are discussing could be a foundation for further research in area of text processing.

3.3.1 Topic extraction from News using the TF-IDF and TF*PDF algorithm

Sungjick [4] introduces a model for extracting the most important keywords from a news domain. The problem can be addressed as follows:

1. $D = \{d_1, d_2, \dots, d_m\}$ $\mathrm{D} = \{\mathrm{d}_1, \mathrm{d}_2, \dots, \mathrm{d}_\mathrm{m}\}$ is a set of documents from news websites.
2. $T_j = \{t_{j1}, \dots, t_{jn}\}$ is a set of terms, which are extracted from single document (d_j).

3. Term $T = \{T_1, T_2, \dots, T_m\}$ shows a set of extracted terms from a document set (D).

Extracted words included valuable and uninformative words (i.e. 'an', 'the'). Sungjick tries to extract valuable words (T) after giving weights to extracted keywords and sorting them. Below we briefly review the algorithm.

The term frequency–inverse document frequency (TF-IDF) weighting model is one of the common algorithms for information retrieval and text mining applications to evaluate the importance of words in a document. Term Frequency (TF) value indicates the more frequent words in a document. To prevent a bias towards longer documents vs. shorter documents for measuring their level of importance of words, the TF value is usually normalized by using:

$$\mathrm{tf}_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \quad (10)$$

$n_{i,j}$$\mathrm{n}_{\mathrm{i,j}}$ shows the number of occurrences of the considered term in a document d_j and $\sum_k n_{k,j}$ $\sum_\mathrm{k} \mathrm{n}_{\mathrm{k,j}}$shows the number of occurrences of all terms in document d_j. The IDF value shows the rarity in the whole collection. For example, words like 'today', 'the', appear a lot in documents, and they should be regarded as uninformative words. The IDF equation is:

$$\mathrm{idf}_i = \log \frac{|D|}{|\{d_j : t_j \in d_j\}|} \quad (11)$$

$|D|$ shows the total number of documents in the corpus and $\left|\{d_j: t_j \in d_j\}\right|\left|\{\mathrm{d_j}: \mathrm{t_j} \epsilon \mathrm{d_j}\}\right|$ shows the number of documents where the term appears. Finally for the TF_IDF we have:

$$\mathrm{tfidf}_{\mathrm{i,j}} = \mathrm{tf}_{\mathrm{i,j}} \times \mathrm{idf}_{\mathrm{i}} \tag{12}$$

The collected terms and their weight that have been calculated by using the TD-IDF algorithm are sorted in a temporary table, which is called the Table Term Frequency (TTF).

Ishizuka [5] used the TF*PDF (term frequency–proportional document frequency) algorithm to extract words which described the main topics from weekly news archives. Additionally, he defined a sentence vector for each sentence. A sentence vector consists of unit vectors, which are the terms with the highest weight in an article. Sentence vectors are used for clustering sentences into their topics and forming a summary of topics. After and during the process, vector sentences are categorized in four groups [5]:

1. CS (cluster sentence): sentence clustered to a certain topic correctly.

2. MS (miss sentence): sentence clustered to a certain topic but incorrectly.

3. FS (fail sentence): sentence belonging to an existing topic cluster but fails to be clustered.

4. NC (not clustered sentence): sentence not belonging to any existing topic cluster.

After clustering, many sizes of clusters can be produced and summarization techniques are used to reduce the length of summary text [33].

Ishizuka [5] recognized the fact that prominent news usually is discussed in most of the local or global news sources, and terms that explain the prominent news should be used frequently in those source wires. After giving equal importance to information from each source, TF*PDF is applied to find the common words with a term heavily weighted in the majority of channels. These terms should be the main words to explain the main topic. Next, we explain the TF*PDF algorithm:

In the TF*PDF algorithm, the weight of a term is calculated and normalized as follow:

$$\mathrm{w_j} = \sum_{\mathrm{c=1}}^{\mathrm{c=D}} \mathrm{F_{jc}} \exp\left(\frac{\mathrm{n_{jc}}}{\mathrm{N_c}}\right) \tag{13}$$

$$\mathrm{F_{jnorm}} = \frac{\mathrm{F_{jc}}}{\sqrt{\sum_{\mathrm{n=1}}^{\mathrm{K}} \mathrm{F_{nc}^2}}} \tag{14}$$

$W_j =$ Weight of term j; $F_{jc} =$ Frequency of term j in channel c; $n_{jc} =$ Number of a document in channel c, where term j occurs; $N_c =$ Total number of documents in channel c; $k =$ Total number of terms in a channel; $D =$ number of channels. TF*PDF consists of three major parts:

1. The first part is the summation of a term's weight gained from each channel. Using more channels can provide more accuracy in recognizing a term that explains prominent topics.

2. The second part is normalization of the term frequency of a term in a channel $|F_{jc}|$, as shown in Equation 2. This normalization eliminates the effect of the document's length in finding important words.

3. The third part is the PDF (proportional document frequency) of a term in a channel $\exp(\mathrm{n}_{\mathrm{jc}}/\mathrm{N}_{\mathrm{c}})$; this means that words that occur in more documents are more valuable than words that occur in less documents.

We tried to use TF-IDF and TF*PDF methods during our implementation; however text processing and keywords extraction is an independent research area and needs separate investigation. In addition, in the final version of the application, we decided to show the whole news header on the map. Therefore, we investigated keyword extraction as a second priority and future work; and we removed these algorithms from the final version.

3.3.2 Geographic content extraction

The application needs to have geographic information to put tags in the right place on the map. Below we describe four ways to find geographic information for news articles, and their limitations:

1. Using the geographic location of the publisher and geographic information about the news article provided by the publisher. Our application uses this as an indicator of the geographic location of news. Import limitations of this method are:

   a. Some news websites do not show the publisher's geographic location. Some of them just show a general geographic location, for example the name of the province, or important cities.

2. Publisher geographic location cannot always show the real geographic location of news. For example, federal news story belongs to different parts of the country; however, they are usually published in capital cities.

3. Finding words in an article that references geographic locations, (i.e. the story content's geography), is a problem that is known as Named-Entity Recognition (NER) in Natural Language Processing (NLP). We worked on this method in an early version of the application, but we have chosen to develop this method in future work because of the complexity of the method as demonstrated in NLP research. There are three limitations of this method:

   a. Geo/non-geo ambiguity: words could refer to a geographic location, or could refer to some other information;

   b. Aliasing: where multiple names refer to the same geographic location

   c. Geographic name ambiguity or polysemy, for example Halifax is name of a city in both Canada and England.

4. Using the reader's location

    a. This method tries to determine the location based on the majority of readers; but it cannot be a reliable method, especially for popular websites that have a wide variety of readers from different parts of the world.

5. Using the news concept to extract geographic information from the news articles. For example if a news story about the London Olympic games, probably is located in London.

3.4 Placement algorithms

In this section, we explain the methods we have used in our application. Our placement algorithm sees tags as rectangles with different sizes. The size of the rectangles depends on font size, font type and length of the words; these are calculated based on the importance of the keywords.

This section discusses details about the dividing algorithm and some concepts that we used from Seifert's [22] proposed algorithm for placing tags in a polygon. To have an attractive and effective tag layer, the tag placement algorithm should also be designed with regard to visual properties. After revieing [22][34][14] We have defined some constraints on the visual properties of a tag:

- The font size is not fixed, but θ_{min} and θ_{max} defines the minimum and maximum possible font size (too small tags are unreadable and too big tags are not visually attractive).
- The font family and color should be unique to the tag layer.

- All tags should be horizontal in orientation (for example for news, vertical tags are not attractive and are sometimes unreadable).

- Two distinct tags should not overlap.

- Tags with the same relevance value should use the same font size.

- Tags with higher relevance value use a larger font size.

- All r_i should be fully inside the polygon and no two r_i and r_j overlap for $i \neq j$,

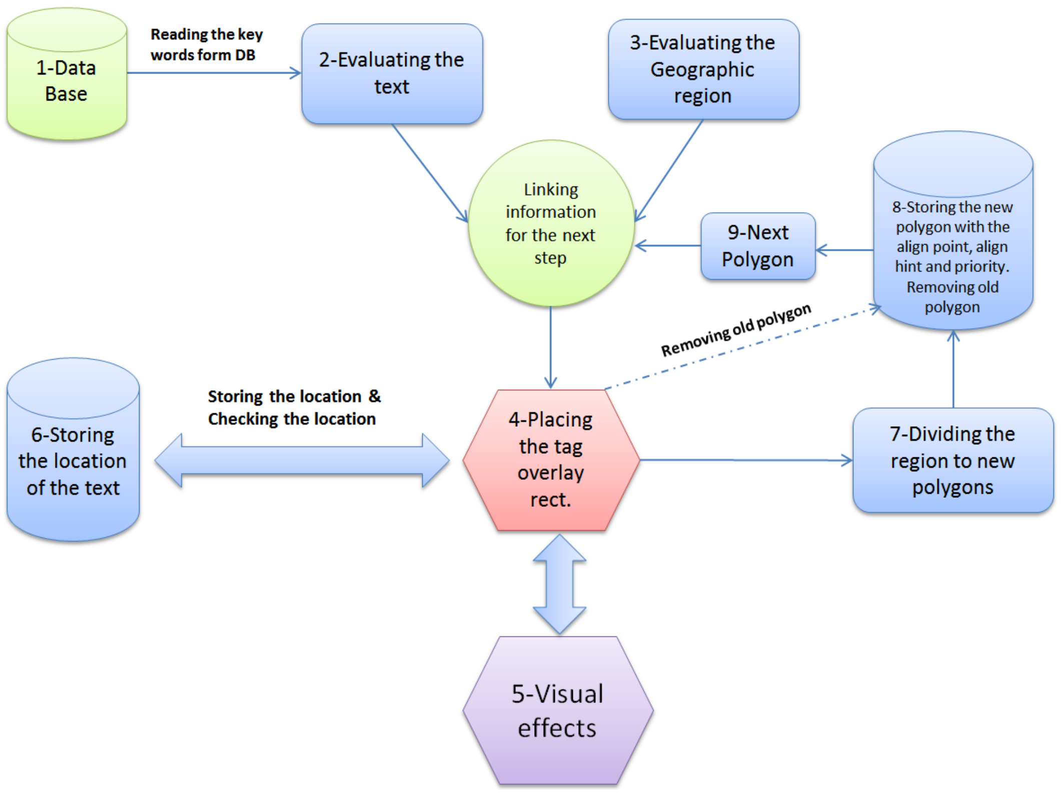

1-Data Base
Reading the key words form DB
2-Evaluating the text
3-Evaluating the Geographic region
Linking information for the next step
9-Next Polygon
8-Storing the new polygon with the align point, align hint and priority. Removing old polygon
Removing old polygon
6-Storing the location of the text
Storing the location & Checking the location
4-Placing the tag overlay rect.
7-Dividing the region to new polygons
5-Visual effects

Figure 11: Dividing Algorithm flowchart.

Figure 11 shows the flowchart of our dividing algorithm. A rectangle $r = (w, h)$ is known by its width (w) and height (h).

1- After extracting the keywords from target documents (in our case from the CBC[9] news website), the keywords are saved in a database.

2- The algorithm connects to the database to read keywords and their information such as, html news links, news summaries, geographic news coordinates, key word weights

(importance level) and zoom levels. After reading a keyword, it will be placed on the map.

3- Our application calculates the area of the region (A_T) by using Bourke's method. 4&7-The algorithm places the first rectangle on the center of the polygon. This rectangle split the polygon into four new polygons, which are located in the edges of the rectangle. Figure 12 (right) shows a region after placing the first rectangle.

5- Before placing the text overlay into the chosen polygon, the algorithm will check the previous text locations to prevent the overlay overlapping. In addition, after finding the location for the text, that location will be saved for later text placement checking.

[9] http://www.cbc.ca/rss/

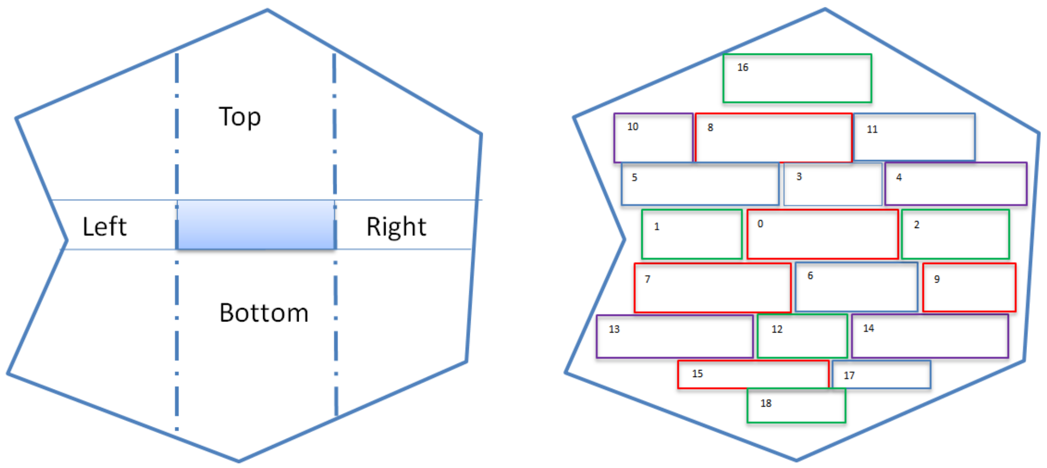


Figure 12: Right, each rectangle makes new polygons on its edges. Left, alignment example.

6- Our algorithm used a combination of three methods proposed in Seifert's [22] work, to make changes in the visual representation of tags and fix them in their locations. These methods are:

- The shift of the font size interval,
- The scaling of the font size interval,
- The truncation of tag strings.

Seifert [22] defined four combinations of these methods for using in his tag layer. Figure 13 shows the overview of the visual properties algorithm.

T is a set of all the tags, $T = \{t_i | \, 1 \le i \le n\}$. t_i^s is a tag string and t_i^w shows tag weight. θ_{min} and θ_{max} shows minimum and maximum possible font size. s_{min}^0 are the initial value for the font size which is set to θ_{min} and s_{max}^{0} shows max value for the font size which is set to θ_{max} . Below we can see these combinations:

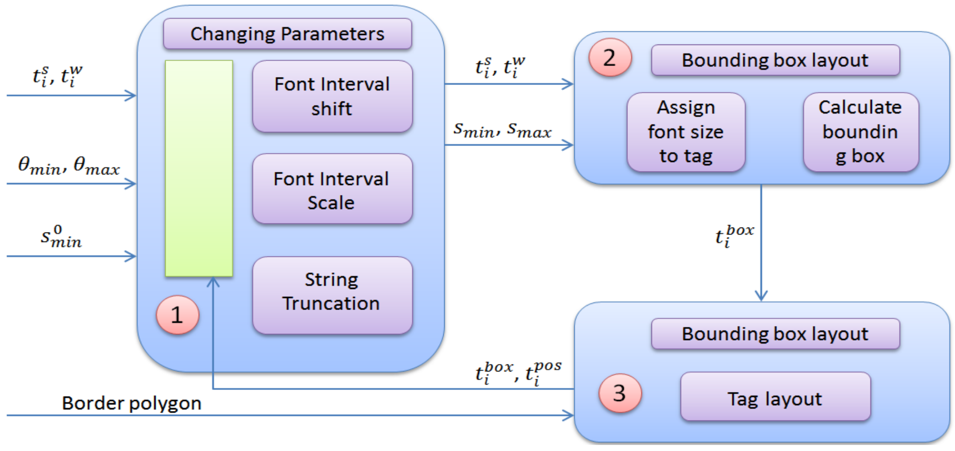


Figure 13: overview of the visual properties algorithm.

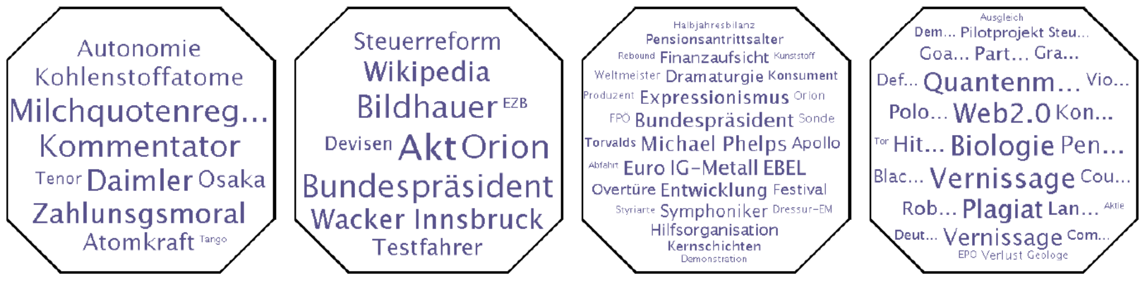
i. Shift-Trunc (ST): The algorithm first uses tag string shifting; if shifting was not successful, the algorithm truncates the length of the tag string. After one truncation.

ii. Shift-Trunc-Scale (STC): This acts like ST but if no truncation was possible, it tries to change the size of the tag.

iii. Shift-Scale-Trunc (SCT): The algorithm tries to use tag shifting first; if it is not possible, it tries to change font size and finally it tries to use string truncating.

iv. Trunc-Shift-Scale (TSC): This acts like SCT but the algorithm uses truncating first and after that it uses shifting and scaling.

Figure 14 shows sample results of these combinations. The authors [22] performed a technical and user evaluation on this algorithm to evaluate the reaction time (in milliseconds), correctness, beauty and helpfulness. The user study result showed that if correctness and reaction time play an important role for the application of a tag cloud algorithm, TSC is the preferred choice [22]. For applications with a strong focus on aesthetics one should think of using algorithm SCT [22].


Autonomie
Kohlenstoffatome
Milchquotenreg...
Kommentator
Tenor Daimler Osaka
Zahlunsgsmoral
Atomkraft Tango
Steuerreform
Wikipedia
Bildhauer EZB
Devisen Akt Orion
Bundespräsident
Wacker Innsbruck
Testfahrer
Halbjahresbilanz
Pensionsantrittsalter
Rebound Finanzaufsicht Kunststoff
Weltmeister Dramaturgie Konsument
Produzent Expressionismus Orion
FPÖ Bundespräsident Sonde
Torvalds Michael Phelps Apollo
Abfahrt Euro IG-Metall EBEL
Overtüre Entwicklung Festival
Styriarte Symphoniker Dressur-EM
Hilfsorganisation
Kernschichten
Demonstration
Ausgleich
Dem... Pilotprojekt Steu...
Goa... Part... Gra...
Def... Quantenm... Vio...
Polo... Web2.0 Kon...
Tor Hit... Biologie Pen...
Blac... Vernissage Cou...
Rob... Plagiat Lan... Aktie
Deut... Vernissage Com...
EPO Verlust Geologe

(a) ST (b) STC (c) SCT (d) TSC

Figure 14: sample results for combination of shift, truncation and scaling.(from [22])

We chose TSC to change tag size and find appropriate locations for tags. Like most of the algorithms, we see each tag as a rectangle. Briefly, TSC first tries to fit the tag in a rectangle by truncating it, if it is not successful, it tries to shift the rectangle up, down, left or right; we added four new directions to the corners for this shifting too; this gave us the opportunity to use space more efficiently and have less wasted space. Finally, if shifting is not successful, the algorithm changes the scale of the rectangles and the font size.

7 + 8- As we said before, each rectangle splits the polygon into 0 to 4 new polygons, which are located on the edges of the splitter rectangle. These new polygons will be stored in a list (L) with the align point P and an align hint (to find the next place for the next rectangle). When a polygon splits to new polygons, it will be removed from the

list (L). In addition, for each new polygon a priority value, which was calculated from the distance (d) of the center of mass to the center of mass of the original polygon, will be stored in the list. The algorithm uses this priority value to choose the next polygon.

9- For the remaining rectangles (keywords), the next polygons will be chosen and processed based on polygons priority values.

CHAPTER 4

IMAGE PROCESSING

According to psychology research, the right hemisphere of the brain processes information from the whole to the parts. In other words, it sees the "big picture" before the details [35]. It is the part of the brain that deals with subjective skills and creative abilities. This hemisphere of the brain helps solve problems via hunches, examining patterns and similarities and receiving information through passive watching [35].

We have used different methods and techniques to extract ROIs from different images, finding appropriate locations for them on the map and finally blending them into the map. This part of our work consists of three main stages: extracting the region of interest (ROI) from an image and converting text tags to images (tag image), finding a location on the host map for the extracted ROIs and tag images, rendering the regions of interest and tag images into the host map.

4.1 Extracting regions of interest

To provide ROIs on the map, first we needed to find a way to extract ROIs from different images with different contents. We tried active contours and object detection algorithms and we designed our novel method for extracting ROIs. In this section we review methods to solve the ROI extraction.

The ROIs is the name used to describe the meaningful and important part of an image. Detecting the ROI helps image processing, vision applications avoid the processing of irrelevant regions of an image, and make significant improvements in processing time. In our case, we find regions of interest to highlight the essential contents of a news story. To create an effective and attractive ROI we have used different methods such as active contours, object detection and face detection during the implementation of our system.

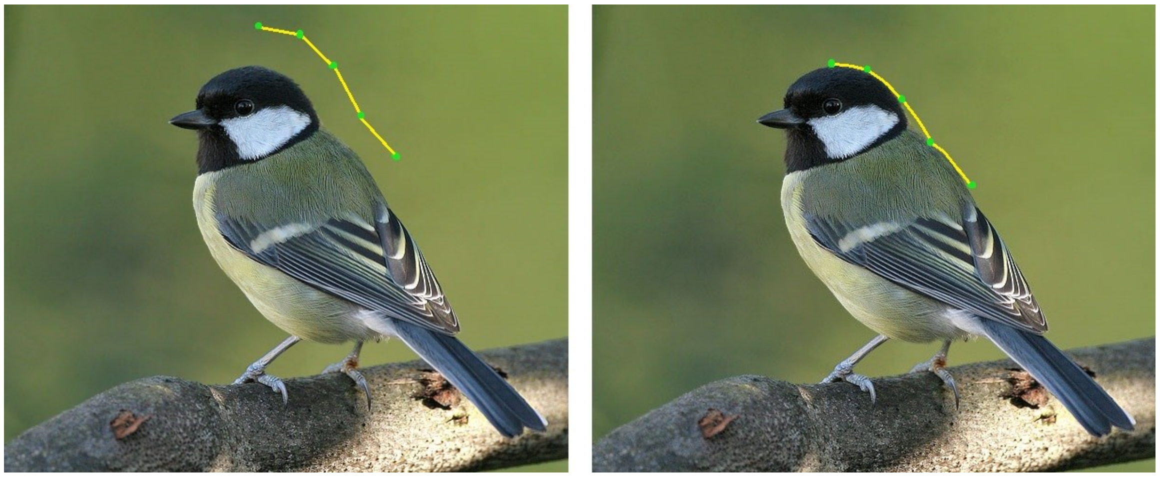


Figure 15: Simple active contour with five snaxel (green points).

4.1.1 Active contours

Active contours (known as snakes or deformable contours) was introduced by Kass et al. [36] as a segmentation technique for detecting edges and boundaries in vision and image processing problems. A snake is defined as an "energy-minimizing spline guided by external constraint forces and influenced by image forces that pull it toward features such as lines and edges" [36]. Figure 15 shows a simple active contour with 5 snaxels (Variable number of points of the contour[10] are called snaxels or snake elements [37]).

[10] distance between adjacent snake points

Image segmentation problems are mostly solved by defining and trying to minimize an energy function. In active contours, a parametric contour encloses an object to reach its boundaries by using the sum of internal and external energy where it tries to find the local minimum energy. External energy is usually used for determining the segmentation and boundaries while internal energy is used for regularizing noisy data to handle vague sections of the contour [38]. Active contour models can update their position and shape when the segmented object moves; therefore, it is possible to use active contour models for object tracking in different fields such as medical imaging and satellite image processing.

Kass et al. [36] used variational calculus to find and optimize the derivation of an object energy function for his algorithm. A variational approach needs estimates of higher order derivatives of the discrete data to guarantee the global optimality of the solution [39]. But limitations in calculating higher order derivatives makes the Kass approach less effective in practice. Later, Amini et al. [39] used a dynamic programming algorithm for minimizing the energy function and solving variational problems to ensure global optimality of the solution.

Radeva et al. [40] tried to use a gradient orientation of image edge points. They used a new potential field and external energy for deformation convergence near the edges. Prince [37] tried a variational approach to detect the final convergence of the snake; he proposed a new external energy model for active contours which is called Gradient Vector Flow (GVF). But "Active contours involve iterative processes, great algorithmic complexity, and high average run-time" [38]. An important issue around active contours' algorithms is the high number of iterations especially when the energy function traps the contour in a local minimal. In addition, the accuracy of the result depends on the convergence criteria that are used in the energy minimization function, where it requires more time for higher accuracies. All these make active contours inappropriate for use in applications that need fast processing time.

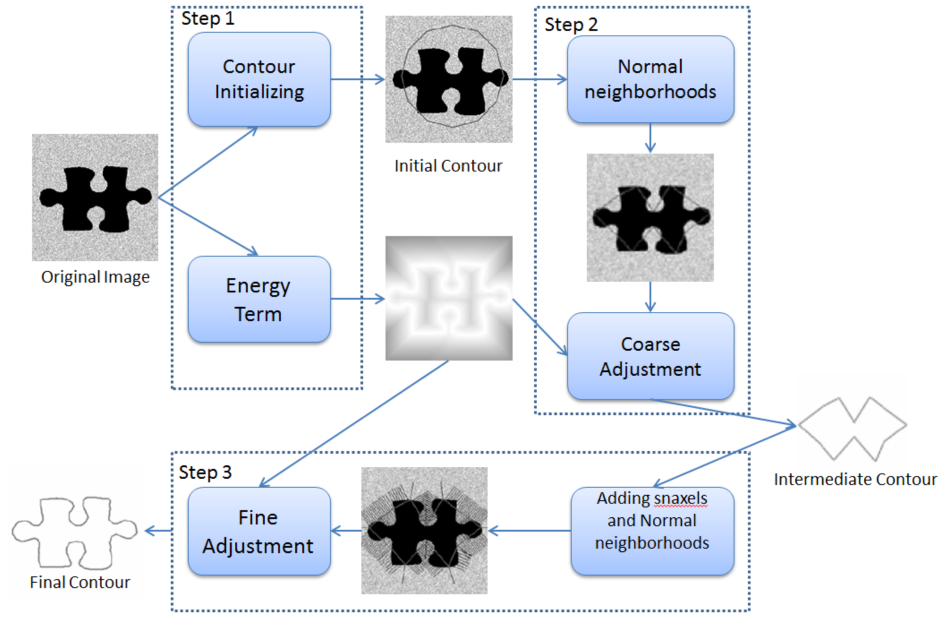


Figure 16**:** Flow chart of Cheng's algorithm.

A real time active contour

Cheng's [38] method operates based on a Coarse-to-Fine Normal Neighborhoods strategy (CoFiN2) which is considered a greedy algorithm. Cheng's [38] suggests that his algorithm is fully automatic and has a lower computational cost in comparison to other algorithms. Cheng's algorithm has three features:

- Using diffusion gradient images [39] as an energy function, which moves the contour to find the edges even if they are located at the center of the region.

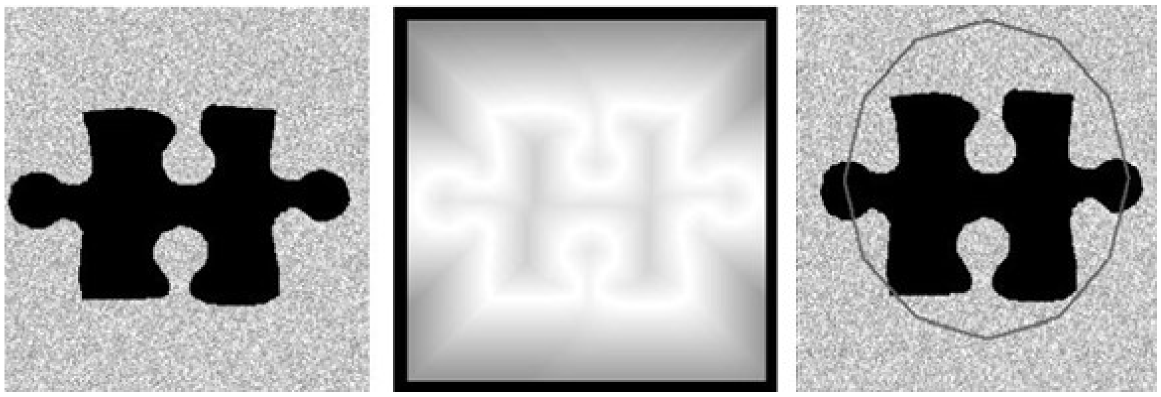
- Snaxels (snake elements) allow the snake to work with evenly spaced snaxels to recognize objects with high accuracy.

- Normal neighborhoods to improve convergence into concavities.

Cheng's model has three main stages. Figure 16 shows these three stages in more detail:

Pre-processing

Step 1 consists of computing the image energy and placing an initial snake. Image energy is seen as a gradient value, where high values show boundaries and low values indicate the places that snaxels cannot find a correct way to expand. Cheng used a distance transformation [41] to compute the energy term to force snaxels to expand the



(a) (b) (c) (d)

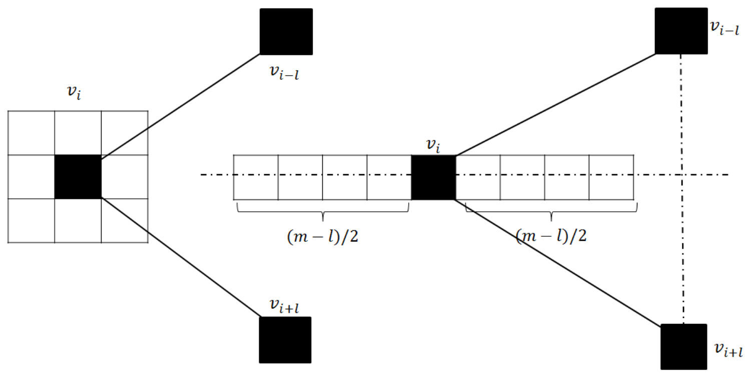
Figure 17: (a) Original image and (b) Diffusion of the gradient (c) Possible initial snake for images for n = 12 snaxels. Original image used as Image Energy term.

contour and uses a simple circular contour for an initial snake. Figure 17 shows the diffusion of the gradient and possible initial snake with 12 snaxels.

Coarse adjustment

In step 2, a normal neighborhood (in differential geometry, normal neighborhood at a point p in a differentiable manifold is a symmetric affine connection in a neighborhood of p) is built up for the snaxels and the intermediate snake is placed in the object boundaries (coarse adjustment) by moving the initial snake from its original position (intermediate contour).

The algorithm defines a neighborhood for each snaxel, and a contour evolution for the intermediate snake. Figure 18 (a) shows the classical square neighborhood; a square neighborhood of points (m_i) is defined for each point of the snake (v_i). Figure 24 (b) shows the normal neighborhood; In CoFiN2. there is a linear neighborhood for every v_i, which is defined in the vertical direction to its tangent line [38]. Cheng used a greedy algorithm to move v_i to a new point with the lowest energy level with the hope that a greedy algorithm can reduce complexity and time cost.



(a) (b)

Figure 18: (a) Classical square neighborhood, $m = 9$, and (b) normal neighborhood (N2), $m = 9$.

Fine adjustment

Step 3 includes fine adjustment to the contour. The number of the snaxels affects the accuracy of edge recognition; by increasing the number of snaxels. The algorithmwill have slower evaluation but higher accuracy. This means, choosing the right number of snaxels plays an important role in obtaining a result with good accuracy and in an acceptable time. For adding and removing snaxels, based on bilinear interpolation two distances d_{max} and d_{min} are defined. In mathematics, bilinear interpolation is defined as an extension of linear interpolation for interpolating functions of two variables on a regular grid[11]. If the distance between two sequential points contravene d_{max} and d_{min} , a

[11] http://en.wikipedia.org/wiki/Bilinear_interpolation

new point is added or removed, respectively [38]. Figure 19 and Figure 20 shows our sample results of running the algorithm. In figure 25: At the top left, we can see image with initial contour. At the top right, the external energy is shown in light and dark colors, dispersion in color shows the dispersion values of energy. At the bottom left, the external force field (using gradient around the contour position) shows object boundary position. At bottom right: The snake movement is shown in green.

4.1.2 Object detection

In addition to edge based processing, we may also try to identify specific objects. Viola-Jones [42] is one of the popular object detection algorithms. It uses a learning algorithm based on AdaBoost (a weak classifier that repeatedly calls a weak learner on a weighted training set) and a method for combining classifiers into cascades to quickly remove the background of an image and detect the target objects inside the image.

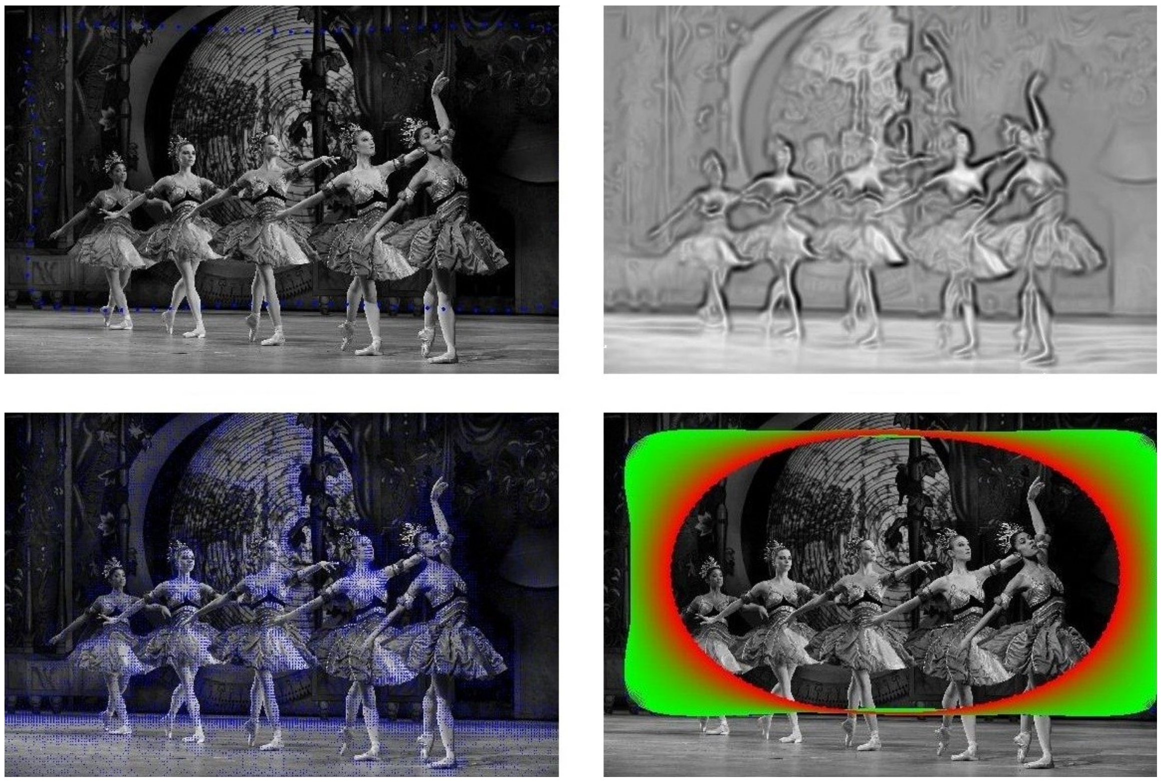


Figure 19: Top left: Image with initial contour. Top right: The external energy in light and dark colors. Bottom left: The external force field. Bottom right: Snake movement.

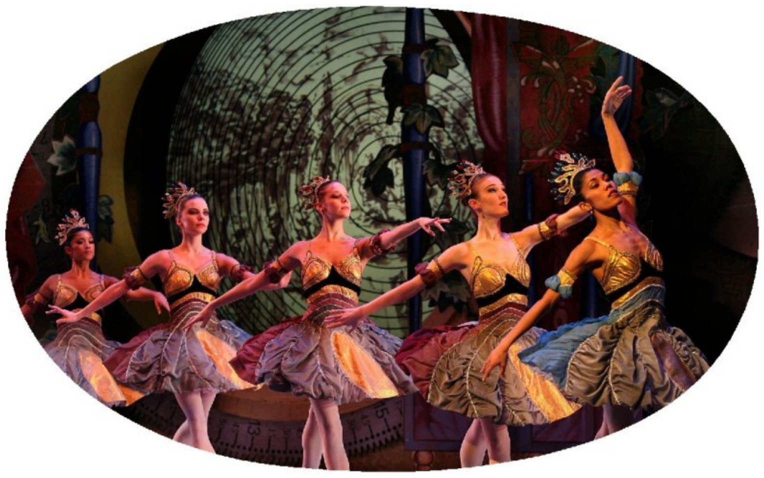


Figure 20: Region of interests after using active contours.

A learning process can generate many classifiers, but evaluating all of them is not feasible in real-time processing. To solve this problem, strong classifiers are managed in a hierarchical cascade. Cascading is built based on the concatenation of different classifiers, where the result and information of a given classifier is used for the next classifier in the cascade. These classifiers are trained only on the samples, which are passed through the preceding classifiers. Cascade architecture is like a tree; if at any level in the cascade a classifier rejects the result, the search continues on the next sub-window. To explain, Haar wavelets were used in the first real-time face detector. Viola and Jones [42] used Haar wavelets in their development and called them Haar-like features. A Haar-like defines detection windows (i.e. 24x24 pixel windows) known as sub-windows. In each sub-window an adjacent rectangular regions is being considered. The pixel intensities in each region are added to each other. Then the algorithm calculates the difference between these sums to use for categorizing subsections of an image. A Haar-like feature is known as the difference of the sum of pixels of areas inside the rectangle, with arbitrary position and scale[12] [13]. To have a faster operation, Viola-Jones uses value of simple features. In Figure 21 we can see three different features that Viola-Jones uses[42]:

- A two-rectangle feature uses the difference between the sum of the pixels of two rectangular regions with the same size and shape.

[12] http://en.wikipedia.org/wiki/Haar-like_features

[13] http://en.wikipedia.org/wiki/Viola%E2%80%93Jones_object_detection_framework

- A three-rectangle feature uses the sum within two outside rectangles subtracted from the sum in a center rectangle.
- A four-rectangle feature uses the difference between diagonal pairs of rectangles.

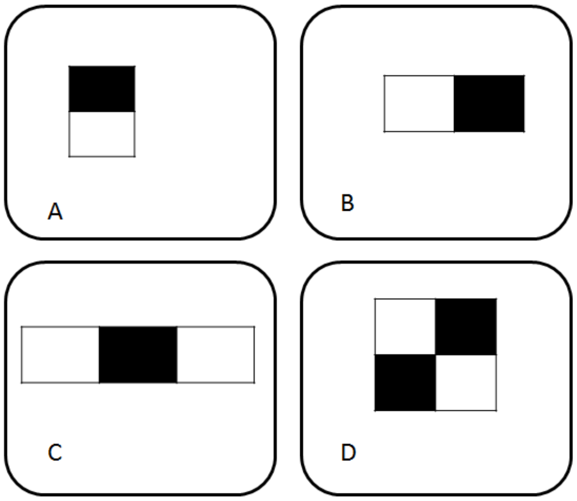


Figure 21: A&B: two-rectangle feature, C: three-rectangle feature. D: four-rectangle feature.

We used the Viola-Jones algorithm to detect faces on our images to help the application to detecting the region of interest and prevent the cutting faces during the extraction of the region of interest. Figure 22 shows sample results of running the Viola-Jones algorithm.

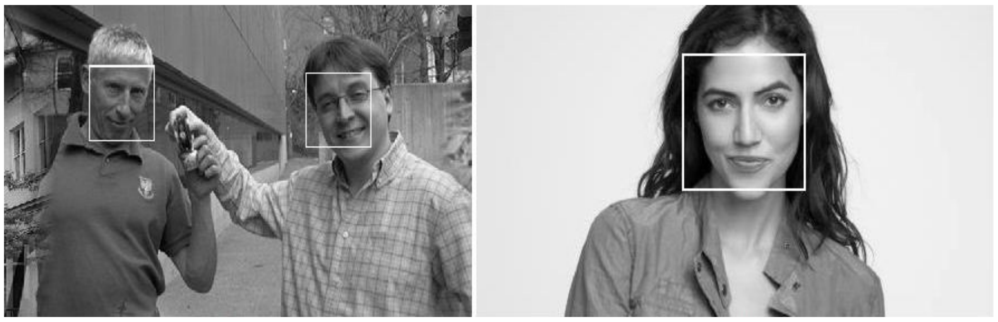


Figure 22: Sample results of Using Viola-Jones algorithm.

4.1.3 ROI extraction with the fourier transform

In our system, we have used different methods for extracting regions of interest, especially active contours and object detection algorithms. However, none of these options provided what we required for our application. Figure 23 shows an example of a failed result of running Viola-Jones algorithm. We therefore implemented a method to process an image in several stages that employed canny edge detection and frequency high pass filtering to find the ROI.

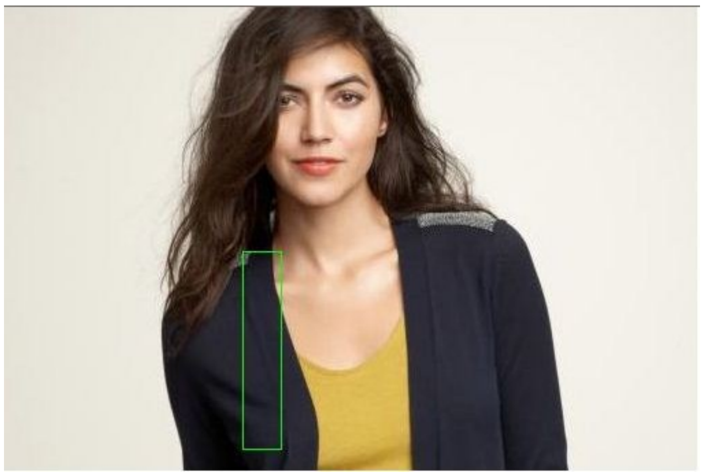


Figure 23: A sample failed result of running Viola-Jones algorithm.

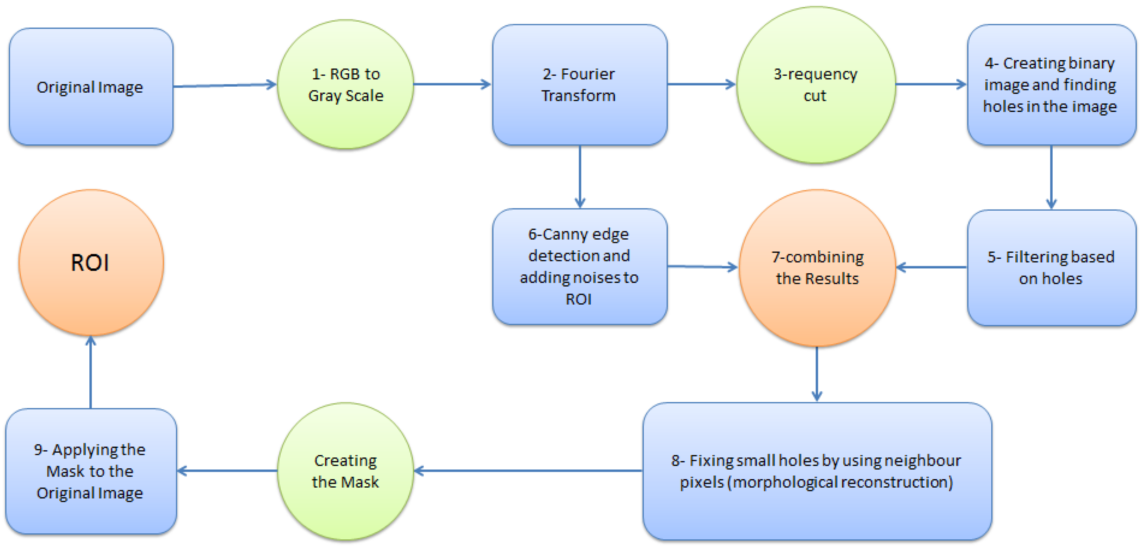

Original Image
1- RGB to Gray Scale
2- Fourier Transform
3-requency cut
4- Creating binary image and finding holes in the image
5- Filtering based on holes
6-Canny edge detection and adding noises to ROI
7-combining the Results
8- Fixing small holes by using neighbour pixels (morphological reconstruction)
Creating the Mask
9- Applying the Mask to the Original Image
ROI

Figure 24: the mask algorithm flowchart.

Figure 24 shows the flow chart of our algorithm for ROI extraction, and we will now discuss each key step:

1. After reading an image from the database, in the first step the image is converted from an RGB image to a gray-scale image. Each pixel of a color image is made from three colors (red, green and blue). By eliminating the hue and saturation information while retaining the luminance, the RGB image is converted to a gray-scale image.

2. In the next step, the algorithm computes the image's two-dimensional discrete Fourier Transform by using Discrete Fourier Transform (DFT) (equation 9):

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{k=0}^{N-1} f(x,y) e^{-2\pi i(\frac{ux}{M}+\frac{vy}{N})} \qquad (15)$$

This transfers the image data into frequencies for applying a high pass frequency filter, which produces an initial logical mask for the image.

The Fourier transform is an important image processing tool for decomposing an image into its sine and cosine components. The Fourier Transform has been used in many applications as a part of image analysis, image filtering, image reconstruction and image compression. The Fourier transform moves an image from the spatial domain to the frequency domain. In the Fourier domain image, points represent a particular frequency included in the spatial domain image. Figure 25 (a) shows the result of Fourier transforms on the image of a tiger.

3. A high pass filter then sharpens the image (keeps the details and edges) with the high frequencies of the Fourier transform being relatively unchanged, while the low frequencies are attenuated. Automatically finding the best cutoff frequency remains a challenge. Currently, we are using experimental values between 45 and 60, as a cut-off frequency. Figure 25 (b) shows the result of high pass filter transforms on image of a tiger.

4. In the next step, the algorithm converts the grayscale image to a binary image. A binary image has only two possible values (zeros and ones) for each pixel and it only uses two colors (white and black) to show the image. Using a binary image allows our program to perform a simple contour detection to find the boundary of objects quickly in the image.

5. The result of the contour detection in step four provides separate objects that we call holes. In the next step, the algorithm applies a filter on the result of contour detection. During the filtering, separated holes close to each other will be connected together to create a bigger hole. This hole is more likely to contain the ROI. In addition, holes far

from other holes are removed with the assumption that they are less likely to be a part of ROI.

6. In parallel with the use of a high frequency filter, we use Canny edge detection [43] to help the algorithm find the ROI. The Canny edge detection uses a multi-step procedure and finds most edges in the image. For this stage of processing, we define “good detection” as finding as many real edges in the image as possible. In addition, Canny uses a filter based on the first derivative of a Gaussian to remove noise. Both the Canny edge detection and the high pass filtering provide a kind of sketch of existing objects in an image. The Canny edge detector first uses Gaussian convolution to smooth the image and remove the noise. It then applies a 2-D first derivative operator on the image to highlight regions with high first spatial derivatives. This can be seen as a tendency for edges to give rise to ridges in the gradient magnitude image. In the next step, the Canny tracks along the top of these ridges to set the pixels on top of the ridges to one, and other pixels to zeros. This process is called non-maximal suppression. For the remaining pixels, the Canny uses hysteresis to set them to zero or one. Hysteresis uses two thresholds: pixels with a magnitude below the first thresholds are set to zero, and pixels with magnitude greater than the second thresholds are set to one. Pixels with a magnitude between the two thresholds are set to one, if they have a connection to other pixels with a gradient of more than the second threshold; otherwise they are set to zero.

7. In the next step, the results of previous stages are added to each other and create a new mask.

8. In the next step, the algorithm removes the holes in the mask by using a morphological reconstruction technique. Morphological reconstruction is defined as repeated of expansions dilations of an image that is known as a marker image; this repetition continues until the contour of the marker image fits under the original image; each expansion is constrained to lie underneath the mask [44]. Our algorithm limits morphological reconstruction to the area bounded by the ROI. Therefore, reconstruction only happens inside the ROI, and outside of the ROI will remain unchanged. Figure 25 (d) shows the result of Morphological reconstruction on the tiger image.

9. In the last step, the final mask will be applied to the original image to extract the ROI. This ROI will be used by the algorithm to blend in to the map. Next, we discuss the blending technique.

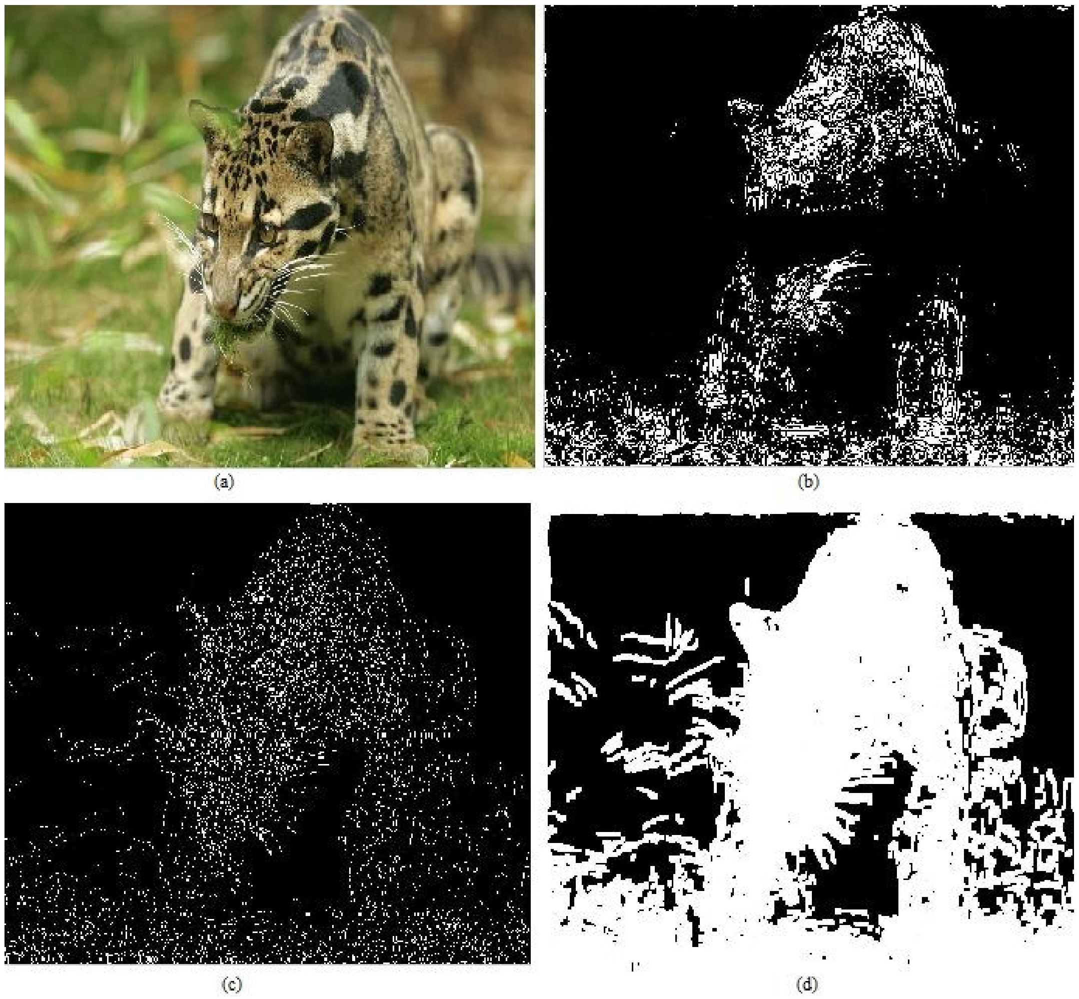


Figure 25: (a) Original image, (b) Frequency Filtering result, (c) Canny result, (d) Final Mask.

4.2 Locations and blending

We have developed Geo-Collage maps to combine ROIs, and tag images on geographic maps, as a novel news visualization system. We need to find an appropriate location on the geographic maps to render extracted ROI, in an attractive and effective way, which is also fast and reliable. We tried several image summarization techniques such as active contours, image Poisson blending and morphological reconstruction. However, most of the existing methods suffer from a slow processing time and/or low quality results (overlapped image, cut faces, etc.). Image summarization software uses a variety of methods for image placement. For example, Mediating Photo Collage [45] uses predefined templates for different image arrangements. In our system, we are using maps as the background for the extracted ROIs, and we need to define several limitations on image placement, as follows:

- Images should be placed close to locations where news events happened.
- Political borders should not cut faces and important objects.
- An image should be chosen that could represent the main information of a document.
- Collage pictures should represent as much information as possible.
- The blending between images and the background should be subtle and visually smooth.
- The photo arrangement should be optimized towards an objective function.

We are also using “saliency ratio balance” [46] to arrange image cuttings in the collage layer and to balance between visible parts of all extracted ROIs and tag images.

The aim here is to balance between visible parts of all extracted ROIs and the background map in order to create a balanced collage picture. Before giving more details about our final solution, we first quickly review on the image collage approach that we have in our application.

4.2.1 Related work in image collage

Wang in [46] tried to formulate picture collage as a Bayesian problem and used the Markov chain Monte Carlo (MCMC) sampling for the optimization. He argued that a good collage picture should have the following properties:

- A picture collage should show maximum visible salient regions.
- A picture collage should use all available space and minimize the blank spaces.
- Images in the collage should have a similar salience ratio (Salience ratio balance).
- Images in the collage could have different orientations (*orientation diversity*).

Wang [46] has used a visual attention model and a mix of different features such as texture, color and orientation for extracting salience regions from input images. He has used weighted rectangles to approximate the salient map. Figure 26 shows an example of weighted rectangles. We have replaced Wang's method of extracting ROIs with our own ROI extracting method.

For placement formulation, assume $\{I_i\}_{i=1}^{N}$ shows input images, $\{a_i\}_{i=1}^{N}$ shows their saliency maps and C is the host. Three variables S_i, L_i, O_i are defined for each image, where S_i is the center of image I_i on the host, O_i is the orientation angle and $L_i$is a layer index to determine the image.

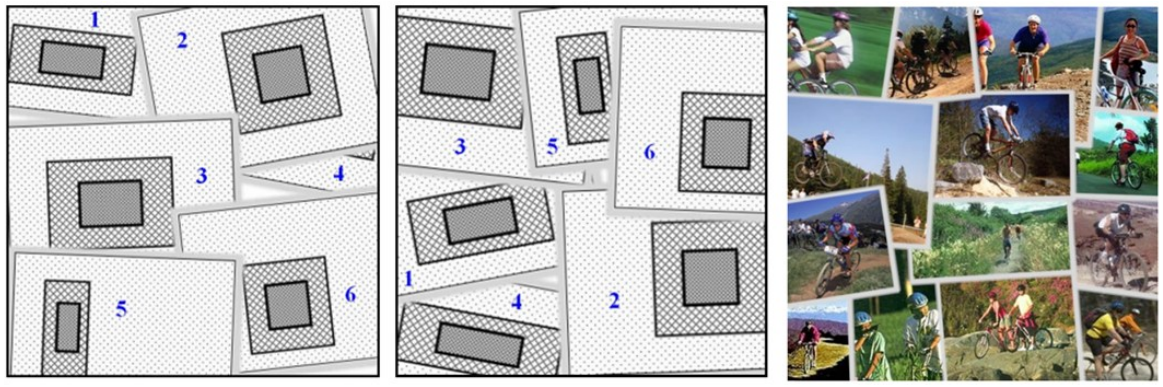
Wang formulated the picture collage creation in a Bayesian framework:

$$X^* = \arg\max p\,(z|x)\,p(x) \tag{16}$$

$p(z|x)$ is the likelihood to measure the quality of a picture collage which can be written as an exponential distribution:

$$p(z|x) = \frac{1}{Z}\exp(-(\overline{A_{occ}} + \lambda_B \overline{B}) - \lambda_V V) \tag{17}$$

Z is a normalization constant, A_{occ} is the normalized sum of occluded saliency regions, B is the normalized sum of uncovered regions on the canvas, and V is the variance of saliency ratios.



(a) (b)

Figure 26: (a) weighted rectangles (b) Wang results.

Carsten in [34] in a similar work to Wang [46], defined an automatic procedure for creating a collage from a set of input pictures $L = \{I_1 \ldots I_N\}$. Auto collage uses a blending technique to create more attractive collage layers especially when images are overlapping. Comparing Figure 26 (b) and Figure 28 shows this difference.

AutoCollage tries to find the best labeling $l \in L$, in the space L of possible labeling which minimizes an energy E (l).The energy of a labeling l is shown in four terms, as follows:

$$\mathrm{E(l) = E_{rep}(l) + w_{imp}E_{imp}(l) + w_{trans}E_{trans}(l) + w_{obj}E_{obj}(l)} \qquad (19)$$

E_{rep} tends to select the most representative image, in two senses: first it should be interesting and second it should be distinct from other chosen images to prevent duplication. The E_{imp} ensures that the (ROI) is selected from a chosen image. E_{trans} is a term which penalizes any unappealing transition between images and finally, E_{obj} incorporates information on object recognition.

Carsten's [34] approach can be summarized in four steps:

- **Image ranking:** based on E_{rep} images are ranked and relabeled statically. Therefore nth image will be selected greedily. After this step, the result will be passed to the ROI optimization.

- **ROI optimization:** Carsten's algorithm determines ROIs by selecting the rectangle that optimizes E_{imp}. Nevertheless, we are using our own method for extracting ROI.

- **Constraint satisfaction:** Auto collage uses a two-step approach to solve the packing problem, Figure 27 shows the details of this step:

    - The algorithm places rectangles as much as possible on the image without overlapping.

  - The rectangles are shifted to cover all pixels in the host; in this stage rectangles are allowed to overlap.

Our application tries to render ROIs on the map close to location of the news. In addition, it uses geographic coordinates of the location of the news, therefore we did not need this step in our application.

- **α-Poisson Image Blending:** The algorithm uses Poisson image editing [6] to create seamless fading in ROIs overlapping. An example result is shown in Figure 28.

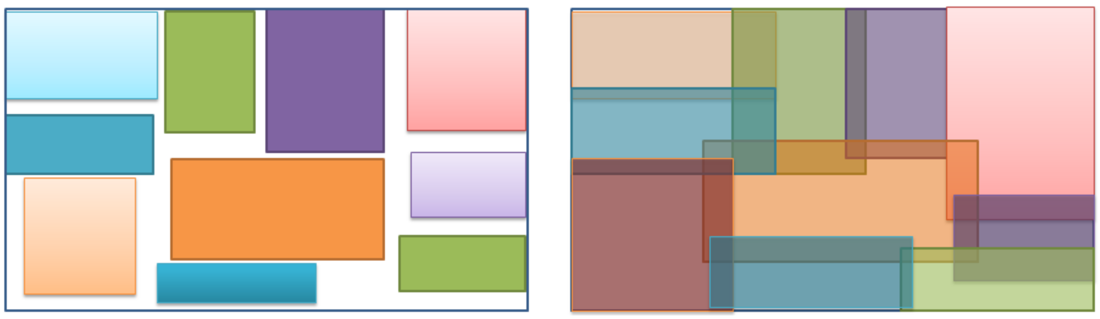


Figure 27**:** (Left) ROIs are shown packed without overlap. (Right) Images do overlap, and all pixels are covered.

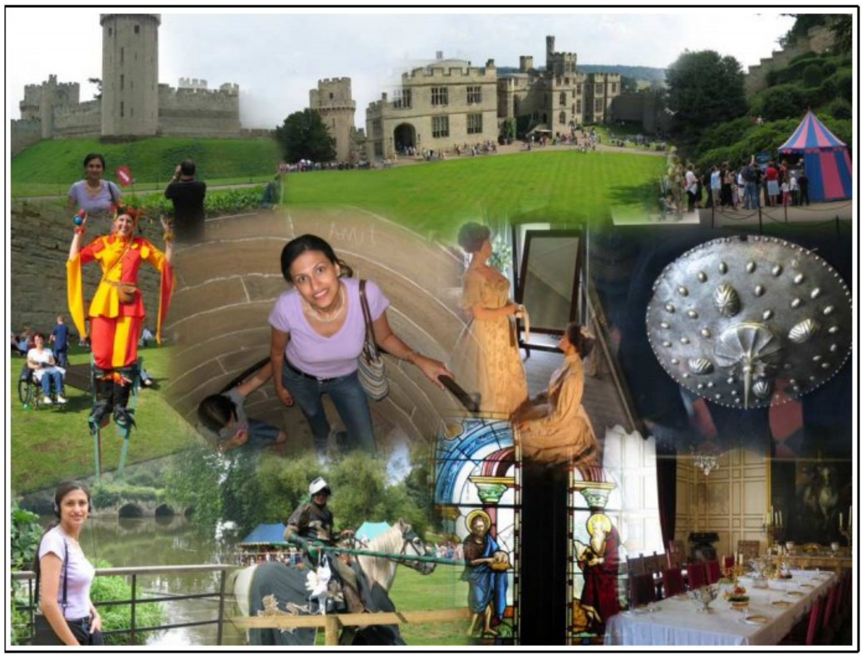


Figure 28: A result from the original auto collage algorithm.

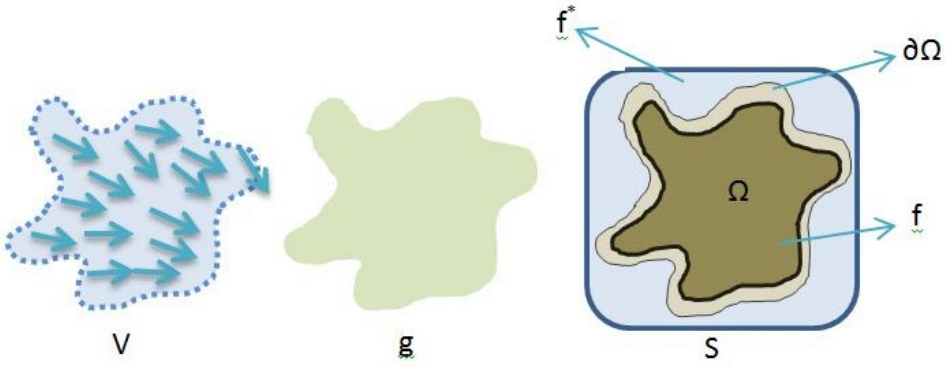


Figure 29: Guided interpolation notations.

4.2.2 Rendering the picture onto the map

For creating a seamless, clean transition between ROIs and the background map we use a combination of the Poisson image blending algorithm that is given by Perez [6] and our own averaging algorithm. By using Poisson technique on large images we experienced an overwhelming amount of time processing, in addition Poisson technique has unacceptable results (blurry result) on small images. Therefore, we developed average technique to use on small and large images.

In the Poisson image editing, algorithm the guidance vector field is used to solve the interpolation problem for the different colors in an image. Figure 29 shows Guided interpolation notations. Assume we want to blend an extracted a ROI onto the destination map. We can see the source image as a black and white mask, where the ROI region is white and the reminder of the image is black. The algorithm is summarized below. For each pixel under the mask, its value satisfies the equation:

$$\text{for all } p \in \Omega \quad |N_p| f_p - \sum_{q \epsilon \Omega \cap N_p} f_q = \sum_{q \epsilon \partial\Omega \cap N_p} f_q^* + \sum_{q \epsilon N_p} v_{pq} \tag{20}$$

$$\text{For all } (p, q), v_{pq} = g_p - g_q \tag{21}$$

f* = source image, g = target image, p = pixel, q=neighbor pixel, Ω = pixels under the mask, $\partial\Omega$ = pixels who have at least one neighbor under the mask, N_p = neighborhood of pixel p, f_p = new value of pixel p, f^*_p = value of pixel p in the source image, v_{pq} =gradient of the pixel to a neighbor in source image g.

Each pixel in the target is the sum of the gradient of a pixel and its neighbors in the source image(v_{pq}), and the average of its neighbors $\sum_{q \epsilon \Omega \cap N_p} f_q$. Considering the fact that these neighbors can be unknown themselves, to find the results we need to find pixel values simultaneously. This can be done by using the matrix equation $AX = B$. The vector B is the guiding gradient plus the sum of all non-masked neighbor pixels in the destination image. The matrix A is a sparse matrix and each row of it holds an equation for a pixel in the output image to calculate its average values from its neighbors. Using this Poisson image blending technique requires a significant amount of calculation time especially for large images. To solve the problem we decided to use our own averaging algorithm for images with a large size. In the averaging algorithm, based on the size of the image, pixels in the result will be the average of the different percentage of each pixel's value in the three color channels from the ROI and the background image.

The averaging algorithm operates within a strip of pixels on the ROI's borders; the width of this strip is calculated with regards to the size of the ROI; see the equation (22)

$$\mathrm{p} = \frac{(0.05 \times \sqrt{\mathrm{S}})}{\mathrm{z}} \tag{22}$$

S = area of the image in pixels

Z = zoom level

We aim to create a visually smooth crossing between the ROI and the background image. The averaging algorithm calculates the average of pixels on the chosen strip in the three separate color channels (RGB) with a different percentage of each pixel's value from the ROI and the background image. This gradually uses a higher percentage of the background pixels' value by crossing from inside of the ROI to the outside. Pixels inside the ROI that are not on the strip are moved directly from the source image to the background image to replace the background image pixels. Figure 30 and Figure 31 shows a sample result of the image blending. The reader may wish to zoom into the image to see the results of the blending more clearly.

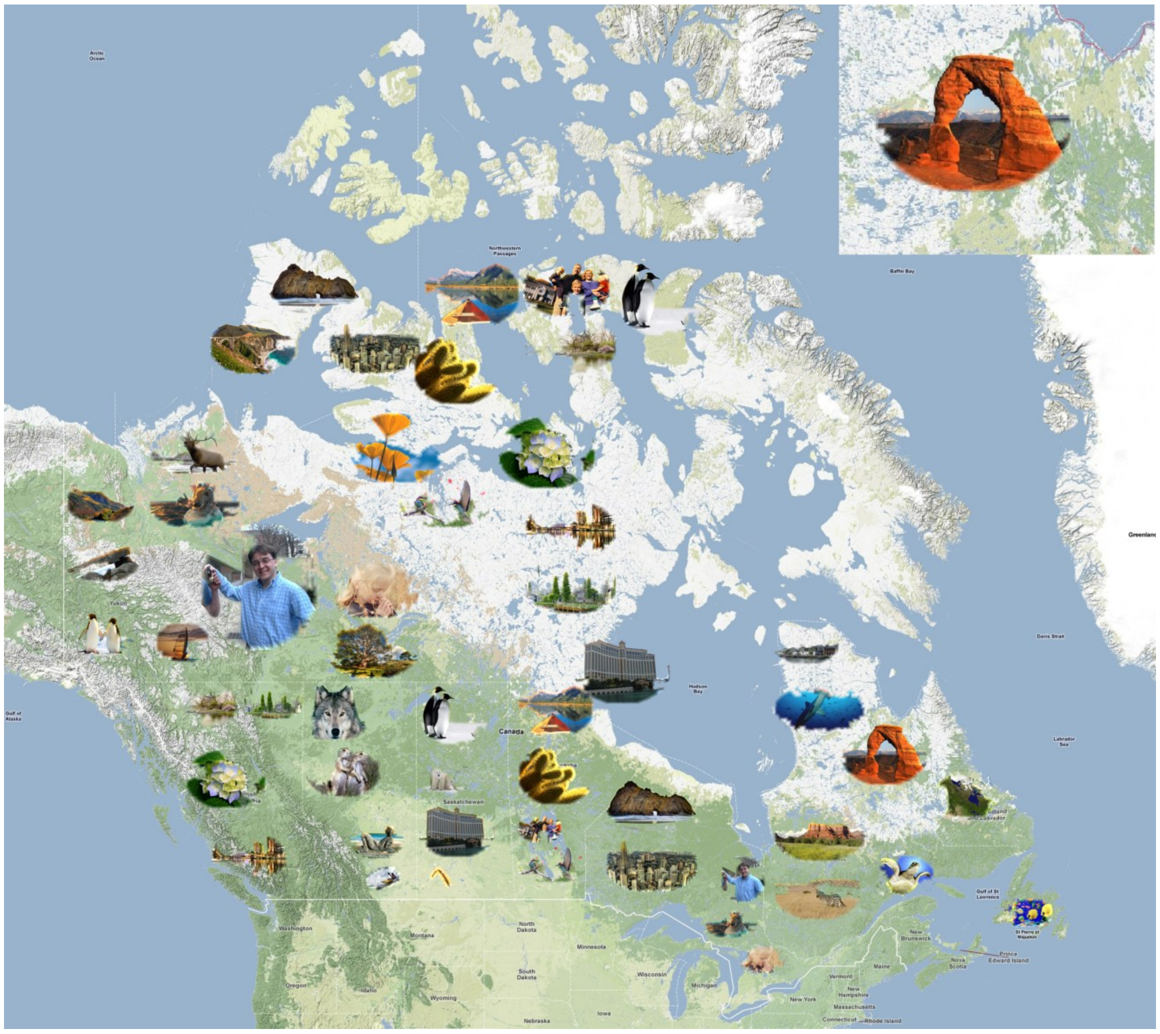


Figure 30: Sample result of the image processing algorithm; on the corner, zoomed view of one of blended ROI is showed.

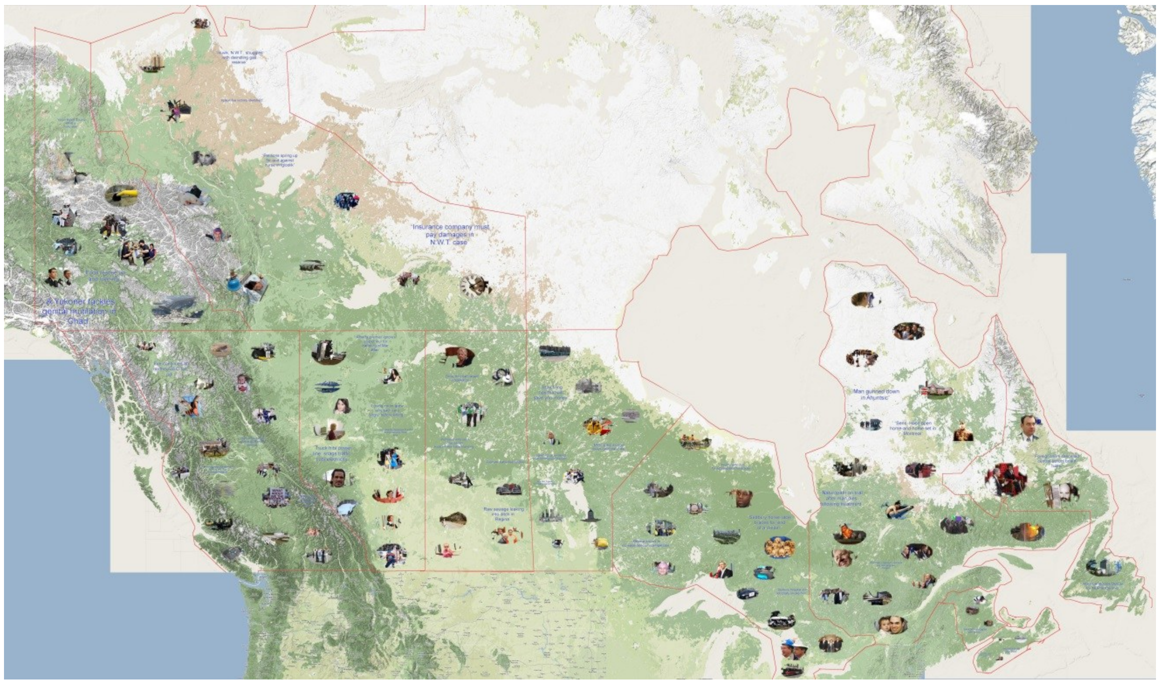


Figure 31: Sample result of the image processing algorithm. (Image is high resolution and zoomable)

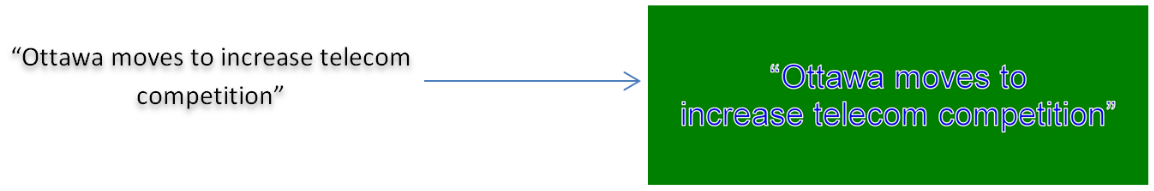

"Ottawa moves to increase telecom competition"
"Ottawa moves to increase telecom competition"

Figure 32: Covert text to image.

4.3 Packing

In the previous chapter, we talked about a method for placing tags on the map. After developing a successful method for putting ROIs on the map, we converted text tags to tag images. Therefore the algorithm can treat tags as images and use the algorithm to place the tag images on the map. This is accomplished by converting text keywords to text images and creating tag images. Figure 32 shows a sample tag image.

To place the ROIs and tag images on the map, our algorithm needs to have the masks of different locations on the map. For example, if we want to put ROIs on a city or a province; the algorithm needs to have the masks of the province or the city at different zoom levels. These masks help the algorithm to place ROIs inside the political borders and to prevent overlapping between different ROIs.

We are using Google Maps as a platform for our application. Google Maps uses 256*256 pixel size tiles in its maps and lets users upload their own tiles to replace the default Google Maps tiles. After rendering the ROIs into the map, we divide the results into 256*256 pixels size tiles to replace the Google Maps tiles. Our database provides geographic coordinates associated with news and our application uses these geographic coordinates to find the approximate location of the news on the map. Geographic coordinates contain longitude and latitude in world geodetic system[14] (WGS), however, the weight algorithm needs the pixel information of the location for the image processing stage. Below, we explain more about the geographic coordinate systems and the method that the application uses to convert geographic coordinates to pixel information and vice versa.

Google Maps uses four coordinate systems:

[14] http://en.wikipedia.org/wiki/World_Geodetic_System

- World Geodetic System: WGS is a standard for use in cartography, geodesy, and navigation. It includes a standard coordinate frame for the Earth, a standard spheroidal reference surface for raw altitude data, and a gravitational equipotential surface that defines the nominal sea level.[15]

- World Coordinate: After translating latitude and longitude values into a world coordinate, the coordinates can show the location on a map (screen). Google Maps use the Mercator projection for this translation. The Mercator projection is a widely used cylindrical map projection first suggested by the Flemish people in 1569. [16] Equations 16 & 17 show the formulas that our application uses to convert latitude and longitude to world coordinates:

$$\mathrm{WorldCoordinate}_x = \frac{\mathrm{TileSize}}{2} + \mathrm{longitude} \times \frac{\mathrm{TileSize}}{360} \tag{23}$$

$$\mathrm{WorldCoordinate}_y = \frac{\mathrm{TileSize}}{2} + \frac{1}{2}\log\left(\frac{1+\sin(\mathrm{Latitude})}{1-\sin(\mathrm{Latitude})}\right) \times \frac{\mathrm{TileSize}}{2\pi} \tag{24}$$

- Pixel coordinate: World coordinates show a unique location on a given projection, but for image processing, we need to translate world coordinates to pixel coordinates. This is accomplished by the following formula:

$$\mathrm{PixelCoordinate}_{x,y} = \mathrm{WorldCoordinate}_{x,y} \times 2^{\mathrm{zoomLevel}} \tag{25}$$

[15] http://en.wikipedia.org/wiki/World_Geodetic_System

[16] http://en.wikipedia.org/wiki/Mercator_projection

- Tile coordinate: It is impossible to load all map imagery in different zoom levels; therefore, Google Maps breaks up imagery at each zoom level into a set of map tiles with a size of 256*256 pixels. These tiles are logically arranged and they have unique coordinates in each zoom level. Figure 33 shows a sample tile division in zoom level 4. The arrangement of these tiles and their constant size makes it easy to find which tile contains the imagery for any given point. Equation 19 shows the formula:

$$\text{TileCoordinate}_{x,y} = \left\lfloor \frac{\text{PixelCoordinate}_{x,y}}{256} \right\rfloor \qquad (26)$$

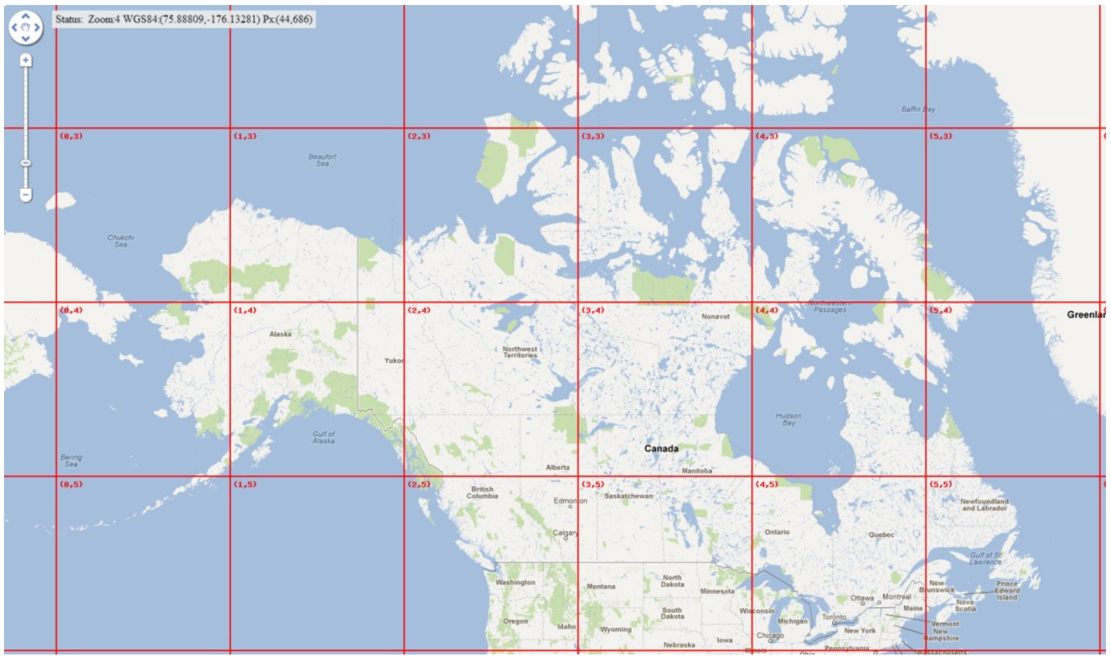


Figure 33: tiles in zoom level four.

Figure 34 shows the flow chart of the location algorithm.

1. In the first stage, the application reads the mask for target location. This target location can be a country, province, or city.

2. In the next step, the application connects to the database to read the news and its associated image. The application applies the ROI extraction algorithm on the image to find and extract the ROI.

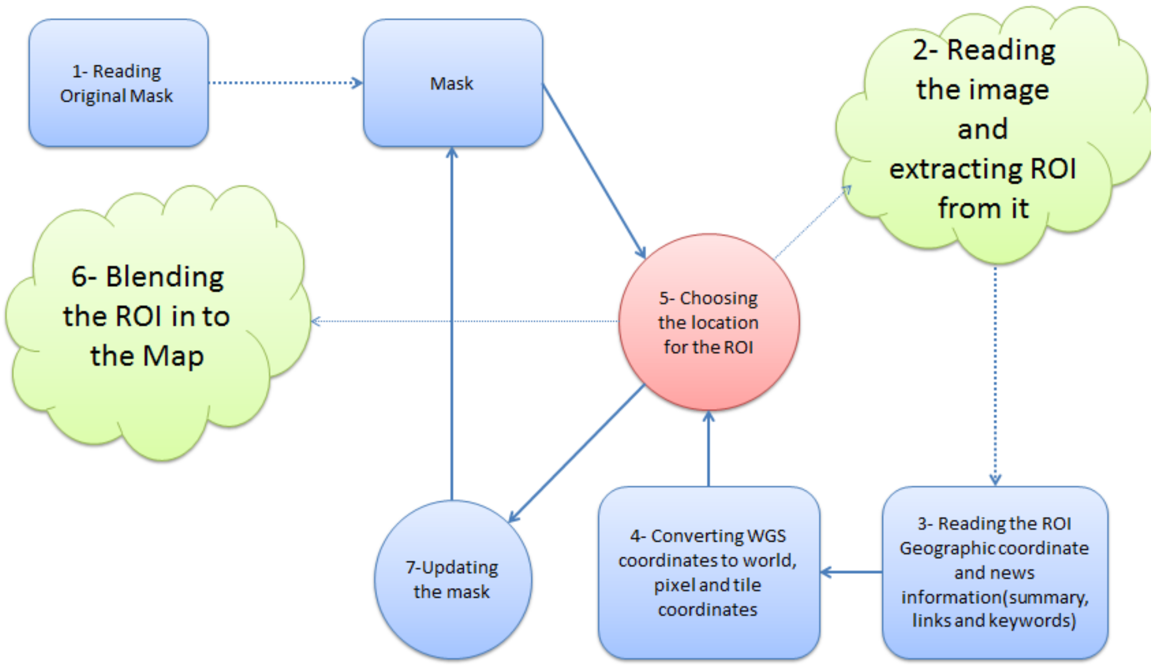
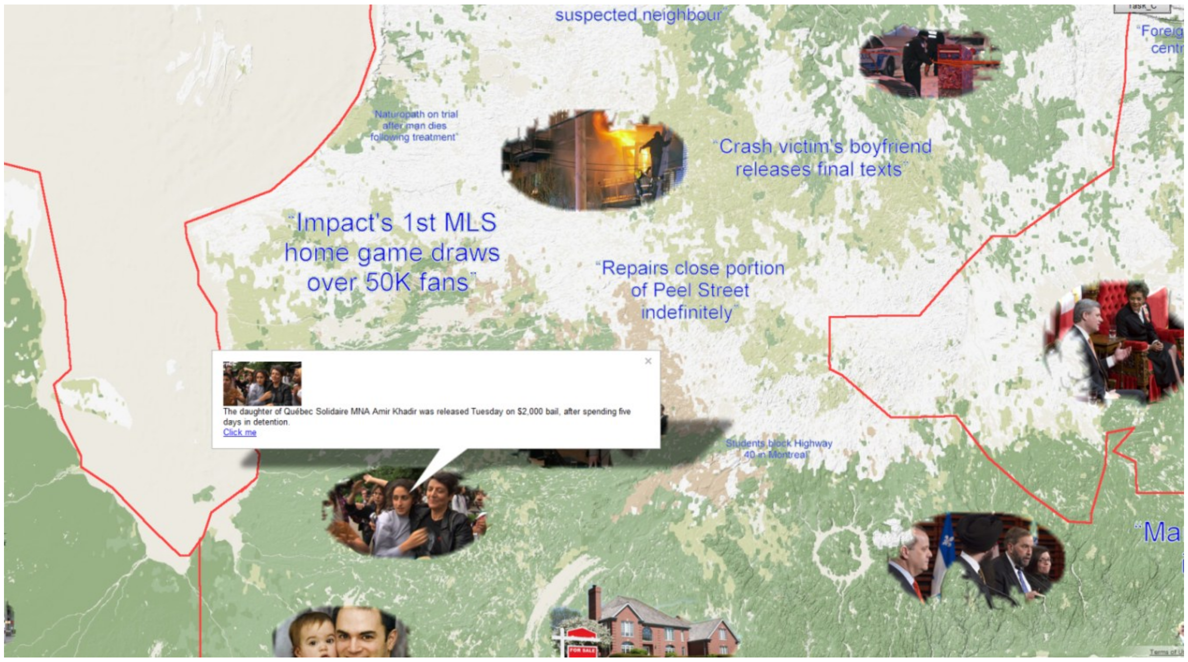

1- Reading Original Mask
Mask
2- Reading the image and extracting ROI from it
6- Blending the ROI in to the Map
5- Choosing the location for the ROI
7-Updating the mask
4- Converting WGS coordinates to world, pixel and tile coordinates
3- Reading the ROI Geographic coordinate and news information(summary, links and keywords)

Figure 34: Flow chart of location algorithm.

3. After extracting the ROI from an image, the application connects to the database to read the ROI geographic coordinates and the news information (summary, links and keywords). Key words and links are placed in the center of ROI to show in a Google Maps balloon. When users click on the center of the ROIs or tags on the map a Google map balloon pops of to show the keywords (header) of the news, a summary of the story and the link of the news source. In addition, users have this chance to click on the link to go to the source of the news and see more details. Figure 35 shows a sample of a Google balloon in the application.

4. After reading the geographic coordinates, the algorithm calculates these coordinates to the other coordinate systems as follows:

    a. World coordinate: to calculate the pixel coordinates, the algorithm needs to have the world coordinates.


suspected neighbour
"Naturopath on trial after man dies following treatment"
"Crash victim's boyfriend releases final texts"
"Impact's 1st MLS home game draws over 50K fans"
"Repairs close portion of Peel Street indefinitely"
Students block Highway 40 in Montreal
"Ma

The daughter of Québec Solidaire MNA Amir Khadir was released Tuesday on $2,000 bail, after spending five days in detention.
Click me

Figure 35: Sample Google balloon for representing news summary and link.

    b. Pixel coordinates: by calculating the pixel coordinates, the algorithm will know the exact location of the event on the map imagery. The application uses this location to find an available free space close to this point for placing the ROI or tag image there.

    c. Tile coordinate: after saving the result in 256*256 pixel tiles, tile coordinates are used to replace the Google Maps default tiles.

5. In the next step, the algorithm uses Bourke's equation to calculate the available free space in the mask. The algorithm applys a new size on the ROI and Tag images based on the available free space and number of tag images and ROIs. Equation (13) shows the formula that the application uses to calculate the size:

$$\mathrm{A_i} = \frac{4}{5}\mathrm{C_i}\left(\frac{\mathrm{A_T}}{\sum_{\mathrm{i}=1}^{\mathrm{N}}\mathrm{C_i}}\right) \tag{27}$$

Where A_T shows the area of the region, A_i is the candidate area for the tag and ROI, C_i is a constant number that shows the level of importance for each tag (weight) or ROI.

We assume approximately 20% of the total area will not be usable because of the shape of the region and the gaps between tag images and ROIs. If the algorithm cannot put the image on the map, due to lack of available free space, overlapping with previous rendered images or cutting political borders; it reduces the size of the image and tries to place the image in another free location near the place of event. Figure 36 shows a sample mask after putting ROIs and tags on the map.

6. After finding the appropriate location, the ROI or tag image will be rendered in to the map by using Poisson image editing and the average algorithm, create a smooth and attractive transaction between the image and the background.

7. Finally, the mask will be updated to use to place the next image. Figure 36 (right) shows the mask after four updating.

Our system is able to provide an effective and visual attractive collage layer in a short time. We used a combination of existing image processing techniques in our system, such as active contour, Fourier transforming filters, morphological reconstruction, and Poisson image editing. In addition to implementing existing image processing techniques, we developed our technique for blending ROIs into the host map. We developed a novel method for extracting ROI an image. Figure 37 shows a sample result of the algorithm.

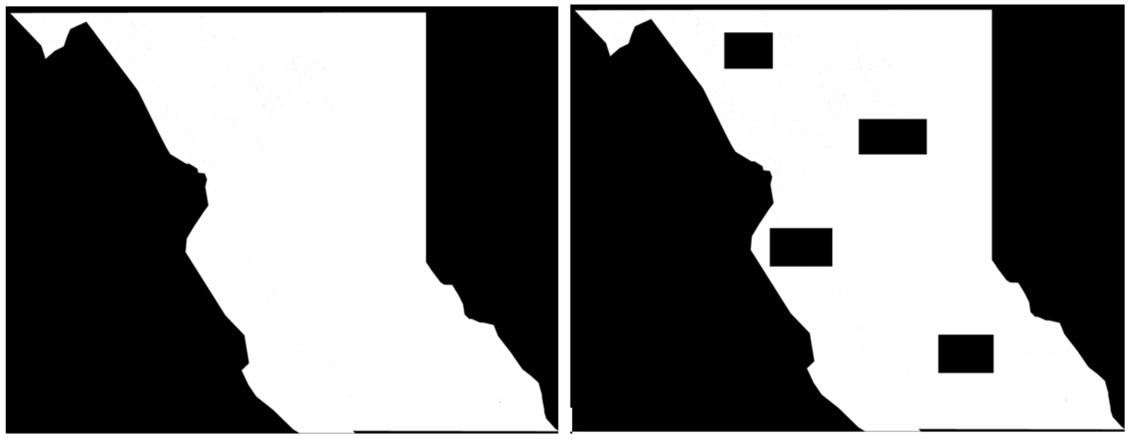


Figure 36: (left) original mask for province of BC. (Right) updated mask after adding four ROIs.

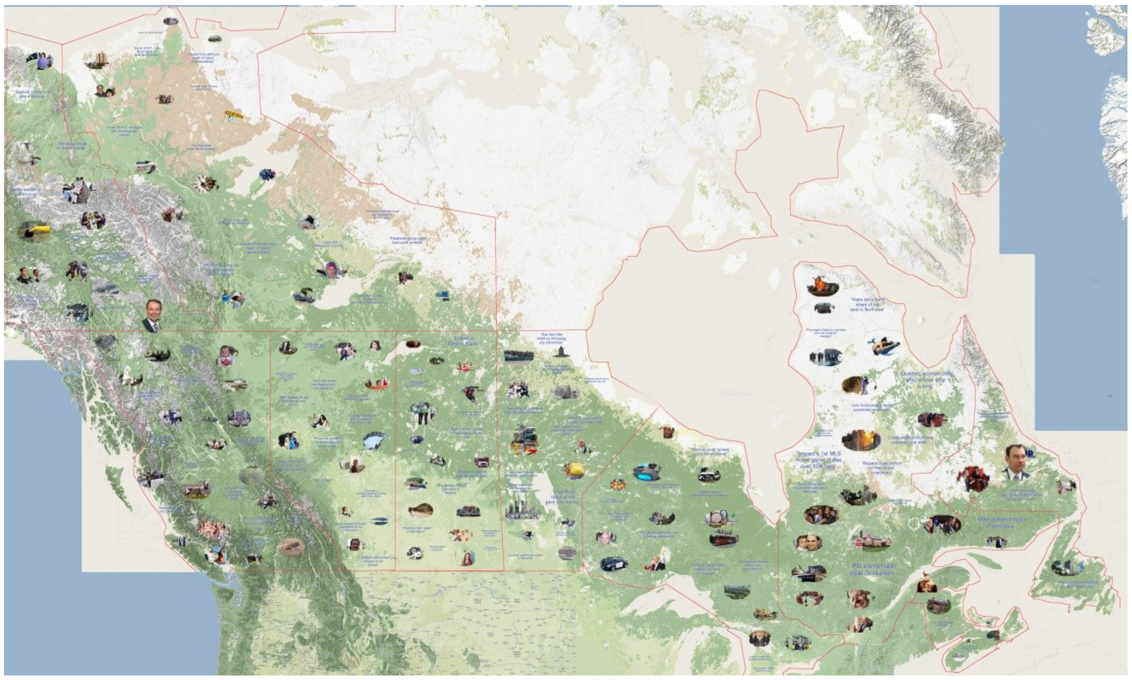


Figure 37: Sample result of the image processing algorithm. (Image is high resolution and zoomable).

CHAPTER 5

USER STUDIES

We evaluated the usability of our application in two studies. We wanted to identify which sizes, numbers, and orientations of images and tags are more attractive and useful for users, while still operating within the constraints of application performance. We also wanted to identify how combinations of tags and images can help users to find out about different news stories in different geographic areas. Furthermore we wanted to understand if users understand the relationship between zoom level on the map and the regional relevance (provincial, national) of news items displayed at that zoom level.

5.1 Research questions

In addition to the research objective of obtaining general usability feedback about our application design, three primary research questions are addressed in this study:

1. Find out about user preferences and the effectiveness of different information presentations:
    a. Size of images and text tags on the map
    b. Ratio of images to text tags.
2. Determine how does an interactive map with zoom and pan helps users find out more about news stories in different regions.
3. Understand general interaction patterns when using the application

a. How do users scan news on the map?

b. Is the application used mainly as a tool to gain a general overview of news without checking the source of news, or do they use the application to find out about details of particular news stories by going to the related news website through provided links?

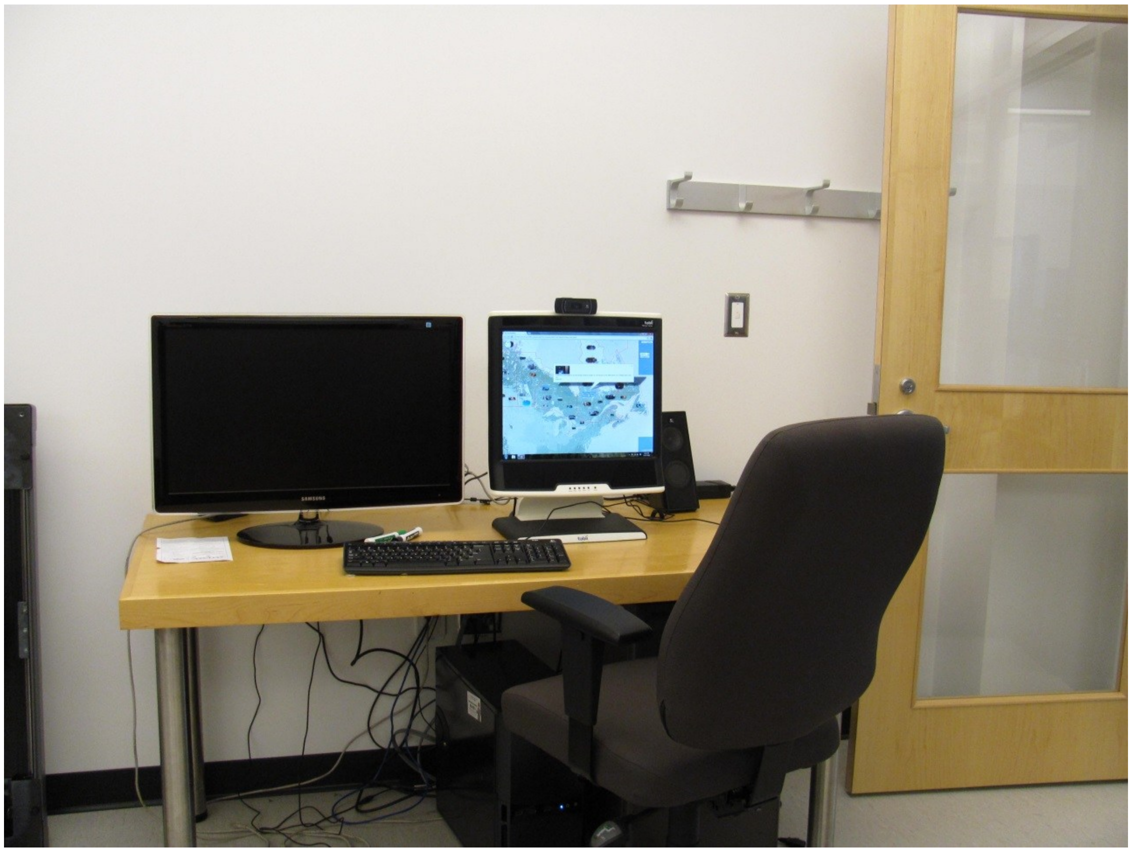


Figure 38: study setting and system configuration with eye tracker. (Ethic # 2012-2625, R# 1010375)

5.2 General study design

We conducted a laboratory study using members of the Dalhousie University community as participants. The ethics approval number for the thesis is project# 2012-2625, R# 1010375. We used a desktop computer with a dedicated eye tracker system to perform tasks. Figure 38 shows the eye tracking system and the lab. We timed each task, logged interactions with the interface via software logging, recorded gaze using the eye tracking system, and employed a "think aloud" protocol to elicit reasoning and impressions from users while they were completing the tasks. An experimental observer took notes based in part on participant comments. After performing a set of tasks, participants filled in a questionnaire asking them to rate the different methods of displaying and interacting with the application. Participants also filled in a background questionnaire that asked about their experience with similar applications, and a post-condition questionnaire to gather their opinions and feedback about the system in general and on all the tasks that they performed. At the end of the study, the experimental observer conducted a brief unstructured interview, asking participants to reflect on behaviors that were noted during the study. These interviews, and the completion of all tasks, were audiotaped. Each participant was compensated with $10 cash.

The Tasks

We aimed to identify proper sizes, numbers and orientations of images and tags to provide attractive and useful geotag layers for users. We also wanted to identify how combinations of tags and images can help users to find out about different news stories in different geographic areas. Using the eye tracking system allowed us to study participants' user attention when using the application. The different conditions for size, numbers and arrangement are listed in table 1. Table 2 shows the task for each condition.

Table 1: Information Types and Conditions for the first user study.

Condition	Description
A: High number of images and low number of text tags	In this condition, a high number of images is placed on the map with a small number of tags. This model represents more visual details than tag details. We used an 80% image to text ratio.
B: Medium number of images and medium number of text tags	In this condition, a medium number of images is placed on the map with a medium number of tags. This model represents a trade-off between images and tags. We used a 50% of text to image ratio.
C: Low number of images and high number of text tags	In this condition, a low number of images are placed on the map with a larger number of tags. This model represents more tags than images. We used an 80% of text to image ratio.

Table 2. Information Types and Conditions for the second user study

Condition	Description
A: High number of images and low number of text tags	In this condition, a high number of images is placed on the map with a small number of tags. This model represents more visual details than tag details. We used an 80% image to text ratio.
B: Medium number of images and medium number of text tags	In this condition, a medium number of images is placed on the map with a medium number of tags. This model represents a trade-off between images and tags. We used a 60% of text to image ratio.
C: Low number of images and high number of text tags	In this condition, a low number of images are placed on the map with a larger number of tags. This model represents more tags than images. We used a 75% of text to image ratio.

Table 3. Tasks for the both user studies

Tasks	Description
Zoom and pan on the map to find out about different stories	Participants zoom on the map and report their impressions about the news in a specific area. We ask: "using the map panning and zooming controls, please have a general review of the news in province X"
Zoom and pan on the map to find the top three stories	Participants zoom on the map and report their impressions of the top news stories in a specific area. We ask: "using the map panning and zooming controls, review news in province X, and after reviewing the news at different zoom levels, please select three news items that you believe are important."
Using interactive map to	We tell participants to find specific news items in an area. We direct the participant as follows: "for this part of the study, I will explain a piece of news, and I will tell you this news belongs to a specific part of the country. After I finish my explanation, please try to find the news on the

find specific news	map. Do not forget, this news item may be found at a different zoom level, it can be represented by an image, text or both image and text, and sometimes there will be more than one news header about the subject. Example: "More than 1,000 obese Manitobans are waiting to see if the provincial government will approve a weight-loss surgery program that could save their lives." We asked several of these types of question, some referring to text news items, and some to images.

Participants completed each type of task under each of the four conditions. Task types were completed in the same order across all conditions. The order of conditions was counterbalanced using the following 3x3 Latin square:

ABC; CAB; BCA

The Study protocol

1. Participants were recruited from email mailing lists.

2. Researcher arranged a time to meet with each interested participant.

3. At the start of the study, the study was explained; participants gave informed consent, and filled in the background questionnaire (Appendix A).

4. Participants did each task for each condition, and fill in a post-condition questionnaire (Appendix B) after finishing the tasks for that condition.

5. After finishing all conditions, the participant filled in the post-experiment questionnaire (Appendix C).

6. Participants were compensated and filled in the payment form.

User studies procedure

In an initial user study, we recruited 24 participants (12 Canadian and 12 international) who were all members of the Dalhousie University community. None of them had experience with Wordles before. We used the results of the first user study to make some improvements to the application. Our experience with the first user study showed that international participants would not be a good study due to lack of knowledge about Canadian geography and Canadian news. For example, we had to show the location of provinces to international students; and when we asked questions about political subjects, they didn't have a strong sense about the news. This lack of knowledge meant international participants would be more likely to draw solely on perceptual cues such as font and image size, central items, and background knowledge about different subjects when identifying most important news items; while Canadian participants might be more influenced by the content of news headlines in addition perceptual cues.

In the second user study, we recruited 12 Canadian.

Use of eye tracking system

There was a possibility that participants might become confused or distracted while tracking news on the map. We also had evidence of different scanning methods in our pilot tests (i.e. scanning tags or images predominantly). In addition, we wanted to study user attention during the use of the application. We capture users' visual attention using an eye tracking system. We believed studying user attention and users' operation during completing the tasks could help us to identify the elements that can affect users' attention and performance. For example through eye tracking system data, we were able to identify the elements that detracted users from the task, or we could identify the method that each user used for scanning news over the map. All this information gives us a stronger characterization of how the application is used and how we can provide a better arrangement of tags and images for the application.

The application was run on a core i7 CPU, 3.2 GHz machine with 16GB of RAM, with a Tobii T120 eye-tracker acting as the primary screen. The Tobii T120 eye-tracker is a non-invasive device that a 17" display with dedicated eye tracking sensors with the capability of collecting data at a rate of 60 Hz (or 120 Hz). The eye-tracking data software collects all keystrokes, mouse clicks made and records the audio.

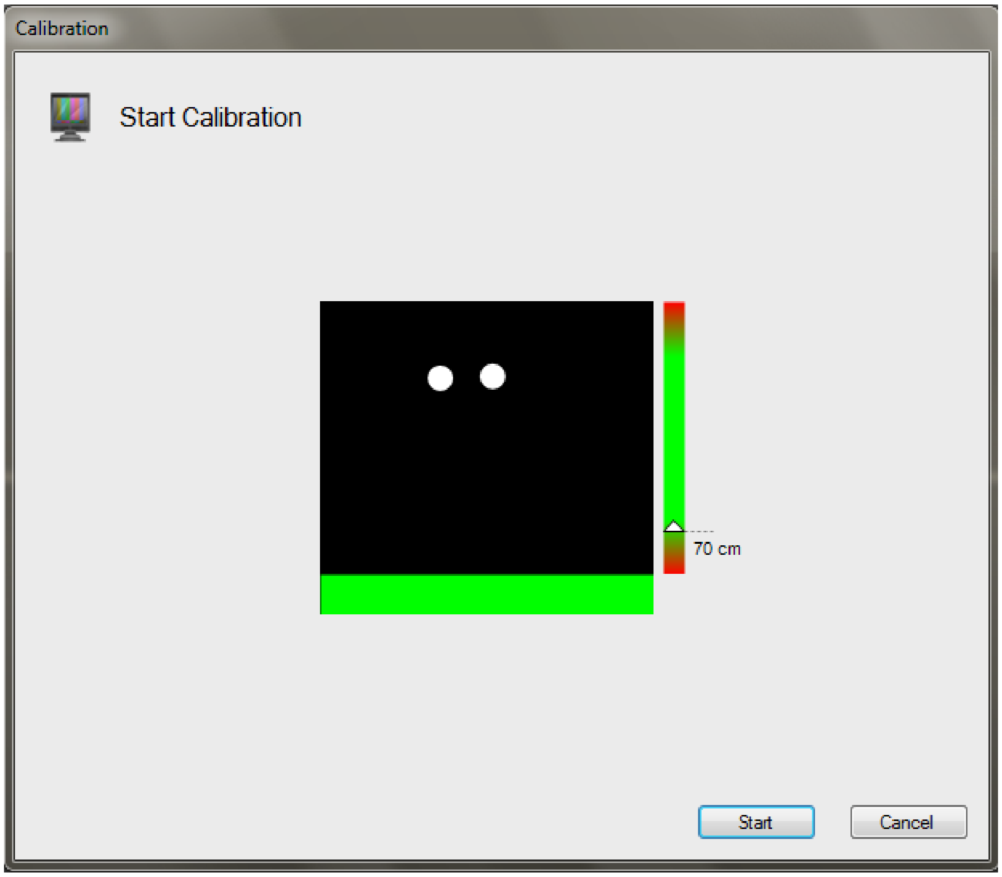


Figure 39: Calibration display.

The eye tracker has a pre-calibration display (Figure 39); users can see their eyes being captured in this display. There is a bar at the bottom of the display, so that if the eye tracker captures data without any difficulties, the bar will be green, and if the eye tracker has difficulties to capture data, this bar becomes yellow, and if it does not capture data, its color is changed to red. The bar on the right side shows the appropriate distance for the best capture of data. The calibration display makes an automatic system calibration for each user to collect more accurate data. In addition, the calibration display helps users to become more familiar with the eye tracking system through seeing their eye captured with it. We gave the participants some time to "play" with this display, so they could become more aware of how their movement affected data capture. This short experience with the eye tracking system increased the accuracy of data by decreasing missing data due to excessive head movements.

5.3 Data Analysis

After the study, we analyzed questionnaire responses to gauge user preference and satisfaction for each condition. We used the questionnaires to measure user preference for different image sizes, numbers and arrangements of images on the interactive map. We used the logs (time, mouse interaction and eye tracking results) to assess the impact of image and tag arrangements and sizes on interaction patterns. Through eye tracking logs, think aloud, and observer notes, we kept track of any difficulties in doing any of the tasks to help determine the effectiveness of the options. Task responses, questionnaire responses and think aloud data have been used to build a sense of how well each condition arrangement supported users in correctly answering task questions.

To find an answer for our research questions, we designed questions one to six to evaluate the usability of the application, and questions six to eleven evaluate the visual properties of the application.

Figure 40 shows the results of the first user study with 12 Canadian participants. Studying the chart suggests that participants found the “medium images-medium text” (Medium Image) condition more effective for scanning news and to find a particular news item in a region, than the “high images low text” (High Images) and the “low images-high text” (Low Image) condition. On the other hand, participants liked the visual properties of the High Images and Low Image conditions more than the Medium Image condition.

Figure 41 shows the results of the first user study for 12 international participants. It is hard to find a pattern for the results that are shown in Figure 41. In general, Figure 41 shows that international participants prefer to scan news more based on images. To find evidences for the suggested idea, we reviewed eye tracker data for each 12 International participants. In addition, we used notes that had been written by researcher, while users were completing the task. Reviewing eye tracking data and scripts, suggest the same result that we can learn from the chart in Figure 41.

Table 4 List of questions

1	**It was easy to use the application**
2	It was easy to find out about general news headers at different zoom levels on the map.
3	It was challenging to find out about news headers in a particular region.
4	I sometimes lost track of where I was on the map.
5	It was easy to locate news by zooming and panning the map
6	The number of images on the map was appropriate for scanning news.
7	The size of images on the map makes it hard to scan for news.
8	The number of the text tags on the map was appropriate for scanning news.

9	The size of the text tags on the map was appropriate for scanning news.
10	The arrangement of images and text tags made it hard to find out about news
11	The arrangement of images and text tags was visually attractive

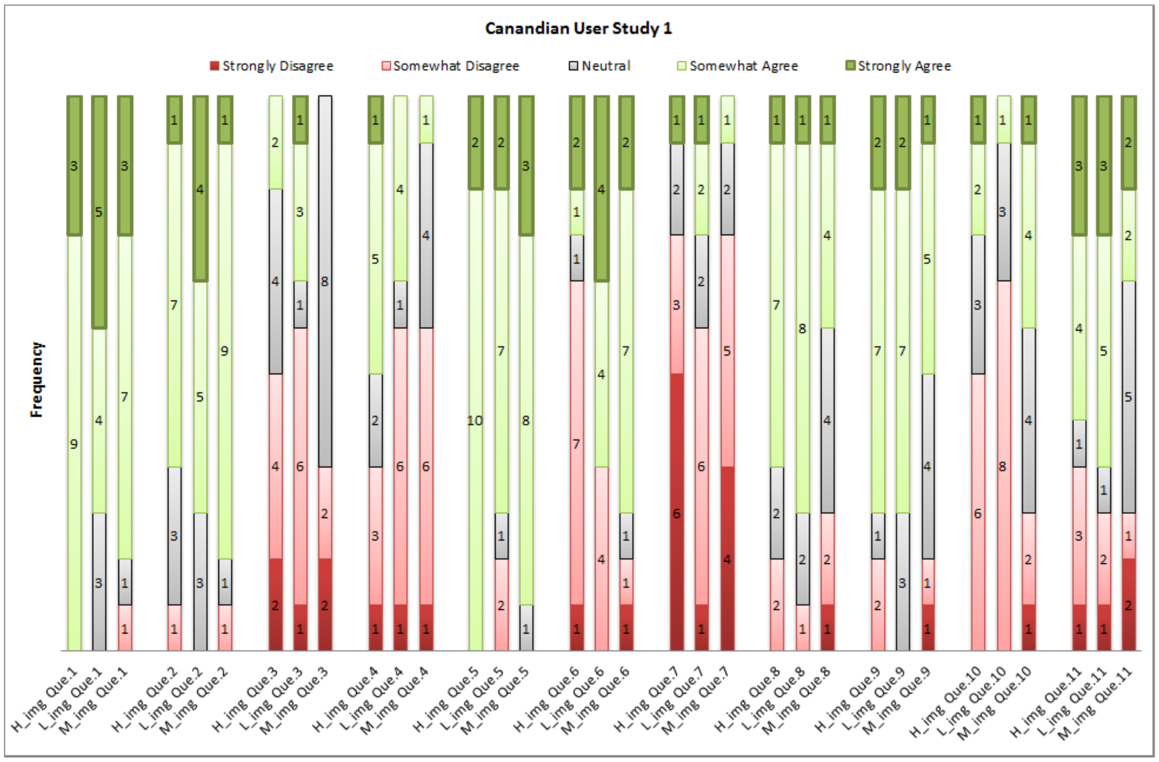


Figure 40 Results of the first user study for 12 Canadian participants (4 shows list of the questions).

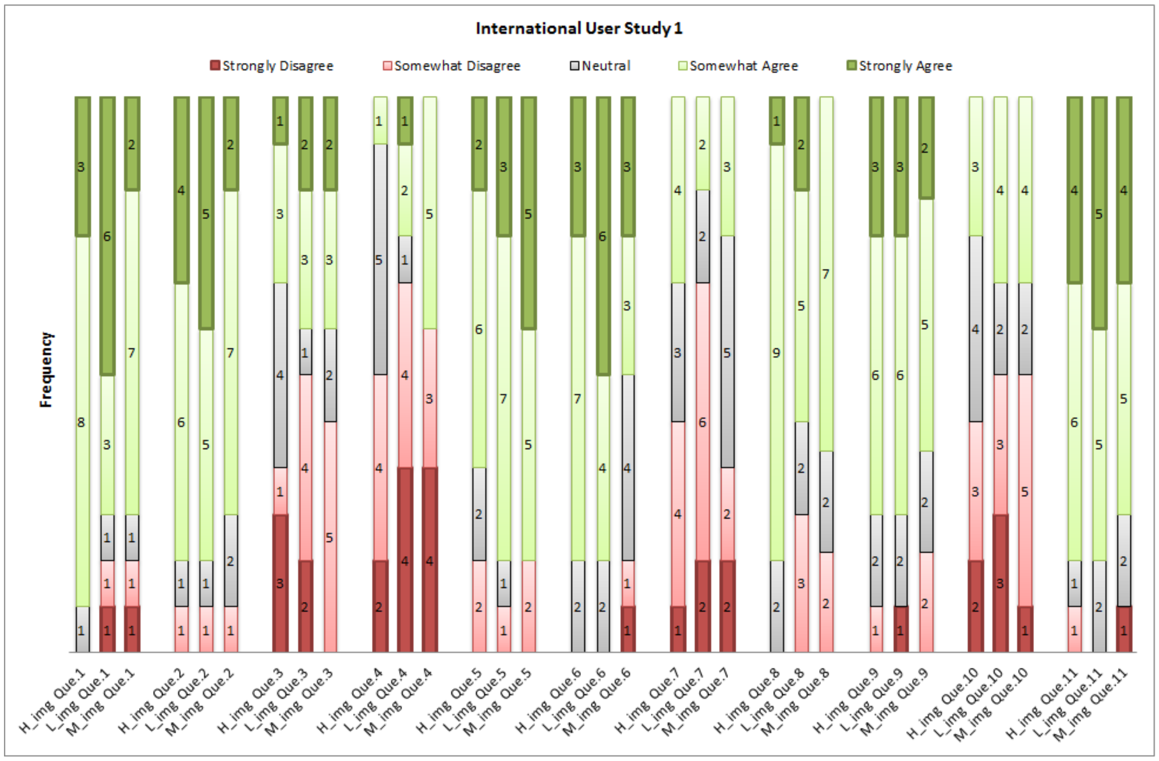


Figure 41 Results of the first user study for 12 International participants (4 shows list of the questions).

Eye tracking records show that most international participants started to scan news items from images and if they did not find the desired results, they started to review text. However, we could find a different pattern for Canadian participants. We cannot say that most of Canadian participants start scaning from text. It was most likely a balance in the methods that Canadian participants used for the scanning news, however it was obvious that independent from their method of scanning , Canadian participants spent much more time on reading text. We believe international participants were more interested in images for the following reasons:

- Having English as a second language made international participants more interested in images than text.
- Unfamiliarity with Canadian events, politics and culture made international participants less interested in news and reading text.
- Unfamiliarity with Canadian geography made participants scan news without regard for regions.

In addition, we made some improvements in our application based on feedback from the first user study. Figure 42 shows the application before improvements and Figure 43 shows the after improvements. These improvements were:

- Solving the overlapping problem between text and images.
- Adding news summaries and source links to each news item (for both images and text tags).
- Use more pronounced differences in the layers for each condition (i.e., have more extreme ratios of images to text in the Low Image and High Images conditions).

Our experience with international participants in the first user study encouraged us to concentrate only on Canadian participants for the second user study.

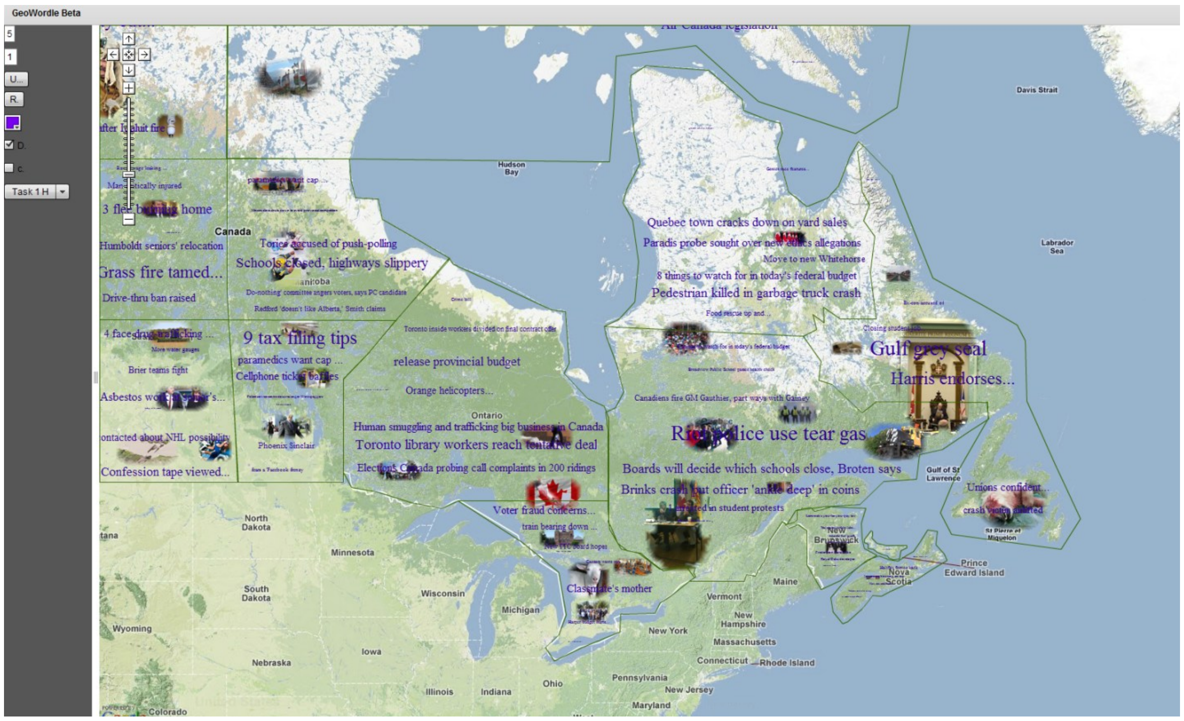
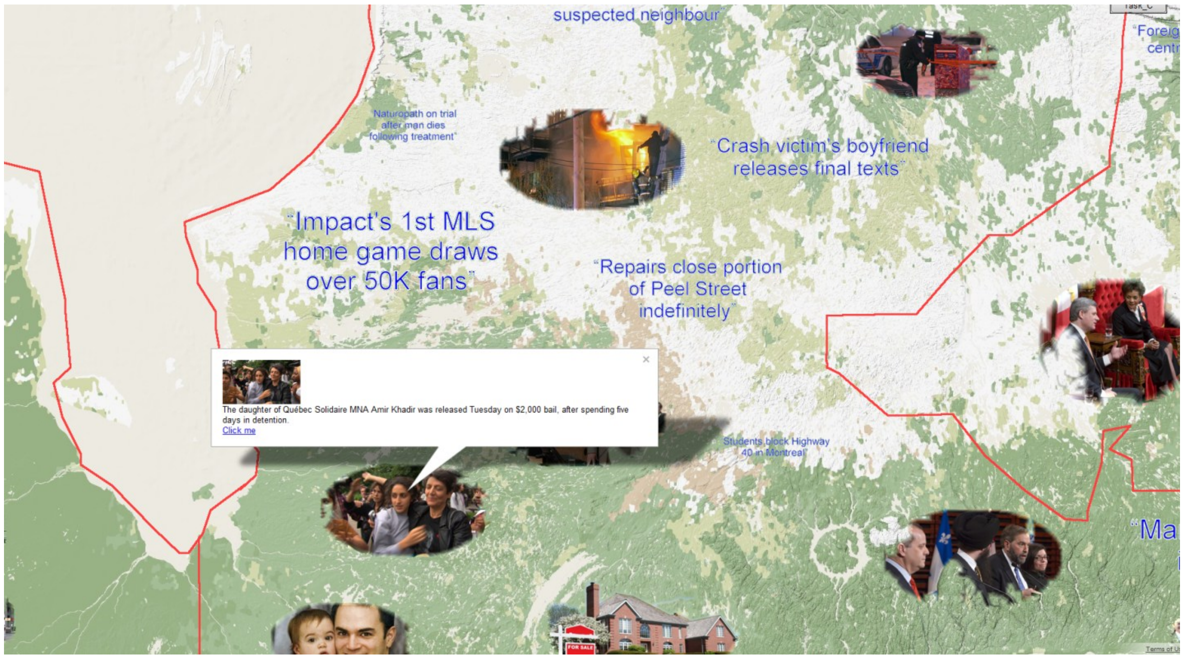


Figure 42: application before improvements.


suspected neighbour
Naturopath on trial after man dies following treatment
"Crash victim's boyfriend releases final texts"
"Impact's 1st MLS home game draws over 50K fans"
"Repairs close portion of Peel Street indefinitely"
Students block Highway 40 in Montreal
Click me

The daughter of Québec Solidaire MNA Amir Khadir was released Tuesday on $2,000 bail, after spending five days in detention.

Figure 43: Application after improvements

Figure 44 shows the results of second user study with 11 Canadian participants. Studying the chart for the second user study shows:

For the questions 4, One participant indicated that he lost the track of where he was on the map, and for question 8, one participant indicated that the number of text tags on the map was not appropriate for scanning the news. For question 7, two participants thought that the size of images on the map made it hard to scan for new. In other questions we did not have any other negative feedback for the Medium layer. For High image layer and Low image layer, we usually have negative feedbacks.

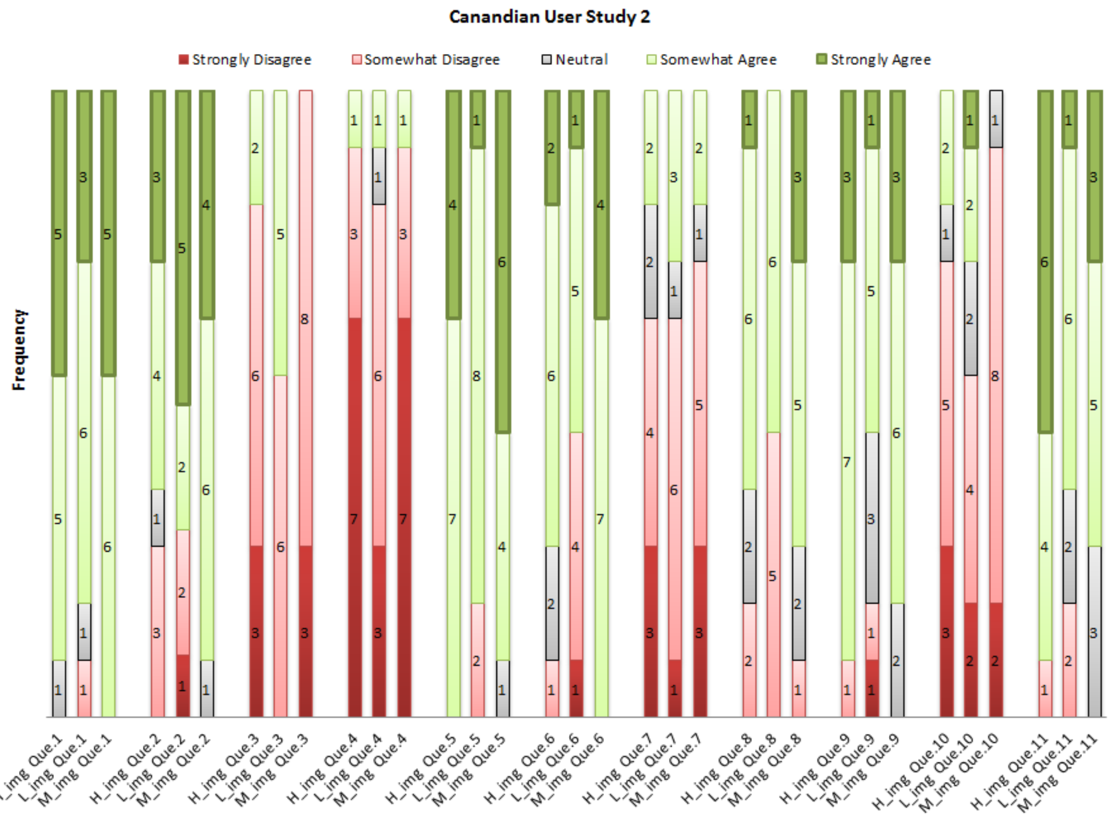


Figure 44 Results of the second user study for 11 Canadian participants (4 shows list of the questions).

Reviewing notes, audio and eye tracking data emphasizes the results that are shown in Figure 44. In the second user study, after completing the tasks on Medium Image condition and starting the tasks on Low Image conditions, P2 said: "*I think I liked it more with more images. But I guess you can communicate more with this. So it's not intuitive to me why some text is small and some text is bigger. I'm wondering whether that means it's local news, or whether it's more important*". From this comment, we noticed that difference in size of elements on the map could be understood in different ways. P2 mentioned that high numbers of images on the layer made him confused, and sometimes he attributed meanings to the images that were at odds with the actual linked news item. For example, we had an image of a moose in the province of Saskatchewan. Participants usually thought this image should be related to nature news, for example the population of moose; but, when they clicked on the image, the news summary showed that news is about traffic and related problems that were caused by different animals. In the second user study,P3 mentioned: "*without the text I'm kinda lost.*" On the other hand, five participants did not like the Low Image layers. They explained that text on the map was boring for them and when they wanted to find a news item, it became harder to find the news among many other text items.

Reviewing the eye tracking data gave us interesting information about different semantic and visual methods that users used for scanning news. In the visual scanning methods (how participants first scan a region in order to identify potential items of interest), we noticed 4/11 of participants started to review news by scanning text first and then reviewing images; 4/11 of participants started to review news by scanning images first and then reviewing text, and 3/11 of participants shows different behaviors in facing with different layers. Eye tracker data confirms the fact that news layers can affect user scanning behavior; during reviewing the eye tracker data of Canadian participants for the first and the second task, we identified four users who changed their method of scanning by changing the news layer, for example for the High Images layer they started scanning with images but for the Low Image layer, they started scanning with text. It was interesting to see that all five participants who first scanned text to review news liked the Low Image and Medium Image layers and all four users who first scanned images liked the High Images and Medium Image layers. This means the Medium Image layer can satisfy most of participants to have their favorite scanning method.

In semantic scanning methods (logical methods of reviewing news), we noticed that five of our participants were scanning news on a high zoom level to get an overall picture of the news, they read headers and sometimes the summaries but three of them zoomed in for more news or clicked on links to go to website to read more details. Two of these participants explained that, in general they would like to have important news on top of the map and to use the application just for a general review of the news. And they do not like to zoom in or out. Participant number one "*I like to zoom in to a certain level and stay at that level for a long time. I don't like zooming in and out a lot. Which can sometimes mean I am missing a story just 'cause different stories appear at different levels.*" Six participants zoomed in to regions of interest to see more news, they usually read news summaries, and four of them clicked on the links to go to the news website for more details. Seven participants explained that they liked the application when it shows more news by zooming into a region. As P3 said: “*So I like how that's local, how, if you zoom it, you get more pictures and it seems to be more local”.*

Our tasks specifically asked participants to find news in a particular region. We believed if we asked questions about more prominent and well-known regions, participants would use their background knowledge and this could affect the accuracy of our evaluation, because users could use their background knowledge instead of information that was provided with the system; Two of our questions asked about news in Toronto and Vancouver. We chose these questions to have a chance of studying behavior of participants when searching for news in familiar geographic regions (having background knowledge). Eight participants directly went to these cities to find the news. For example, P2 in answer of our question, “*Two members of Toronto’s newly constituted transit board said Tuesday they hope Monday’s move to dissolve the former board will clear the way for progress on transit expansion.”* explained: *“it looks like it should be in Toronto, and it's about busses. So I'll zoom out, and zoom into Toronto, at least where I think Toronto is*”.

CHAPTER 6

CONCLUSION

The Internet is a technology that has changed people's lives significantly, and it has become an inseparable part of daily life. Online services, such as online news websites, media sharing services, social networks are some of popular services can be found on the Internet. With the rapid growth of Internet and its services, people are facing a huge flow of information in different area such as news, images, science and it becomes a challenge for them to browse through thousands of websites, images and news stories returned from a query on the Internet. Systems have been developed that provide summaries from online news websites in order to provide a visualization tool to rapidly scan news stories. In this thesis, we introduced a novel visualization system that uses geographical location combined with image collages and tag clouds.

Our unique Geo-Collage application amplifies the user's ability for fast reviewing of news in different geographical regions and in a variety of news categories. This application introduced from a number of new techniques, including a novel fast algorithm for extracting ROIs and a new way of visualizing images on an interactive map. We also used the results of the user studies to improve the application and make it more useful to average users.

We split the project into three main sections: creating tag cloud layers, image processing to create geo-collage layers, and user studies to evaluate the work.

In the first part, we reviewed existing methods that create tag cloud and tag map layers. We listed the strengths and weaknesses of existed methods to define a prototype for our tag layer development. The most important issues with tag layer design were:

- Analysis of the geographical region to have a mechanism for the association of the tags with the correct locations.
- Having visually attractive tags with flexibility in size, color, orientation, transparency and appropriate tag density.

We reviewed each issue and possible solution in detail and then discussed our dividing algorithm. This is a hybrid approach that combines Bourke's method[17] for calculating geographic areas and aspects of Seifert's [22] proposed method for handling visual properties.

We also began with the assumption that images on a map can provide a quick way to transfer information which is visually attractive and engaging for users. We therefore focused on image processing in the second phase of the project to develop geo-collage maps, which combine image cuttings and tags on geographic maps. We discussed this part of the work in three stages:

- Determining ROIs and generating tag images.

[17] http://paulbourke.net/geometry/polyarea/

- Finding a good location on the host map for the extracted ROIs and tag images
- Rendering the ROIs and tag images into the host map.

We talked about active contours, visual attention methods and object detection algorithms in detail. But none of these options provided what we required for our application. For example active contours [38][47] took excessive computation times and Viola-Jones [42] failed in recognizing all faces. We developed our own method for extracting ROI from an image, by using Fourier transform function, canny edge detection and morphological reconstruction.

We note that currently our approach works best for medium to high quality images. A limitation of the algorithm is that it is not generally suitable for low quality images (blurry, noisy, etc.). However, if the result of the algorithm on low quality images is unsuccessful, the application defaults to predefined masks which assume that a broad center region of the image is the ROI.

We used geographic coordinates of news for placing ROIs and tag images on the host map. We discussed different geographic systems that have been used by Google Maps and different equations for converting these coordinates systems to each other. We used geographic coordinates and binary mask of each province to pack ROIs and tag images inside the political borders of each province without any overlapping between ROIs and tag images.

To create a seamless, clean transition between ROIs and the background map we used a Poisson image blending algorithm [6] and our own averaging algorithm. Using the

Poisson technique on large images takes a prohibitive amount of processing time, and on small images, it has unacceptable results. Therefore, we developed the average technique (section 4.2.2) to use on small and large images.

After developing our application, we evaluated its usability in two studies. We were interested to find out how users interpret text and image size, and placement as indications of a news item's prominence. We also wanted to determine if they understood the semantic relationship between zoom level on the map and the regional relevance of news items displayed at that zoom level. In addition, we used an eye tracking system to study user behaviors and have data that are more accurate.

We used results of the first user study to make some improvements to the application to make it more usable for users. From results of the first user study, we understood that our news application is more practical for people who are familiar with Canadian news topics. A lack of geographic knowledge of the country in which news stories occur confounds the utility of the system. Therefore, in the second user study, we only used Canadian participants.

In the second study, three primary research goals were addressed:

1. Find out about users' preference and the effectiveness of different size and ratio numbers of images and text to map size.

2. Determine how does an interactive map with zoom and pan help users find out more about news stories in different regions.

3. Find out about user interaction with the application. For example, how do users scan news on the map? How do users use the application?

From analyzing collected data (questionnaires, eye tracking data, users recorded audio), we identified different methods of scanning that have been used by participants, for example some participants preferred to scan text first and then scan the images, and vice versa. Some participants liked to use the application for understanding general news and some of them liked to use it for knowing more details. We understood that our application is more usable for participants who are familiar with Canadian news and geography.

6.1 Future work

Although our application has already enjoyed positive feedback, some improvements remain to be developed, some of which were identified in the user study. Below we review some possible subjects for further research.

For our prototype we focused on creating an effective and visually attractive news tag layer, from single news and geographic coordinates databases that we processed beforehand. Adding the capability of extracting news and geographic coordinates from different news websites in real time would make the application more practical.

Geographic content extraction is one of the other areas that require more research and development. We discussed some possible solutions for geographic content extraction in chapter three, such as using the publisher's geographic location, finding words in an article that references geographic locations, using the reader's location, and using news concept to extract geographic information from news articles. The best method for handling this issue remains an open problem.

Visualization features of the application could also be extended in a number of ways:

- Time controls: The size of words could be mapped to temporal relevance with words/objects shrinking and growing as the user adjusts a time slider.
- Layer-based filtering: User could enable or disable different types of content. These would shrink away leaving more room for the rest.
- Similarity based filtering: User could click on an object or word to indicate that they want “more like this” or “less like this”.

Our application could also be adapted for a wide variety of different uses on an interactive map. For example, it could be used for sports events such as the Olympic Games to show news, images, match schedules, match results, etc. on an interactive map of the sport venues. Specifically, a corollary project has been proposed in collaboration with The University of British Colombia to provide a Canadian application for E-government documents. This would modify our application to present Canadian government news, event, etc. on the interactive maps.

BIBLIOGRAPHY

[1] D. Frohlich, A. Kuchinsky, C. Pering, A. Don, and S. Ariss, “Requirements for photoware,” in *Proceedings of the 2002 ACM conference on Computer supported cooperative work CSCW 02*, 2002, vol. New Orlean, no. 2, pp. 166–175.

[2] D. Stout, “New Work: A Texas Designer’s Map of the World,” 2010. [Online]. Available: http://new.pentagram.com/2008/02/new-work-sappi/. [Accessed: 10-Sep-1BC].

[3] B. E. Teitler, M. D. Lieberman, D. Panozzo, J. Sankaranarayanan, H. Samet, and J. Sperling, “NewsStand: a new view on news,” in *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems - GIS ’08*, 2008, vol. 2008, no. November, pp. 1–10.

[4] S. Lee and H.-J. Kim, “News Keyword Extraction for Topic Tracking,” in *2008 Fourth International Conference on Networked Computing and Advanced Information Management*, 2008, pp. 554–559.

[5] M. Ishizuka, “Topic extraction from news archive using TF*PDF algorithm,” in *Proceedings of the Third International Conference on Web Information Systems Engineering 2002 WISE 2002*, 2002, pp. 73–82.

[6] P. Pérez, M. Gangnet, and A. Blake, “Poisson image editing,” in *ACM SIGGRAPH 2003 Papers (SIGGRAPH ’03)*, 2003, pp. 313 – 318.

[7] C. Arora, S. Banerjee, P. Kalra, and S. N. Maheshwari, “An efficient graph cut algorithm for computer vision problems,” in *Proceedings of the 11th European conference on computer vision conference on Computer vision: Part III* , 2010, pp. 552–565.

[8] Y. Boykov and V. Kolmogorov, “An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision.,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 9, pp. 1124–1137, Sep. 2004.

[9] D. Greig, B. Porteous, and A. Seheult, “Exact Maximum A Posteriori Estimation for Binary Images,” *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 271–279, 1989.

[10] V. Kolmogorov and R. Zabih, “What Energy Functions Can Be Minimized via Graph Cuts?,” in *ECCV ’02 Proceedings of the 7th European Conference on Computer Vision-Part III*, 2002, pp. 65–81.

[11] V. Kwatra, A. Schödl, I. Essa, G. Turk, and A. Bobick, “Graphcut textures,” *Graphcut textures: image and video synthesis using graph cuts*, vol. 22, no. 3, pp. 277–286, Jul. 2003.

[12] Y. Boykov, O. Veksler, and R. Zabih, “Fast approximate energy minimization via graph cuts,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 11, pp. 1222–1239, 2001.

[13] D. S. Hochbaum, “The Pseudoflow Algorithm: A New Algorithm for the Maximum-Flow Problem,” *Operations Research*, vol. 56, no. 4, pp. 992–1009, Jul. 2008.

[14] S. Lohmann, J. Ziegler, and L. Tetzlaff, “Comparison of Tag Cloud Layouts : Task-Related Performance and Visual Exploration,” in *Proceedings of the 12th IFIP TC 13 International Conference on Human-Computer Interaction: Part I*, 2009, pp. 392–404.

[15] H. Halpin, V. Robu, and H. Shepherd, “The complex dynamics of collaborative tagging,” in *Proceedings of the 16th international conference on World Wide Web - WWW ’07*, 2007, pp. 211–220.

[16] V. Robu, H. Halpin, and H. Shepherd, “Emergence of consensus and shared vocabularies in collaborative tagging systems,” *ACM Transactions on the Web*, vol. 3, no. 4, pp. 1–34, Sep. 2009.

[17] D.-Q. Nguyen and H. Schumann, “Taggram: Exploring Geo-data on Maps through a Tag Cloud-Based Visualization,” in *2010 14th International Conference Information Visualisation*, 2010, pp. 322–328.

[18] A. Slingsby, J. Dykes, J. Wood, and K. Clarke, “Interactive Tag Maps and Tag Clouds for the Multiscale Exploration of Large Spatio-temporal Datasets,” in *11th International Conference Information Visualization (IV ’07)*, 2007, pp. 497–504.

[19] Y. Hassan-Montero and V. Herrero-Solana, “Improving tag-clouds as visual information retrieval interfaces,” *Multidisciplinary Information*, vol. I, no. 2, pp. 25–28, 2006.

[20] B. Y.-L. Kuo, T. Hentrich, B. M. . Good, and M. D. Wilkinson, “Tag clouds for summarizing web search results,” in *Proceedings of the 16th international conference on World Wide Web - WWW ’07*, 2007, pp. 1203–1204.

[21] O. Kaser and D. Lemire, “Tag-Cloud Drawing: Algorithms for Cloud Visualization,” vol. CoRR abs/c, Mar. 2007.

[22] C. Seifert, B. Kump, W. Kienreich, G. Granitzer, and M. Granitzer, “On the Beauty and Usability of Tag Clouds,” in *12th International Conference Information Visualisation*, 2008, pp. 17–25.

[23] B. Shaw, “Utilizing Folksonomy : Similarity Metadata from the Del . icio . us System CS6125 Project,” 2005, pp. 1–10.

[24] K. Bielenberg and M. Zacher, “Groups in Social Software : Utilizing Tagging to Integrate Individual Contexts,” Universität Bremen, 2005.

[25] A. K. Jain, M. N. Murty, and P. J. Flynn, “Data clustering: a review,” *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, Sep. 1999.

[26] M. J. Halvey and M. T. Keane, "An assessment of tag presentation techniques," in *Proceedings of the 16th international conference on World Wide Web - WWW '07*, 2007, pp. 1313 – 1314.

[27] J. Sinclair and M. Cardew-Hall, "The folksonomy tag cloud: when is it useful?," *Journal of Information Science*, vol. 34, no. 1, pp. 15–29, May 2007.

[28] A. W. Rivadeneira, D. M. Gruen, M. J. Muller, and D. R. Millen, "Getting our head in the clouds: toward evaluation studies of tagclouds ," in *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '07*, 2007, pp. 995–998.

[29] S. Milgram, "Psychological Maps of Paris, The Individual in a Social World," 1977, pp. 68– 90.

[30] J. Schrammel, M. Leitner, and M. Tscheligi, "Semantically structured tag clouds: an empirical evaluation of clustered presentation approaches," in *Proceedings of the 27th international conference on Human factors in computing systems - CHI '09*, 2009, pp. 2037–2040.

[31] S. Bateman, C. Gutwin, and M. Nacenta, "Seeing things in the clouds: the effect of visual features on tag cloud selections ," in *Proceedings of the nineteenth ACM conference on Hypertext and hypermedia - HT '08*, 2008, pp. 193–202 .

[32] M. Luboschik, H. Schumann, and H. Cords, “Particle-based labeling: Fast point-feature labeling without obscuring other visual features.,” *IEEE transactions on visualization and computer graphics*, vol. 14, no. 6, pp. 1237– 1244, Jan. 2008.

[33] P. E. H., D.G. Stork R.O. Duda, “ R.O. Duda, P.E. Hart, and D.G. Stork, Pattern Classification, New York: John Wiley & Sons, 2001, pp. xx + 654, ISBN: 0-471-05669-3,” *Journal of Classification*, vol. 24, no. 2, pp. 305 – 307 , Sep. 2007.

[34] C. Rother, L. Bordeaux, Y. Hamadi, and A. Blake, “AutoCollage,” *ACM Transactions on Graphics*, vol. 25, no. 3, pp. 847–852, Jul. 2006.

[35] M. A. Elliott, C. S. Herrmann, A. Mecklinger, and H. J. Müller, “The loci of oscillatory visual-object priming: a combined electroencephalographic and reaction-time study,” *International Journal of Psychophysiology*, vol. 38, no. 3, pp. 225–241, Dec. 2000.

[36] M. Kass, A. Witkin, and D. Terzopoulos, “Snakes: Active contour models,” *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, 1988.

[37] J. L. Prince, “Gradient vector flow: a new external force for snakes,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1997, pp. 66–71.

[38] C. Jin, “On real time active contours,” in *Proceedings of the 48th Annual Southeast Regional Conference on - ACM SE ’10*, 2010, pp. 1–6.

[39] A. A. Amini, T. E. Weymouth, and R. C. Jain, "Using dynamic programming for solving variational problems in vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 9, pp. 855–867, 1990.

[40] P. Radeva and J. Serrat, "Rubber Snake : Implementation on signed distance potential," in *Proc of International Conference SWISS VISION93*, 1993, vol. 93, pp. 187–194.

[41] L. D. Cohen and I. Cohen, "Finite-element methods for active contour models and balloons for 2-D and 3-D images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 11, pp. 1131–1147, 1993.

[42] P. Viola and M. Jones, "Robust Real-time Object Detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137 – 154 , 2004.

[43] J. Canny, "A Computational Approach to Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.

[44] Matlab on-line help, "Morphological Reconstruction," *The MathWorks*, 2012. [Online]. Available: http://www.mathworks.com/help/toolbox/images/f18-16264.html. [Accessed: 20-Jul-1BC].

[45] N. Diakopoulos and I. Essa, "Mediating photo collage authoring," in *Proceedings of the 18th annual ACM symposium on User interface software and technology - UIST '05*, 2005, pp. 183 – 186.

[46] T. Liu, J. Wang, J. Sun, N. Zheng, X. Tang, and H.-Y. Shum, “Picture Collage,” *Trans. Multi.*, vol. 11, no. 7, pp. 1225–1239, 2009.

[47] T. F. Chan and L. a Vese, “Active contours without edges.,” *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, vol. 10, no. 2, pp. 266–277, Jan. 2001.