11-2010

# 3D oceanographic data compression using 3D-ODETLAP

You Li

Tsz-Yam Lau

Christopher S. Stuetzle

Peter Fox

W. Randolph Franklin

# 3D oceanographic data compression using 3D-ODETLAP

**Article** *in* SIGSPATIAL Special · November 2010

DOI: 10.1145/1953102.1953104 · Source: DBLP

Christopher Stuetzle
Merrimack College

Peter Fox
Rensselaer Polytechnic Institute

# The SIGSPATIAL Special

Newsletter of the Association for Computing Machinery Special Interest Group on Spatial Information

Volume 2 Number 3 November 2010

# The SIGSPATIAL Special

The SIGSPATIAL Special is the newsletter of the Association for Computing Machinery (ACM) Special Interest Group on Spatial Information (SIGSPATIAL).

ACM SIGSPATIAL addresses issues related to the acquisition, management, and processing of spatially-related information with a focus on algorithmic, geometric, and visual considerations. The scope includes, but is not limited to, geographic information systems.

Current ACM SIGSPATIAL officers are:
Chair, Hanan Samet, University of Maryland
Vice-Chair, Walid G. Aref, Purdue University
Secretary, Chang-Tien Lu, Virginia Tech
Treasurer, Markus Schneider, University of Florida
Newsletter Editor, Egemen Tanin, University of Melbourne

For more details and membership information for ACM SIGSPATIAL as well as for accessing the newsletters please visit http://www.sigspatial.org.

The SIGSPATIAL Special serves the community by publishing short contributions such as SIGSPATIAL conferences' highlights, calls and announcements for conferences and journals that are of interest to the community, as well as short technical notes on current topics. The newsletter has three issues every year, i.e., March, July, and November. For more detailed information regarding the newsletter or suggestions please contact the editor via email at egemen@csse.unimelb.edu.au.

Notice to contributing authors to The SIGSPATIAL Special: By submitting your article for distribution in this publication, you hereby grant to ACM the following non-exclusive, perpetual, worldwide rights:

- to publish in print on condition of acceptance by the editor,
- to digitize and post your article in the electronic version of this publication,
- to include the article in the ACM Digital Library,
- to allow users to copy and distribute the article for noncommercial, educational or research purposes.

However, as a contributing author, you retain copyright to your article and ACM will make every effort to refer requests for commercial use directly to you.

Notice to the readers: Opinions expressed in articles and letters are those of the author(s) and do not necessarily express the opinions of the ACM, SIGSPATIAL or the newsletter.

**The SIGSPATIAL Special (ISSN 1946-7729) Volume 2, Number 3, November 2010.**

# Table of Contents

ACM SIGSPATIAL GIS 2010 PhD Showcases

Announcements

# Editorial

Dear Colleagues,

Welcome to the November 2010 issue of the SIGSPATIAL Special. This issue is dedicated to the PhD Showcases from the 18th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (ACM SIGSPATIAL GIS 2010) held in San Jose, California on November 2-5, 2010. Our goal is for the appearance of this issue to coincide with the conference so that all attendees will have simultaneous access to all of the presentations at the conference. We would also like to bring to your attention the continued use of the SIGSPATIAL logo starting with the July 2010 issue. The logo was created by our long term sponsor ESRI under the guidance of Erik Hoel.

This issue is organized as follows. The first item is a letter describing the highlights of SISAP 2010, the International Conference on Similarity Search and Applications which was held in cooperation with SIGSPATIAL from September 18th through the 19th in İstanbul, Turkey. The newsletter continues with the presentation of five PhD Showcases from ACM SIGSPATIAL GIS 2010. Next, we present a call for papers for Ubicomp 2011 which will be held in Beijing, China in September 2011 in cooperation with SIGSPATIAL. The issue concludes with membership information for ACM and SIGSPATIAL.

Looking forward to your comments and suggestions.

Egemen Tanin, Editor
Department of Computer Science and Software Engineering
University of Melbourne, Victoria 3010, Australia
Tel: +61 3 8344 1350
Fax: +61 3 9348 1184
Email: egemen@csse.unimelb.edu.au

# Highlights from SISAP 2010 The 3rd International Conference on Similarity Search and Applications (İstanbul, Turkey - September 18–19, 2010)

Vladimir Pestov

University of Ottawa

(Invited Speaker)

www.sisap.org/2010

## Subject matter of the conference

This year's SISAP conference was the third event in a relatively new annual conference series, held in İstanbul, Turkey after the very successful workshops in Cancun, Mexico (2008) and Prague, Czech Republic (2009).

The International Conference on Similarity Search and Applications (SISAP) is a series devoted to similarity searching. A typical object of study is a domain, $\mathbb{U}$, equipped with a (dis)similarity measure $\varsigma(x, y)$, which may or may not satisfy the axioms of a metric, and a dataset, or instance, $\mathbb{X}$, contained in $\mathbb{U}$. The problem consists of building an indexing scheme supporting an efficient and effective retrieval of nearest neighbours in $\mathbb{X}$ to a given query point $q \in \mathbb{U}$. One may be interested in the exact NN search, or approximate NN search, range search, etc. In view of the role played by similarity-based information retrieval, either on its own or as a component of various data mining and machine learning algorithms — in particular in multimedia applications as well as in sequence-based biology — the subject matter of the SISAP series is certainly important.

Specifically, the SISAP conference series aims to fill in the gap left by the various scientific venues devoted to similarity searching in spaces with coordinates, by providing a common forum for theoreticians and practitioners around the problem of similarity searching in general spaces (metric and non-metric) or using distance-based (as opposed to coordinate-based) techniques in general. Here are some topics of interest:

Range search
$k$-NN search
Limited range and reverse $k$-NN search
New complex similarity query types
Similarity joins
Languages for similarity databases
Clustering (applied to indexing)
Parallelism & distributed algorithms
Approximate searching
Computation of intrinsic dimension
Cost models
Embeddings

SISAP is seen as a forum for exchanging real-world applications, new indexing techniques, common testbeds and benchmarks, and source code. Authors are expected to use the testbeds and code from the SISAP web site[1] for comparing new applications, databases, indexes and algorithms.

## Conference structure

SISAP 2010 was organized in cooperation with the ACM Special Interest Group on Spatial Information (SIGSPATIAL) and the papers are indexed in the ACM Digital Library. This year's program co-chairs were Paolo Ciaccia and Marco Patella (both from Università di Bologna, Italy). Local organization was done by Çengiz Celik (Bilkent University, Turkey). The program committee included 9 researchers.

Contributions to the conference fall in the three main categories: full papers, posters, and demos. Here is some statistics to give an idea of the event scale.

| | Invited | Full | Posters | Demos | Total | Submitted | Acceptance rate |
|---|---|---|---|---|---|---|---|
| 1st SISAP | 2 | 15 | - | - | 17 | 33 | 45% |
| 2nd SISAP | 2 | 14 | 2 | 7 | 25 | 34 | 60% |
| 3rd SISAP | 2 | 11 | 2 | 6 | 21 | 29 | 66 % |

Even though the event is still small in size, one can feel its vigour and vitality. The conference ran as a single-track (with no parallel sessions) over one day and a half, leaving sufficient time for scientific discussions.

The two invited papers chosen by the program chairs provided quite a contrasting view of the subject. The author of this report gave a talk entitled *Indexability, concentration, and VC theory,* exploring some geometric and complexity-theoretic aspects of the curse of dimensionality affecting some indexing schemes. This talk was largely driven by a mathematician's curiosity and based on a number of standard assumptions about the metric access model.

Such standard assumptions, often assumed uncritically, were put to a severe test in the invited talk by Tomáš Skopal, revealingly entitled *Where Are You Heading, Metric Access Methods? A Provocative Survey*. This talk offered a fresh and very interesting perspective of a range of actual problems in the area as seen through the eyes of one of the most active current developers of models and software for content-based retrieval.

The majority of the talks' slides are available on the conference web site.[2] The four best papers will receive invitations (subject to additional reviewing) to be published in a special issue of Journal of Discrete Algorithms (Elsevier). At the moment of writing this report, a final selection has not yet been made. For this reason, the author of the report feels compelled to make his own (very subjective) pick and mention a few memorable contributions.

Pivot-based indexing schemes are about the most commonly used and best understood, in particular the paper by Sergey Brin stands out as an often-cited classics. Building better pivot-based schemes through dimensionality reduction was the subject of the talk by Rui Mao who reported

[1] `http://www.sisap.org/Metric_Space_Library.html`

[2] `http://www.sisap.org/2010/?src=program`

on a joint work with W. Miranker and D. Miranker. Ilaria Bartolini and Corrado Romani (Università di Bologna) presented an efficient and effective similarity-based video retrieval engine. An audio similarity retrieval engine was the subject of a demonstration by Pavel Jurkas, Milan Štefina, David Novak and Michal Batko (all of Masaryk University). Even though it is at its prototype stage, with just over a thousand sound sequences being indexed, the engine raises a number of interesting questions both at the theoretical and practical level. The quadratic distances are of great importance for instance in content-based image and video retrieval, and at the same time they are computationally expensive. The group from Aachen University — Christian Beecks, Merih Seran Uysal, and Thomas Seidl — have presented a new efficient method for computing the quadratic distance signature computation through similarity matrix compression.

## Varia

A large number of participants (about a half) came from the Czech Republic. This reflects on the importance of the Czech school of similarity search, largely due to the influential work of Pavel Zezula. The strong Italian school was also well-represented (5 participants), while the computer scientists from Latin America, where another major school of similarity search is based, were largely absent, in part due to the problem of getting a Turkish visa.

Figure 1: Some conference participants against the backdrop of Topkapı Palace. (Photo ©Paolo Ciaccia.)

The conference was held at Hotel Armada, and the lunches were served in its rooftop restaurant, described by the Lonely Planet guide to İstanbul as the best one that the tourist neighbourhood of

Sultanahmet has to offer. Another rooftop restaurant — that of the Aşkın Hotel, where the majority of participants were staying and the breakfasts were served — commanded a spectacular view of the Bosphorus strait, as well as two of the most famous mosques of İstanbul.

The local organization has not always been smooth, and some issues with local organization by the tourist agency (Star Tours/Star Turizm Ankara) proved to be a bit of a nuisance. In spite of this, the participants were unanimous in praising the impressive "Bosphorus by night" boat trip (which was held in place of the traditional conference dinner), as well as a guided tour to Hagia Sophia and Topkapı Palace, led by a highly competent and friendly guide.

Among the ideas floated at the conference was a possibility of aligning the future SISAP events with one of the larger and well-established conferences in databases and/or data mining, specifically promoted by Pavel Zezula.

If the fifth SISAP event (2012) was to be held as a stand-alone, among the prospective venues considered is Ottawa, the capital city of Canada. Meanwhile, the fourth SISAP conference will be held on June 30 – July 1, 2011 in Italy, on the island of Lipari, near Sicily. The program chairs are Alfredo Ferro, Alfredo Pulvirenti, and Rosalba Giugno.

# ACM SIGSPATIAL GIS 2010 PhD Showcases

# PhD Showcase: 3D Oceanographic Data Compression Using 3D-ODETLAP

PhD Student:You Li
Rensselaer Polytechnic Institute (RPI)
110, Eighth Street
Troy, NY, USA
liy13@cs.rpi.edu

PhD Student:Tsz-Yam Lau
RPI
110, Eighth Street
Troy, NY, USA
laut@cs.rpi.edu

PhD Student:Christopher S. Stuetzle
RPI
110, Eighth Street
Troy, NY, USA
stuetc@cs.rpi.edu

PhD Supervisor:Peter Fox
RPI
110, Eighth Street
Troy, NY, USA
pfox@cs.rpi.edu

PhD Supervisor:W. Randolph Franklin
RPI
110, Eighth Street
Troy, NY, USA
wrf@ecse.rpi.edu

## ABSTRACT

This paper describes a 3D environmental data compression technique for oceanographic datasets. With proper point selection, our method approximates uncompressed marine data using an over-determined system of linear equations based on, but essentially different from, the Laplacian partial differential equation. Then this approximation is refined via an error metric. These two steps work alternatively until a predefined satisfying approximation is found.

Using several different datasets and metrics, we demonstrate that our method has an excellent compression ratio. To further evaluate our method, we compare it with 3D-SPIHT. 3D-ODETLAP averages 20% better compression than 3D-SPIHT on our eight test datasets, from World Ocean Atlas 2005. Our method provides up to approximately six times better compression on datasets with relatively small variance. Meanwhile, with the same approximate mean error, we demonstrate a significantly smaller maximum error compared to 3D-SPIHT and provide a feature to keep the maximum error under a user-defined limit.

## Categories and Subject Descriptors

I.3.5 [**Computing Methodologies**]: Computer Graphics Computational Geometry and Object Modeling

## General Terms

Algorithm, Experimentation, Performance

## Keywords

PDE solver, 3D, Compression, Oceanographic

The SIGSPATIAL Special, Volume 2, Number 3, November 2010.

## 1. INTRODUCTION

As technology progresses, the availability of massive oceanographic data with global spatial coverage has become quite common. Various types of data, including salinity, temperature and oxygen of sea water, require measurement and storage in 3D, or even 4D over time for historical record. Researchers expect to be able to transmit these data over the internet, exposing them to the public, since the data provide a raw source of information about the environment we are living in.

Nevertheless, the research of processing and manipulating 3D oceanographic data has not advanced with the data inflation. Marine datasets are still stored as 3D value matrices where traditional compression algorithms don't perform well, and it's rare to find specially designed algorithms to compress 3D marine data. For example, Salinity[2] and Nutrients[5] in World Ocean Atlas 2005 are compressed with gzip, which was originally designed for plain text compression. Images have been produced for observation and analysis. Figure 1 visualizes the one original marine dataset [4] derived from WOA 2005.

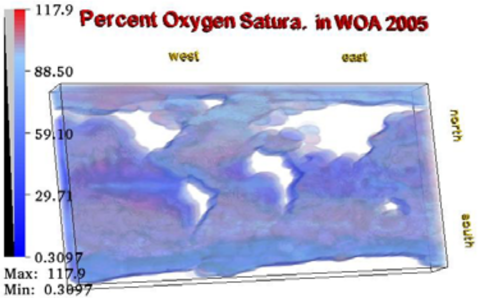


**Figure 1:** $180 \times 360 \times 33$ Percentage oxygen saturation data

In this paper, we use a 3D Over-determined Laplacian Partial Differential Equation (3D-ODETLAP) to approximate and lossily compress 3D marine data. First we construct an over-determined system using regular grid point selection; we then use an over-determined PDE to solve for a smooth approximation. The initial approximation might be very coarse due to the limited number of selected points. Furthermore, we refine this approximation with respect to the original marine data by adding points with the largest error, and running 3D-ODETLAP again on the augmented representation to gain a better approximation. These two steps work alternately until a stopping criteria is reached, which is usually the maximum error.

## 2. PRIOR ART

Generally speaking, most of the methods used for 3D environmental scalar data compression are based on image and video compression techniques since they provide good compression with no (or low) loss of information. 2D and 3D wavelet-based methods are among the best ones.

In general, 2D compression techniques such as 2D wavelets decomposition, JPEG compression, and JPEG2000 compression decompose the data in rectangular sub-blocks and each block is transformed independently. Furthermore, the image data is represented as a hierarchy of resolution features and its inverse at each level provides sub-sampled version of the original image. 3D environmental scalar data can be split into 2D image slices. Therefore the above advanced 2D compression techniques can be applied on the slices.

While the above methods are basically applications of the 2D method on 3D data, there are also truly 3D volumetric data compression techniques. Muraki[7] introduced the idea of using a 3D wavelet transformation for approximation and compression of 3D scalar data. Luo et al[6] used a modified embedded zero tree wavelet (EZW) algorithm[11] for compression of volumetric medical images. The EZW-approach has been improved upon by Said and Pearlman[9] with the SPIHT algorithm for images. This approach has been extended to a 3D-SPIHT algorithm which was used for video coding and with excellent results also for medical volumetric data[14] using an integer wavelet packet transform. A compression performance comparison has been conducted between 3D-SPIHT and our proposed 3D-ODETLAP method later in this paper.

## 3. 3D-ODETLAP

### 3.1 Definition

As implied by the name, 3D-ODETLAP, or Three Dimensional Over-Determined Laplacian Partial Differential Equation, is an extension of a Laplacian PDE $\frac{\delta^2 z}{\delta x^2} + \frac{\delta^2 z}{\delta y^2} = 0$ to an overdetermined system of equations[12, 13]. Each unknown point induces an equation setting it to the average of its 3, 4, 5 or 6 neighbors in three dimensional space. We have the equation:

$$\begin{aligned} u_{i,j,k} = (&u_{i-1,j,k} + u_{i+1,j,k} + u_{i,j-1,k} \\ &+ u_{i,j+1,k} + u_{i,j,k-1} + u_{i,j,k+1})/6 \end{aligned} \tag{1}$$

for every unknown non-border point, which is equivalent to saying the volume satisfies 3D Laplacian PDE,

$$\frac{\delta^2 u}{\delta x^2} + \frac{\delta^2 u}{\delta y^2} + \frac{\delta^2 u}{\delta z^2} = 0 \tag{2}$$

In marine modeling this equation has the following limitations:

- The solution of Laplace equation never has a relative maximum or minimum in the interior of the solution domain, this is called the maximum principle[10] so local maxima are never generated.
- For different marine data distribution, the sole solution may not be the optimal one.

To avoid these limitations, an over-determined version of the Laplacian equation is defined as follows: apply the equation 1 to every non-border point, both known and unknown, and a new equation is added for a set S of known points:

$$u_{i,j,k} = h_{i,j,k} \tag{3}$$

where $h_{i,j,k}$ stands for the known value of points in S and $u_{i,j,k}$ is the "computed" value as in equation 1.

Because the number of equations exceeds the number of unknown variables, this means the system is over-determined. Since the system is very likely to be inconsistent, instead of solving it for an exact solution (which is now impossible), an approximated solution is obtained by trying to keep the error as small as possible. Equation 1 is approximately satisfied for each point, making it the average of its neighbors, which makes the generated surface smooth and resembles the real world situation. However, since we have known points where equation 3 is valid, they are not necessarily equal to the average of their neighbors. This is especially true when we have adjacent known points. Therefore, for points with multiple equations we can choose the relative importance of accuracy versus smoothness by adding a smoothness parameter when solving the over-determined system[3].

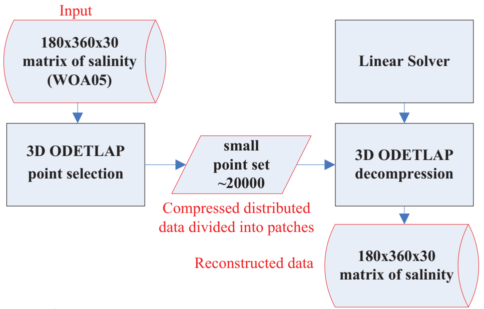


**Figure 2: 3D-ODETLAP Algorithm Outline**

In our implementation, equation 1 is weighted by R relative to equation 3, which defines the known locations. So a small R will approximate a determined solution and the surface will be more accurate while a very large R will produce a surface with no divergence, effectively ignoring the known points. Instead of interpolation, approximation is a more suitable term for this method because the reconstructed volume is not guaranteed to go through the input data points. 3D-ODETLAP can be used as a lossy compression technique since the original terrain can be approximated with some error using the set of points S for equations 1 and 3.

### 3.2 Algorithm Outline

```
Input: $3D - MarineData : V$
Output: $PointSet : S$
$S = RegGridSelection(V)$
$Reconstructed = 3D - ODETLAP(S)$
while $MeanError > Max\_MeanError$ do
    $S = S \cup Refine(V, Reconstructed)$
    $Reconstructed = 3D - ODETLAP(V)$
end
return $S$
```

**Algorithm 1**: 3D-ODETLAP algorithm pseudo code

The 3D-ODETLAP algorithm's outline is shown in Figure 2 and the pseudo code is given below. Starting with the original marine volume data matrix, there are two point selection phases: firstly, the initial point set S is built by a simple regular grid selection scheme and a first approximation is computed using the equations 1 and 3. Given the reconstructed surface, a stopping condition based on an error measure is tested. In practice, we have used the mean percent error as the stopping condition. If this condition is not satisfied, the second step is executed. In this step, k≥1 points with the biggest error are selected based on our "forbidden zone" method described below and then added into the existing point set S; this extended set is used by 3D-ODETLAP to compute a more refined approximation. As the algorithm proceeds, the total size of point set S increases and the total error converges.

## 3.3 Forbidden Zone

The refined point selection strategy has flaws in that those selected points are often clustered due to high value fluctuation within a small region. In oceanographic datasets, if one point with large error is far away from others, it is most likely that its adjacent points are also erroneous and will be selected as well. Therefore, refined points selected may be redundant in some regions, which is a waste in compression.

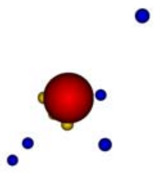


**Figure 3: Forbidden Zone check**

Forbidden zone is the check process we use to add new refined points: the spatial local neighbors of the new point will be checked to see if there is any existing refined points added in the same iteration. If yes, this new point is abandoned and the point with next biggest error is tested until we have a predefined number of refined points. In Figure 3, the big red sphere is the forbidden zone of one point. The yellow points are inside this sphere and thus not included. All the blue points are outside the zone and included.

## 3.4 Implementation Speed-up

### *3.4.1 Normal Equation*

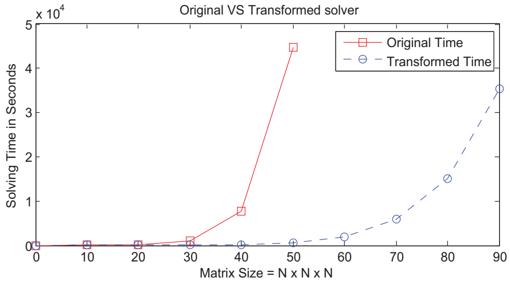


**Figure 4: The red line represents the solving time of original solver and the blue line represents that of transformed solver by normal equations**

In order to solve the linear system $Ax = b$ when A is non-symmetric, we can solve the equivalent system

$$A^T Ax = A^T b \tag{4}$$

which is Symmetric Positive Definite in our case because in our *over-determined* system, $A$ is a rectangular matrix of size $n \times m$, $m < n$. This system is known as the system of the *normal equations* associated with the least-squares problem,

$$minimize||b - Ax||^2 \tag{5}$$

Before applying the *normal equations* method on our *over-determined sparse linear system*, our underlying solver to solve the over-determined system uses sparse QR decomposition in Matlab. It runs much slower than the Cholesky factorization, which solves Symmetric Positive Definite linear system as shown in Figure 4. Even with the overhead introduced by matrix multiplication, the *normal equations* method still solves our linear system significantly faster. The *normal equations* method enables us to solve small linear system very quickly, which makes our next scheme *Dividing into Boxes* feasible.

### *3.4.2 Dividing into Boxes*

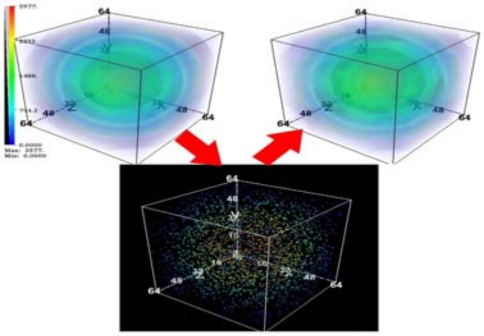


**Figure 5: Reconstruction from 1% random selected points using weighted average method**

As you can see from Figure 4, solving a reasonable large 3D matrix, i.e., 100 × 100 × 100, still takes too much time. Linear regression test shows that 3D-ODETLAP is running approximately at $T = \Theta(n^6)$ for an $n \times n \times n$ matrix. And the memory cost is also high: solving a 90 × 90 × 90 3D dataset takes about 9.8 CPU-hours and 55 GB of writable memory on four 2.4GHZ processors workstation with 60 GB of main memory running Ubuntu 9.04 and 64-bit Matlab R2009a. This is unacceptable because the 3d datasets in the real world is often significantly larger than the test one and the running time may be prohibitively long.

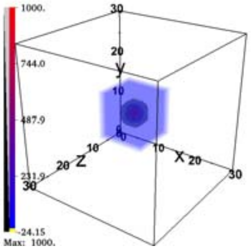


**Figure 6: Altering a single known point in the data has a limited radius of impact during reconstruction.**

Within datasets, some points are quite distant and thus have nearly no influence on each other. Figure 6 shows that when we assign a high value to one point in the input to 3D-ODETLAP, only those points within a small neighborhood of this point are affected. Beyond that small region, the effect becomes negligible. This supports our hypothesis that it should be possible to divide large data sets into separate boxes, run 3D-ODETLAP on them individually, and achieve similar results to the non-box 3D-ODETLAP solution. Specifically, we created a 64 × 64 × 64

| | Size(in bytes) | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Salinity | Temperature | Dissolved oxygen | Apparent oxygen uti. | Percentage oxygen satura. | Phosphate | Nitrate | Silicate |
| Compr.xyz(RLE) | 5983 | 3913 | 3994 | 4904 | 5026 | 5026 | 6118 | 4129 |
| Compr.v | 21394 | 17595 | 17046 | 19871 | 21717 | 19417 | 25787 | 18947 |
| compr.(xyz+v) | 27377 | 21508 | 21040 | 24775 | 26743 | 24493 | 31905 | 23076 |
| Bzip2 | 36733 | 25687 | 24082 | 27829 | 35649 | 27658 | 39844 | 31018 |
| Size Reduction(%) | 34.17% | 19.42% | 14.45% | 12.32% | 33.30% | 12.92% | 24.88% | 34.41% |

**Table 1: File size comparison after compression between Run length encoding method and simply storing quadruplets which contains three coordinates and values. Our RLE method further reduces file size ranging from 12% to 34% as indicated in the fourth row**

3D matrix $A$ based on function $A = (max\_distance - distance)^2$, where $max\_distance$ is the maximum distance from any points within the cube to its center and distance is the distance of this point to the center of the cube. Then we divide compressed $A$ into a $4 \times 4$ matrix whose elements are $16 \times 16 \times 16$ boxes. Then we run 3D-ODETLAP on each box individually and merge them together as shown in Figure 5.

Nevertheless, if we simply merge all the boxes together, we tend to have large errors at the edge and plane of a box, where there is less information to work with than running 3D-ODETLAP on the matrix as a whole.

Our solution is as follows: run two iterations on these data. First, 3D-ODETLAP is run individually on each box. Then, 3D-ODETLAP is run on the inner $48 \times 48 \times 48$ matrix which is the inner part of matrix A with each box size of $16 \times 16 \times 16$. These two iterations enable us to have two calculated values for each point in the inner $48 \times 48 \times 48$ matrix and only one value for the rest of matrix $A$. Simply taking the average of both calculated values will reduce the errors at the edges and planes to some extent. But since we run 3D-ODETLAP redundantly on each point, we have the option to bias those erroneous edge and plane points' values and select those values close to the center of another box. In order to ignore those values of edges and plane points, we use a *Euclidean Distance* to assign weight to the two values of each point. The closer the point is to the center of its box, the more weight its value has and vice versa. Weighted average method produces smaller errors overall than others.

| Variable | Unit | Data Range | Size(Mb) |
|---|---|---|---|
| Salinity | PPS | [5.00, 40.90] | 8.16 |
| Temperature | ° | [-2.08, 29.78] | 8.16 |
| Dissolved oxygen | $\mathrm{m}l l^{-1}$ | [0, 9.55] | 8.16 |
| Apparent oxygen utilization | $\mathrm{m}l l^{-1}$ | [-1.43, 7.87] | 8.16 |
| Percent oxygen saturation | % | [0.31, 117.90] | 8.16 |
| Phosphate | $\mu$M | [0, 4.93] | 8.16 |
| Nitrate | $\mu$M | [0, 54.45] | 8.16 |
| Silicate | $\mu$M | [0, 256.24] | 8.16 |

**Table 2: Before compression, These are annual objectively analyzed climatology datasets on 33 standard depth levels from sea surface to 5500 metres on all 8 variables. Each point uses 4 byte to store its value in single precision and thus total size of each dataset is 8.16 Mb.**

# 4. FURTHER COMPRESSION

The 3D spatial coordinates *(x,y,z)* are different from value $v$ because they distribute evenly within the range $[1, 128]$, $[1, 128]$ and $[1, 32]$ respectively in our test data. But values $v$ v are distributed more closely and require higher precision. The run-length encoding is a simple lossless compression technique and it stores the value and the count of sequence with the same value. Because the *(x,y,z)* values correspond to positions in a 3D matrix, we only need to store a binary value in each position to indicate whether this point is selected or not. So we define *Run* as a consecutive sequence of 0's or 1's to represent the selection of one point at each location. Thus, given a binary matrix of $N1 \times N2 \times N3$, we use simple run-length encoding to store value of *Run*.

# 5. RESULT AND ANALYSIS

## 5.1 Result on Oceanographic Data

We tested 3D-ODETLAP on actual marine data–WOA 2005, which is provided by NODC (National Oceanographic Data Center). WOA 2005 is a set of objectively analyzed climatological fields on a 1-degree latitude-longitude grid at 33 standard depth levels from sea surface to 5500 metres. These variables include temperature, salinity, oxygen, oxygen saturation, Apparent Oxygen Utilization (AOU), phosphate, silicate and nitrate. Table 2 shows the datasets in detail. In the following experiments, all of our test data are derived from these datasets.

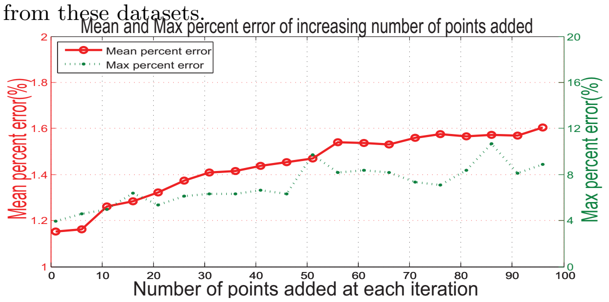


**Figure 7: This figure shows the Mean(left) and Max(right) percentage error as the number of points added at each iteration increases from 1 to 96. The dataset is $64 \times 64 \times 32$ 3D matrix derived from Percent Oxygen Saturation in WOA 2005. The number of selected points is 2100, smooth variable R is set to 0.01, sub-box is of size $32 \times 32 \times 32$, initial point selection picks every 5 along the regular grid and forbidden zone is of size 3.**

In our implementation, several parameters can affect the effectiveness of our method, including initial point selection, points added at each iteration, smooth variable R and size of forbidden zone. In order to achieve a high compression ratio while keeping the errors beneath a tolerant level, we approach as close as possible the optimal parameters for the all eight datasets by finding the best set of parameters for one dataset. This set of parameters works reasonably well with other datasets and if given enough computing resources, even better parameters can be found.

Figure 7 shows that as we increase the number of points added at each iteration, both mean error and max error are getting larger. But this doesn't mean we should keep this number to minimum, because there is still a trade-off between computing time and compression ratio with a predefined error.

Besides the above parameters, the size of each sub-box also has an impact on the compression performance. As table 3 shows, the larger the size of the sub-box, the better performance we will gain. However, the complexity of our algorithm is extremely high as shown in Figure 4, so we chose the largest possible sub-box while keeping running time at tolerant level. For our test on WOA 2005 data, sub-box of size $30 \times 30 \times 30$ runs in about 2000 seconds at each iteration on four 2.4GHZ processors workstation with 60 GB of main memory running Ubuntu 9.04 and 64-bit Matlab R2009a. This is acceptable and we used a sub-box of size $30 \times 30 \times 30$ in all our tests, considering that there are only 33 levels of depth in WOA 2005.

| Box Size | Mean Error(%) | Max Error(%) | Time (seconds) |
|---|---|---|---|
| $8 \times 8 \times 8$ | 1.8554 | 8.5103 | 150.40 |
| $16 \times 16 \times 16$ | 1.3219 | 5.7181 | 293.90 |
| $32 \times 32 \times 32$ | 1.1179 | 5.5175 | 1250.00 |

**Table 3: This shows the result of running 3D-ODETLAP with different sub-box size on part of Percent Oxygen Saturation data from surface to 4000 metres, which is a 3D matrix of size $128 \times 128 \times 32$. Other parameters remain the same for all three tests.**

Based on our implementation, we applied 3D-ODETLAP on the data in 2. This gave us eight distinct datasets in the size of $180 \times 360 \times 30$, which is reasonably large for testing purposes. In table5, we show that with a tolerant absolute percentage mean and max error, 3D-ODETLAP can achieve great compression ratio for all eight test datasets. Furthermore, the Silicate dataset from Table 5 and Table 4 has about the same mean percentage error: 0.9969% and 0.9996% respectively. But the compression ratio on the larger dataset is 165:1 while the one on the smaller dataset is only 81:1. This implies that 3D-ODETLAP may perform even better on larger datasets.

## 5.2 Compression Comparison with 3D-SPIHT

We have reported on experiments using all eight large oceanographic datasets extracted from WOA 2005. The size of each dataset is $128 \times 128 \times 32$ with each scalar caring a single-precision value, stored in 4 bytes each and resulting in a total size of 2 Mbytes. The same data set was used for compression in our experiments with the 3D-SPIHT method in order to provide an objective comparison of the compression performance of the two algorithms.

The quality of the compression is measured as above in terms of the mean and max percentage error over the range of the current dataset. The results are given in Table 4. Please note that we used bzip2 to further compress our results. Because 3D-SPIHT can't compress its results using bzip2 further, the comparison is fair. From Table 4, we can see that while intentionally keeping mean percentage error at approximately 0.15% for salinity dataset and 1% for the seven other datasets, 3D-ODETLAP generally produces a better compression ratio on nearly all eight different datasets and even better maximum percentage error. Except that 3D-SPIHT performs better on temperature data than 3D-ODETLAP. In fact, we can specify the maximum max error when running 3D-ODETLAP to meet the special needs of some applications, while the 3D-SPIHT compression scheme cannot.

| Variable | Mean Error(%) | Max Error(%) | Compress. File Size | Comp. Ratio |
|---|---|---|---|---|
| Salinity | 0.1196 | 1.0870 | 60,037 | 130:1 |
| Temper. | 0.7283 | 4.1620 | 69,375 | 112:1 |
| Dis. oxy. | 1.3055 | 8.7205 | 65,501 | 119:1 |
| A. O. U. | 1.4942 | 10.3178 | 59,019 | 132:1 |
| P. O. S. | 1.4930 | 8.3470 | 73,832 | 105:1 |
| Phospha. | 1.0581 | 7.4413 | 62,693 | 124:1 |
| Nitrate | 1.3299 | 10.8315 | 71,060 | 109:1 |
| Silicate | 0.9969 | 9.8598 | 46,998 | 165:1 |

**Table 5: This demonstrates the result of running 3D-ODETLATP on the 8 datasets derived from WOA 2005. Smooth variable R is set to 0.01, sub-box is of size $30 \times 30 \times 30$, initial point selection picks every 5 points along the regular grid, and refinement of point selection adds 20 points at each iteration and forbidden zone is of size 2. The size of each dataset is 7.42Mb with each point uses 4 bytes. For example, compressed Salinity dataset file size by 3D-ODETLAP is 60,037 bytes and thus has a compression ratio of 130:1.**

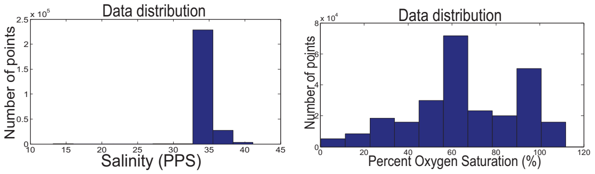


**Figure 8: Histogram of salinity and Percentage Oxygen Saturation datasets used in Table 4**

We can see from Table 4 that 3D-ODETLAP performs exceptionally well on Salinity data when compared with 3D-SPIHT. 3D-ODETLAP has a compression ratio 700% that of 3D-SPIHT's. This is mainly due to the intrinsic data distribution of different datasets. In Figure 8, almost 90% of salinity data scalar value fall into the range of 40 to 45 PPS. In contrast, Percentage Oxygen Saturation data scalar value distributes more evenly in its data range as shown in Figure 8. Because 3D-ODETLAP approximates the whole marine dataset by selecting certain points to "represent" their neighboring ones, a dataset of almost "flat", like salinity data, will require much fewer points to reconstruct from and thus will have better compression ratio while keeping errors the same with other methods. This feature of 3D-ODETLAP is especially valuable when we consider that lots of 3D marine, or even environmental datasets have a distribution with similar salinity in WOA 2005. Nevertheless, 3D-SPIHT is designed to be a general purpose compression method and can't take advantage of this special feature of marine datasets.

|  | 3D-ODETLAP |  |  |  | 3D-SPIHT |  |  |
|---|---|---|---|---|---|---|---|
| Variable | Mean | Max | Compressed | Compression | Mean | Max | Compression |
|  | Error(%) | Error(%) | File(bytes) | Ratio | Error(%) | Error(%) | Ratio |
| Salinity | 0.0532 | 0.2174 | 27,377 | 77:1 | 0.0530 | 0.4946 | 11:1 |
| Temper. | 0.4993 | 2.0673 | 21,508 | 98:1 | 0.50 | 17.91 | 135:1 |
| Dissolve. | 0.9993 | 4.4145 | 21,040 | 100:1 | 1.002 | 24.9965 | 71:1 |
| Apparen. | 0.9999 | 4.0170 | 24,775 | 85:1 | 0.9991 | 20.3609 | 81:1 |
| Percent. | 0.9985 | 4.5672 | 26,743 | 78:1 | 0.9969 | 20.3610 | 65:1 |
| Phospha. | 0.9993 | 4.5241 | 24,493 | 86:1 | 0.99784 | 15.6922 | 65:1 |
| Nitrate | 1.0242 | 4.6946 | 31,905 | 66:1 | 1.0006 | 18.5360 | 59:1 |
| Silicate | 0.9996 | 5.1437 | 23,076 | 91:1 | 1.0018 | 21.6457 | 81:1 |

**Table 4: This table demonstrates the comprehensive compression performance comparison between 3D-ODETLAP and 3D-SPIHT. Due to the data size limitations of 3D-SPIHT, we extract eight $128 \times 128 \times 32$ 3D data matrices and apply both methods on them. Mean Percentage Errors are kept approximately the same for comparison purposes. The size of each dataset is 2Mb with each point uses 4 bytes. For example, compressed Salinity dataset file size by 3D-ODETLAP is 27,377 bytes and thus has a compression ratio of 77:1.**

# 6. CONCLUSION AND FUTURE WORK

This paper demonstrates our recent progress in three dimensional oceanographic data compression and reconstruction that processes eight distinct datasets in WOA 2005 data which stores environmental information such as temperature, salinity, etc. Current popular 3D compression methods like JPEG 20000 and SPIHT extend their 2D compression to 3D. These methods may serve well for general purpose data compression, but for data that preserves features in a specific field, like WOA 2005 in marine study, they lack the flexibility to adjust themselves to the field. By contrast, the 3D-ODETLAP method has great adaptability to enable itself to find a good, though possibly not optimal, set of parameters to meet the needs for compressing data within a specific field.

Moreover, by applying wavelet-based 3D-SPIHT on the same oceanographic datasets we used testing 3D-ODETLAP, we find that 3D-ODETLAP can achieve much better compression ratio while intentionally keeping the mean percentage error on all respective datasets approximately the same. We can also obtain a much smaller maximum percentage error in our comparison.

One possible future extension is to speed up the linear solver for our PDE through parallelization. The emerging GPGPU[1] technology may be a candidate for potential research given the fact that NVIDIA provides CUDA[8]( Compute Unified Device Architecture) for data parallel computing and it's been popular, relatively easy and much cheaper than computing on huge clusters.

# 7. ACKNOWLEDGMENTS

This research was partially supported by NSF grant CMMI-0835762. We also thank Prof. Miriam Katz for her valuable advice on data preparation.

# 8. REFERENCES

[1] GPGPU(General-Purpose computation on GPUs). `http://www.gpgpu.org`, (retrieved 3/22/2010).

[2] Antonov, J. I., R. A. Locarnini, T. P. Boyer, A. V. Mishonov, and H. E. Garcia. World ocean atlas 2005, volume 2: Salinity. page 182, 2006.

[3] W. R. Franklin. Applications of geometry. In K. H. Rosen, editor, *Handbook of Discrete and Combinatorial Mathematics*, chapter 13.8, pages 867–888. CRC Press, 2000.

[4] Garcia, H. E., R. A. Locarnini, T. P. Boyer, and J. I. Antonov. World ocean atlas 2005, volume 3: Dissolved oxygen, apparent oxygen utilization, and oxygen saturation. page 342, 2006.

[5] Garcia, H. E., R. A. Locarnini, T. P. Boyer, and J. I. Antonov. World ocean atlas 2005, volume 4: Nutrients (phosphate, nitrate, silicate). page 396, 2006.

[6] J. Luo, X. Wang, C. Chen, and K. Parker. Volumetric medical image compression with three-dimensional wavelet transform and octave zerotree coding. In *Proceedings of SPIE*, volume 2727, page 579, 1996.

[7] S. Muraki. Volume data and wavelet transforms. *IEEE Comput. Graph. Appl.*, 13(4):50–56, 1993.

[8] Nvidia Corporation. CUDA (Compute Unified Device Architecture). `http://developer.nvidia.com/object/cuda.html`, (retrieved 3/22/2010).

[9] A. Said and W. Pearlman. A new, fast, and efficient image codec based on set partitioning in hierarchical trees. *IEEE Transactions on circuits and systems for video technology*, 6(3):243–250, 1996.

[10] G. Sewell. *The numerical solution of ordinary and partial differential equations*. Wiley-Interscience, 2005.

[11] J. Shapiro et al. Embedded image coding using zerotrees of wavelet coefficients. *IEEE Transactions on signal processing*, 41(12):3445–3462, 1993.

[12] J. Stookey, Z. Xie, B. Cutler, W. Franklin, D. Tracy, and M. Andrade. Parallel ODETLAP for terrain compression and reconstruction. In *16th ACM GIS symp.*, 2008.

[13] Z. Xie, W. R. Franklin, B. Cutler, M. Andrade, M. Inanc, and D. Tracy. Surface compression using over-determined Laplacian approximation. In *Proceedings of SPIE Vol. 6697 Advanced Signal Processing Algorithms, Architectures, and Implementations XVII*, 27 August 2007.

[14] Z. Xiong, X. Wu, S. Cheng, and J. Hua. Lossy-to-lossless compression of medical volumetric data using three-dimensional integer wavelet transforms. *IEEE Trans. Med. Imaging*, 22(3):459–470, 2003.

# PhD Showcase: Haptic-GIS: Exploring the Possibilities

PhD Student: Ricky Jacob
Department of Computer Science
NUI Maynooth
Co. Kildare. Ireland
rjacob@cs.nuim.ie

PhD Supervisor: Peter Mooney*
Environmental Research Center
Environmental Protection Agency
Clonskeagh,Dublin 14. Ireland
p.mooney@epa.ie

PhD Supervisor: Padraig Corcoran
Department of Computer Science
NUI Maynooth
Co. Kildare. Ireland
padraigc@cs.nuim.ie

PhD Supervisor: Adam C. Winstanley
Department of Computer Science
NUI Maynooth
Co. Kildare. Ireland
adamw@cs.nuim.ie

## ABSTRACT

Haptic technology, or haptics, is a tactile feedback technology that takes advantage of our sense of touch by applying forces, vibrations, and/or motions to the user through a device. Haptic enabled devices have recently gained much publicity in the computer games industry due to their ability to provide a more immersive experience. The use of haptic in the context of GIS and navigation assistance has not previously been considered. We present an overview of Haptic technologies and provide a commentary on how GIS and haptics may crossover and integrate. To demonstrate the potential of haptics for navigation assistance a simple case-study of using haptic feedback as a navigational assistant in pedestrian route planning software is presented. This case-study uses the OpenStreetMap(OSM) database and Cloudmade routing API.

## Categories and Subject Descriptors

H.5.2 [**User Interfaces**]: Haptic I/O; H.5.1 [**Multimedia Information Systems** ]: Artificial, augmented, and virtual realities

## General Terms

Human Factors

## Keywords

Haptic Feedback, Pedestrian Navigation, OpenStreetMap

## 1. INTRODUCTION

Haptic technology, or haptics, is a tactile feedback technology that takes advantage of our sense of touch by applying forces, vibrations, and/or motions to the user. Simple haptic devices are now commonly found on computer and video game controllers, in the form of force-feedback joysticks and steering wheels. In reality haptics have been employed mostly in a relatively unsophisticated manner ranging from rumbling video-game controllers and to vibration alerts on cellphones. Haptics has a broad and expansive range of potential applications: from handheld electronic devices to remotely operated robots. Yet outside of the haptic research and engineering community it is a virtually unknown concept [21]. The aim of any haptic system is that a user feels and interacts with a virtual model of their current environment. The potential of haptic technology has only recently started to receive the attention of the research community. This slow uptake can be attributed to two factors. Only recently has it been discovered that the human haptic sense is equally good at perceiving properties such, for example roughness, as the visual system [26]. Secondly the development of small scale wearable haptic enabled devices on which algorithms can be implemented are only starting to become available [31, 9].

In this paper we describe work in progress from PhD research into exploring the possibilities of integrating haptics into GIS. The last decade has seen GIS data and mapping move from the desktop application and desktop webbrowser to the dynamically located mobile device with high expectations regarding user interfaces, query response times, and visualisation. We explore the possibilities for extension or redevelopment of GIS user interfaces to include haptics. This raises the question of whether popular location-based services (LBS) applications such as: pedestrian navigation, map visualisation, city exploration and assisted navigation can benefit or be enhanced by haptics. A case-study application is described for haptic-assisted pedestrian navigation for visually impaired pedestrians. State-of-the-art mobile phones are haptic enabled. This makes developing haptic-enabled software easier as there is a well understood testing environment. The primary source of haptic feedback on mo-

*Also at: Dept of Comp Sci, NUI Maynooth, Ireland.

The SIGSPATIAL Special, Volume 2, Number 3, November 2010.

bile phones is vibration. Despite the falsely perceived limitations of vibrations many mobile devices have the ability to create complex pulsing vibration patterns, not just continuous vibrations. Pulsing techniques allow for a richer display of haptic effects and add another dimension to convey information [30].

The paper is organised as follows. In section 2 we give an overview of the literature on haptics where GIS or spatial data and interaction is explicitly stated or studied. In section 3 we describe the motivation for the development of a pedestrian navigation assistant application which uses haptic-feedback. In this section we describe a model for the haptic-interaction process using spatial data. The pedestrian navigation assistant application is described in detail in Section 4. The paper closes with Section 6 where we outline some discussion and concluding remarks.

## 2. LITERATURE REVIEW

The literature on haptic-related research is very broad as it usually integrates cross-disciplinary research including: engineering, user-interface design, telecommunications, robotics, and intelligent systems. While research into using the sense of touch to communicate has been ongoing for decades the development of haptic-based devices was hampered by the availability of only “bulky or very non-discrete instruments” [28] such as head-mounted devices and backpacks (wearable devices) [31, 9]. Advances in mobile phone technology and design has seen the state-of-the-art mobile phones integrated with multiple sensors [10]. Hoggan and Brewster [10] comment that this “makes it an easier task to develop simple but effective communication techniques on a device as handy as a mobile phone”. They describe the integration of actuators (a mechanical device for moving or controlling a mechanism or system) along with the mobile sensors for haptic feedback in a multi-modal interface. When combined with high resolution in-built cameras, on-board GPS, digital compasses, orientation sensors, and an accelerometer the mobile phone is equipped with the tools to support advanced haptic feedback interfaces. In Dai *et al* [7] the authors have demonstrated how a haptic-enabled mobile phone can be used as a drunk driving detection system. Their system works by reading values from a Google Android phone sensors (accelerometer and orientation sensor). Deviations in signals from the accelerometer and orientation sensor according to that of a normal driver are used to detect a potentially DWI (Driving While Intoxicated) driver. Spatial information analysis and handling requires the use of three cognitive spaces: haptic, pictorial, and transperceptual. Currently Geographical Information Systems interfaces do not yet integrate these three spaces in the same working environment [16]. Knowledge about designing haptic interfaces is still a young field and some key issues and challenges encountered have been outlined in the literature [15, 11].

We believe that the introduction of haptic technology into GIS would be of great benefit. In particular this work focuses on haptics in the context of navigation which is a central topic in GIS. Paneels and Roberts [21], who provide the first broad overview of the haptics field, describe some virtual map reading applications where electronic maps of US states are made tactile on screen - for example users could feel subtle bumps for county boundaries, large bumps for state boundaries, and a constant vibration on cities. Cervantes et al. [25] realized a tool for navigation in 3D virtual environments using the Novint Falcon haptic device. This technology is developed by the company Novint and has previously been exclusively used in the domain of computer game technologies. Cervantes et al. [25] showed that navigation could be greatly assisted in a virtual 3D environment through the use of haptics. This method was implemented on a large scale desktop device. The authors are unaware of any studies which evaluate the usefulness of haptics in a real environment using small scale devices. Ren *et al* [14] describe the development a multi-modal 3D GIS with a haptic interface which allows users to move through 3D representations of thematic maps. As variables such as temperature or humidity increase or decrease vibration feedback is given to alert the user. Newcomb and Harding [17] discuss the need for a multimodal interface in order to improve the human-computer interactions which is less in the traditional map interfaces due to information overload and emphasized on the need of audio, haptic or sonification. Hagedorn [8] investigates and assesses the functionality and viability of a novel multi-modal audio-haptic computer interface intended for non-visual geographic information retrieval. It has been shown that haptic clues allows individuals to build haptic mental models for better navigation skills in 3D virtual environments [25].

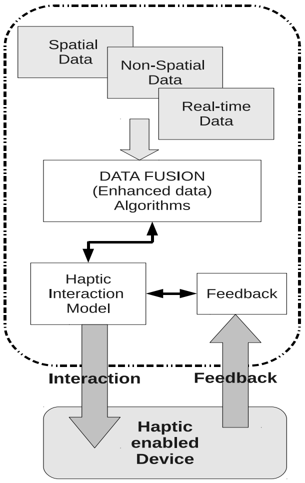


Figure 1: The haptic-interaction process using spatial data

## 3. HAPTIC-ASSISTED NAVIGATION

In this section we provide motivation and justification for the use of haptics in pedestrian navigation. Van Erp [29] argues that current popular navigation techniques are not “reasonable” or possible at all times. For example pedestrians use a “neck-down” approach and take their vision off their current environment. This has serious consequences like not paying attention to traffic in a busy street or not looking out for dangerous edges at the side narrow trail in the case of the hiker. Several authors [29, 24] discuss design of wearable haptic devices for assisting user for navigation and outlines about how the vibration alert can be used in conveying direction and deviation cues to the user in a way where it does not obstruct the users main activity. As stated above there are false perceptions about the limitations of vibrations as a feedback mode. Many mobile devices have the ability to create complex pulsing vibration patterns and not just continuous vibrations [30]. LoroDux [20] and HaptoRender [19] are two projects started by Lulu-Ann in 2009. The Haptorender project was to help map features for the visually impaired on OSM and then make it available for for users to navigate using a mobile device through the LoroDux project.

### 3.1 Motivation and Justification

One of the many difficulties experienced by those with vision impairment emerges from the communication barriers that prevent non-visual access to highly pictorial cartographic products and geographic reference sources. Without access to spatial information, the capacity for independent mobility, geographic learning, and communication concerning spatial concepts may be significantly reduced for anyone who cannot utilize traditional visual maps [13]. Crossing the street or the road, as a pedestrian, is a risk-laden task. This task becomes even more difficult for those pedestrians with visual impairments. There are a number of well-documented problems for pedestrians at the typical signalized crossings that are provided in most countries. These problems are: the fact that pedestrians are supposed to register their demand manually by pushing a button, but frequently do not do so; inadequate crossing time duration for slower pedestrians; insufficient responsiveness in the signals, so that the pedestrian stage is only available at a certain point in the signal cycle, regardless of demand [3]. In poor weather, pedestrians were more likely to walk against a "Flashing Dont Walk" or steady "Dont Walk" signal. The proportion of pedestrians obeying the traffic signals at this two-stage crossing was only 13% and it dropped to 3% when it was cold and snowing. The alarming low compliance rate at this crossing questions the effectiveness of staged crossings with pedestrian refuges at signalized intersections, especially in inclement weather [32].

In this paper our target user group are visually-impaired pedestrians. Several studies [27, 4] show that the usage of mobile devices for navigation amongst the visually impaired is high. While the cane and guide-dog are the oldest navigational assistance methods for the visually impaired the "most effective available mobility aid for visually impaired people is using a mobile phone device as it provides a fully supportive and stress free guidance" [4]. Visually-impaired pedestrians cannot use vision-based feedback systems. Studies also show that visually impaired pedestrians are less like to take the risks of "crossing while red" than other pedestrian road users take. A haptic-feedback approach is considered in this paper where feedback can take the form of audible commentary from the mobile device or vibration of the device. Audible assistance is available at many traffic signals where traffic signals emit a series of sounds to assist blind pedestrians and have been in operation in many countries for several decades now[23]. In Ivanchenko *et al* [12] the authors describe a novel camera phone-based system for helping visually impaired pedestrians to find crosswalks and align themselves properly before they attempt to cross. This application searches an image for crosswalk zebra stripes. Amemiya and Sugiyama [1] build a mechanism for creating a pseudo-attraction force designed for provide a haptic direction indicator. The device was used to assist visually impaired users to navigate on a pre-defined walking route. However, traffic signal detection was not considered.

### 3.2 The haptic-interaction model

In Figure 1 we extend the haptic visualisation process model of Paneels and Roberts [21] to describe a haptic-interaction model which relies heavily on spatial data to drive decision making. In our case-study application in section 4 we use OpenStreetMap as our source of spatial data. As described in Figure 1 this spatial data can be fused with other sources of data including non-spatial data and real-time data such as weather forecast reports, pollen counts, environmental information. The Data Fusion component provides the spatial decision making for the haptic model. In our case this the data fusion component is made up of the shortest path routing algorithms and the parsing and understanding of the computed route. The haptic interaction model component is where haptic actions, signals, or interactions are produced based on output from the data fusion component. The interactions can take any of the haptic forms mentioned above including vibration, pulsing, sound, text, or graphics. Using the software API on the haptic enabled device hardware the interaction is delivered to the device. This instructs the user to take some action. Feedback is then provided by the user. This can take many forms depending on the application. These forms include: movement of the device, pressing or touching the device, voice commands, or selecting a user-interface component from the screen.

## 4. APPLICATION OVERVIEW

In this section we give an overview of our application and will outline the various components of the application including the data, routing algorithms and mobile phone sensors.

- **Spatial Data Source: OpenStreetMap** OSM [18] is a very popular source of Volunteered Geographic Information (VGI). The OSM data model is very flexible and we have mapped our university town to include a great deal of spatial richness. To improve the web-based query performance of our application we maintain a local copy of the OSM database for Ireland (in PostgreSQL PostGIS - as described by Ciepluch *et al* [5]). We have extended the OSM tag/metadata ontology to include height information about traffic lights and lamp-posts. Tactile pavement areas and other physical features on paths and streets which assist visually impaired pedestrians are mapped, tagged and included in the OSM database.

- **Routing: Cloudmade routing API** The Cloudmade Routing API[6] uses OSM spatial data to provide powerful web routing services which delivers turn-by-turn directions. For this project we needed access to each of the nodes in the navigation path between the source and destinations points of a route. Using standard HTTP GET methods the coordinates of the user's start and destination points are sent to the Cloudmade Routing API. In this API shortest paths can be generated for: pedestrians, automobiles, and bicycles. The Cloudmade API is dependent upon a very rich OSM representation of the current location of the user. Provided geographical features in the OSM database are correctly annotated with "tags" the API can find realistic and accurate routings. The routing algorithm (most probably Dijkstra's algorithm) in the API can route pedestrians also through indoor paths provided these features are in the OSM database and correctly annotated to indicate a pedestrian thoroughfare.

- **Sensors: GPS and Orientation:** The Android phone

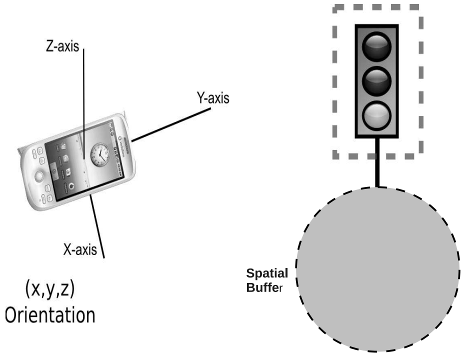


Figure 2: Extrinsic camera parameters in traffic light detection model

software provides API access to the sensors on the phone. We can obtain the user's current location in (latitude, longitude). The orientation of the phone is also accessible from the orientation sensor. This returns the angles for yaw, pitch, and roll. Here yaw is the rotation around the z-axis, pitch and roll represents the rotation around the x and y-axis respectively. The method `addProximityAlert()` is used so that we can alert the user when he has reached within a particular radius of a point. We use this so that we can alert the user with a different frequency of vibration that he has reached a way point or near the pedestrian crossing. These parameters allow us to calculate the extrinsic camera parameters. This is described in Figure 2. The spatial buffer is used to alert the user that they are in the correct location and now must hold the camera at the correct orientation so a picture of the traffic light may be taken. The positions and heights of all traffic lights in the town are stored in OSM.

- **Vibration Alarm:** The vibration alarm in the phone is the most important component of navigation in our work. The vibration alarm class in the Android is used in our work to provide the haptic feedback to the user. We can specify the duration of the vibration by accessing the API method `vibrate(long 2000.0)` to make the phone vibrate for 2 seconds. Vibration to a given pattern is possible by using `vibrate(long[] pattern, int repeat)` where "pattern" is an array storing times at which to turn on and off the vibration alarm and "repeat" holds the index for the pattern where the repeat begins and ends. Different patterns of vibration are used to provide haptic feedback to the user for: (1) path following ; (2) signalling a change of direction in the path; (3) alerting the user that they have reached an area which includes a tactile pavement near a pedestrian crossing; (4) to alert the user that he is pointing towards the traffic light post and can take a picture; and (5) to give feedback after the template matching of the picture of the traffic light to indicate to the user if it is "green to go" or "red to stop".

- **Camera** The camera in the phone is used to capture an image of the traffic light. We developed a simple template matching algorithm which matches the captured image with a set of template images of traffic lights [2]. If template matching returns a high correlation with a "red" signal template the user is given a pattern of vibration to indicate *stop*. On the other hand if template matching returns a high correlation with a "green" signal template the user is given a pattern of vibration to indicate they may proceed to cross the street. A set of template images are used to ensure that all styles of traffic light signal structures found in this environment are available for matching. An example of the template matching is shown in Figure 3

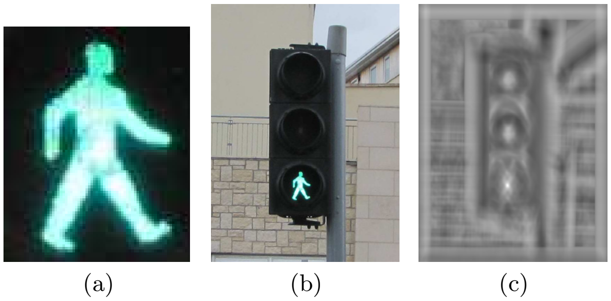


Figure 3: The template in (a) is searched for in the image (b). A visualisation of the corresponding correlation values are shown in (c). A high correlation value is found in the lower center of (c) indicating the presence of the template in question.

## 5. EXPERIMENTAL ANALYSIS

Experiments were carried out in fairly open areas and around low rise buildings. Two visually impaired participants assisted us in our experiments. A route within the university area was generated and traversed by the users. We selected this route as it consisted of a mix of both walkways on campus and a foot path beside a public road. Traffic light signal locations have tactile pavements near the crossing location. To check the accuracy of GPS we recorded the lat/long values at various points in the path by collecting them using the phone and compared it, by map matching, with the lat/long values of the OSM database. Less than 3% deviation was recorded between the values collected from the GPS and matched to the path or walkway objects in the OSM database. We tested the template matching algorithm by capturing different images (photographs of the traffic signals) at different times of the day. By setting a very high matching threshold value (0.95) for correlation we successfully filtered out the majority of False Positives(FP) and False Negatives(FN). In Table 1 we present the results of testing of the template matching component to automatically detect traffic light colors: (R) Red, (Or) Orange, (G) Green and (NL) No Light. There were 50 samples taken for each colour. If the returned correlation value of the template matching algorithm is not $\geq .95$ we return the decision as "No Light". This helps to eliminate cases of incorrect classification of a Red light and prevent the situation where the pedestrian is advised to proceed with crossing under a Red light. From the 50 test samples taken for each traffic light state the system responded correctly in at least 70% of cases. The remaining cases were usually an "NL" response which is provided for the safety of the pedestrian. In Table 2 we present the results of the actions of the user based on a particular suggestion from our system. In all cases the user was aware that it was a trial situation. From the 20 times the system directed the user to "wait", the users did

Table 1: The responses of the Traffic-light component

| | | **Actual** | | | |
|---|---|---|---|---|---|
| | | **R** | **Or** | **G** | **NL** |
| **Predicted** | **R** | 36 | 0 | 0 | 0 |
| | **Or** | 0 | 38 | 0 | 0 |
| | **G** | 0 | 0 | 42 | 0 |
| | **NL** | 14 | 12 | 8 | 50 |

Table 2: User response to the Camera component

| | | **System** | |
|---|---|---|---|
| | | **Cross** | **Wait** |
| **Action** | **Cross** | 11 | 0 |
| | **Wait** | 9 | 20 |

so. However in the case of the system directing the user to "cross", out of 20 trails the the users actually waited on 9 occasions and did not cross. When directed to cross the user did supplement the advice from our system by listening and attempting to gauge the reactions of other pedestrians. From the 11 times where the user crossed based on directions from the system, 3 of these were occasions where the audible sound signal at the crossing was clearly audible with no other distracting noise of vehicles. We feel that the visually impaired users are incorporating environmental awareness skills of their own with the advice from the system to make the decision about crossing.

## 6. CONCLUSIONS

In this paper we have presented a case study describing how haptic feedback in mobile phones can be used as a means for localization and navigation. Scalability of the application to work in new areas depends on the availability of OSM data in those regions. Other research has been found that blind people use both hands while using a cellphone as they hold the phone in one hand and use the fingers of the other hand to explore the phone [22]. One key advantage of our application is that the visually impaired pedestrian is not burdened with carrying another device. Our test participants remarked that navigation using a cane in one hand and the mobile in the other is a likely combination provided effective communication can be provided using different vibration patterns on the phone. The image processing component used in this system will remain as a "pluggable" component which by default is turned off. Our participants commented that they would like to maintain their independence at traffic lights by crossing themselves. In order to provide the user with instantaneous and current information the image processing algorithms must be of low computational complexity for successful implemention on the mobile device for this application. We found from the experimental trials that the participants sometimes ignored the indications from our application and went with their "instinct" while taking a decision making to cross. Using other environmental awareness skills (sound, detection of movement) our participants either waited for a longer duration or "went with the flow" of other pedestrians crossing at the same location.

When developing a non-visual method for exploring geographic data we feel that richer and more meaningful sensory feedback can be obtained from a system that enables intrinsically spatial tactile user interactions. Unfortunately, non-visual map exploration is a mentally demanding activity. In instances where a blind user is exploring a map on a conventional personal computer or mobile device, a standard mouse, pointing device, or the user's finger are the input devices most often used to both explore and manipulate the map scene. While this case-study used an example of the pedestrian navigation for visually impaired users we shall be working to extend this to a multimodal location and navigation system. The use of the full range of haptics cues will be considered including audio feedback and image assisted navigation for fully sighted users. More testing will be carried out in dense urban environments. Given the distractions and noise of busy urban centers how effective will vibration alarms be? To develop a Haptic-GIS the GIS community must think beyond the "slippy-maps" and AJAX-driven interaction and feedback models of web-based GIS and location-based services on mobile devices. The recent changes made to the Android API classes for location and orientation of the mobile phone will be incorporated into our work.

## Acknowledgments

Research presented in this paper was funded by a Strategic Research Cluster grant (07/SRC/I1168) by Science Foundation Ireland under the National Development Plan. The authors gratefully acknowledge this support. Peter Mooney is funded by the Irish Environmental Protection Agency STRIVE programme (grant 2008-FS-DM-14-S4). Padraig Corcoran gratefully acknowledges the support of the Department of Computer Science NUIM.

## 7. REFERENCES

[1] T. Amemiya and H. Sugiyama. Haptic handheld wayfinder with pseudo-attraction force for pedestrians with visual impairments. In *Assets '09: Proceedings of the 11th international ACM SIGACCESS conference on Computers and accessibility*, pages 107–114, New York, NY, USA, 2009. ACM.

[2] R. Brunelli. *Template Matching Techniques in Computer Vision: Theory and Practice*. Wiley Publishing, 2009.

[3] O. M. J. Carsten, D. J. Sherborne, and J. A. Rothengatter. Intelligent traffic signals for pedestrians: evaluation of trials in three countries. *Transportation Research Part C: Emerging Technologies*, 6(4):213 – 229, 1998.

[4] F. Cecelja, V. Garaj, Z. Hunaiti, and W. Balachandran. A navigation system for visually impaired. In *Instrumentation and Measurement Technology Conference, 2006. IMTC 2006. Proceedings of the IEEE*, pages 1690 –1693, 24-27 2006.

[5] B. Ciepluch, P. Mooney, R. Jacob, and A. C. Winstanley. Using openstreetmap to deliver location-based environmental information in ireland. *SIGSPATIAL Special*, 1:17–22, November 2009.

[6] CloudMade. CloudMade Web Maps Lite API. `http://developers.cloudmade.com/projects/show/web-maps-lite` - last accessed, September 2010, 2010.

[7] J. Dai, J. Teng, X. Bai, Z. Shen, and D. Xuan. Mobile phone based drunk driving detection. In *4th International ICST Conference on Pervasive Computing Technologies for Healthcare (Pervasive Health) March(2010). Proceeings of the*, 2010.

[8] Douglas Hagedorn. Exploring new directions in non-visual cartography: an overview of the functionally separated multi-modal map rendering system. In *ICC*, 2009.

[9] B. F. V. erp, C. V. Veen, and C. Jansen. Waypoint tilt angle and providing real time biofeedback has been navigation with a vibrotactile waist belt. *ACM Transactions on Applied Perception (TAP)*, 2, Issue 2:106 âĂŞ 117, 2005.

[10] S. A. Eve Hoggan and S. A. Brewster. Mobile multi-actuator tactile displays. In *Lecture Notes in Computer Science: Haptic and Audio Interaction Design*, volume Volume 4813/2007, pages 22–33. Springer Berlin / Heidelberg, October 2007.

[11] V. Hayward and K. Maclean. Do it yourself haptics: part i. *Robotics Automation Magazine, IEEE*, 14(4):88 –104, dec. 2007.

[12] V. Ivanchenko, J. Coughlan, and H. Shen. Crosswatch - a camera phone system for orienting imparied pedestrians at traffic intersections. *Lect Notes Comput Sci*, 5105(4):1122–1128, 2008.

[13] J. R. Marston, J. M. Loomis, R. L. Klatzky, R. G. Golledge, and E. L. Smith. Evaluation of spatial displays for navigation without sight. *ACM Transactions on Applied Perception*, 3(2):110–124, Apr. 2006.

[14] Ming Ren, Kenji Hikichi, and and Kaoru Sezaki. Multi-Modal 3D Geographical Information Systems with Haptic Interface. In *proceedings of Euro Haptics*, 2004.

[15] C. Moussette and D. Fallman. Designing for Touch: Creating and Building Meaningful Haptic Interfaces. In *Proceedings of IASDR 2009, International Association of Societies of Design Research*, Seoul, Korea, 2009.

[16] N. Neves, J. P. Silva, P. GonÃğalves, J. Muchaxo, J. M. Silva, and A. CÃćmara. Cognitive spaces and metaphors: A solution for interacting with spatial data. *Computers & Geosciences*, 23(4):483 – 488, 1997. Exploratory Cartograpic Visualisation.

[17] M. Newcomb and C. Harding. A multi-modal interface for road planning tasks using vision, haptics and sound. In *International Symposium on Visual Computing (ISVC'06)*, pages 417–426, 2006.

[18] OSM. Open Street Map. `http://www.openstreetmap.org/` - last accessed, September 2010, 2010.

[19] OSM-Wiki. Hapto Render. `http://wiki.openstreetmap.org/wiki/HaptoRender` - last accessed, September 2010, 2010.

[20] OSM-Wiki. LoroDux. `http://wiki.openstreetmap.org/wiki/LoroDux` - last accessed, September 2010, 2010.

[21] S. Paneels and J. C. Roberts. Review of designs for haptic data visualization. *Haptics, IEEE Transactions on*, 3(2):119 –137, april-june 2010.

[22] O. Plos and S. Buisine. Universal design for mobile phones: a case study. In *Conference on Human Factors in Computing Systems ACM SIGCHI 2006 extended abstracts on Human factors in computing systems(CHI 2006)*, 2006.

[23] T. Poulsen. Acoustic traffic signal for blind pedestrians. *Applied Acoustics*, 15(5):363 – 376, 1982.

[24] L. Ran, S. Helal, and S. Moore. Drishti: An integrated indoor/outdoor blind navigation system and service. In *in Proc. of the Second IEEE Annual Conference on Pervasive Computing and Communications*, pages 23–30, 2004.

[25] G. Sepulveda-Cervantes, V. Parra-Vega, and O. Dominguez-Ramirez. Haptic cues for effective learning in 3d maze navigation. In *Haptic Audio visual Environments and Games, 2008. HAVE 2008. IEEE International Workshop on*, pages 93 –98, 18-19 2008.

[26] W. M. B. Tiest and A. M. Kappers. Haptic and visual perception of roughness. *Acta Psychologica*, 124(2):177 – 189, 2007.

[27] M. Uddin and T. Shioyama. Detection of pedestrian crossing and measurement of crossing length - an image-based navigational aid for blind people. In *Intelligent Transportation Systems, 2005. Proceedings. 2005 IEEE*, pages 331 – 336, 13-15 2005.

[28] J. B. F. van Erp. Tactile navigation display. In *Proceedings of the First International Workshop on Haptic Human-Computer Interaction*, 2000.

[29] Wilko Heuten, Niels Henze, Susanne Boll, and Martin Pielot. Tactile wayfinder: a non-visual support system for wayfinding. In *Proceedings of the 5th Nordic conference on Human-computer interaction: building bridges*, Lund, Sweden, 2008.

[30] H.-Y. Yao, D. Grant, and M. Cruz. Perceived vibration strength in mobile devices: The effect of weight and frequency. *Haptics, IEEE Transactions on*, 3(1):56 –62, jan.-march 2010.

[31] J. S. Zelek and M. Holbein. Patent number: 20080120029 wearable tactile navigation system. Details available online at `http://www.freepatentsonline.com/y2008/0120029.html` - last checked July 1st 2010, May 2008.

[32] R. Zhou and W. J. Horrey. Predicting adolescent pedestrians' behavioral intentions to follow the masses in risky crossing situations. *Transportation Research Part F: Traffic Psychology and Behaviour*, In Press, Corrected Proof:–, 2010.

# Ph.D. Showcase: Slope Preserving Lossy Terrain Compression

PhD Student:
Zhongyi Xie
Rensselaer Polytechnic Institute
110, Eighth Street
Troy, NY, USA
xiez2@cs.rpi.edu

PhD Supervisor:
W. Randolph Franklin
Rensselaer Polytechnic Institute
110, Eighth Street
Troy, NY, USA
wrf@ecse.rpi.edu

PhD Student:
Daniel M. Tracy
Rensselaer Polytechnic Institute
110, Eighth Street
Troy, NY, USA
daniel.m.tracy@gmail.com

## ABSTRACT

Accurate terrain representation with appropriate preservation of important terrain characteristics, especially slope steepness, is becoming more crucial and fundamental as the geographical models are becoming more complex. Based on our earlier success with Overdetermined Laplacian Partial Differential Equations (ODETLAP), which allows for compact yet accurate compression of the Digital Elevation Model (DEM), we propose a new terrain compression technique that focuses on improving slope accuracy in compression of high resolution terrain data. With high slope accuracy and a high compression ratio, this technique will help geographical applications that require a high precision in slope yet also have strict constraints on data size. Our proposed technique has the following contribution: we modify the ODETLAP system by adding slope equations for some key points picked automatically so that we can compress the elevation without explicitly storing slope values. By adding these slope equations, we can perturb the elevation in such a way that when slope is computed from the reconstructed surface, they are accurate. Note we are not storing the slope explicitly, instead we only store the elevation difference at a few locations. Since the ultimate goal is to have a compact terrain representation, encoding is also an integral part of this research. We have used Run Length Encoding (RLE) and linear prediction in the past, which gave us substantial file size reduction. In addition to that, we also propose a Minimum Spanning Tree based encoding scheme that takes advantage of the spatial correlation between selected points. On a typical test, our technique is able to achieve a 1:10 compression at the cost of 4.23 degree of RMS slope error and 3.30 meters of RMS elevation error.

## Categories and Subject Descriptors

I.3.5 [**Computing Methodologies**]: Computer Graphics Computational Geometry and Object Modelling

## General Terms

Algorithms, Experimentation, Theory

## Keywords

GIS, PDE solver, terrain modeling, terrain elevation data set compression

## 1. INTRODUCTION

The availability of high resolution Digital Elevation Model (DEM) data including LIDAR, stereo photogrammetry and Doppler radar has greatly assisted the development of GIS related research. High resolution DEM data has made possible terrain analysis in geomorphology, water flow modeling, and the great success of GPS (Global Positioning Systems). However, accurate representation of terrain as well as compact compression and transmission of terrain data still remain problems, especially for topographical derivatives of the terrain. The primary derived topographical parameters associated with DEMs are slope and aspect [9], which are fundamental to several terrain related applications including hydrology, visibility and environmental sensing [6]. However, while slope is technically the derivative of the elevation, lossy compression of the elevation could amplify errors so that when recomputing the slope from the lossily compressed terrain, it could contain far more error than the elevation. Perhaps because of the incorrect assumption that accurate elevations imply accurate slopes, there appears to be no prior art on this topic.

This paper presents a lossy compression technique that preserves elevation and slope with high accuracy and also produces a good compression ratio, which facilitates all kinds of DEM data handling, including high speed data transmission. The work is based on the ODETLAP framework, which is very effective in reconstructing smooth terrain with high accuracy from lossily compressed elevations with certain properties [18]. Since the previous versions of ODETLAP compress elevation well, our focus will be on improving the effectiveness of preserving the slope accuracy throughout the compressing and decompressing process. In addition to getting higher accuracy, improving compression ratio is also among our goals. Due to the cost of solving large linear system, this technique has cubic complexity in terms of the input DEM size. However, we can still process DEM of $16,000 \times 16,000$ using parallelism in about 30 minutes [11].

The SIGSPATIAL Special, Volume 2, Number 3, November 2010.

# 2. PRIOR ART

How shall the elevations of the unknown positions be determined? The first law of geography is a well accepted principle. It states that everything is related to everything else, but near things are more related than distant things [14].

Some well known methods that directly apply this principle include proximity polygon, inverse distance weighting and kriging. All of them essentially set the elevation of an unknown position (i,j), $z_{i,j}$, to be a weighted average of known elevations $h_l$ where $l = 1, 2, \ldots, k$:

$$z_{i,j} = \sum_{l=0}^{k} w_{(i,j),l} h_l \tag{1}$$

*Proximity polygon* (or *Voronoi polygon* or *nearest point*) sets $z_{i,j}$ to its nearest known neighbor, which means $w_{(i,j),l} = 1$ for the nearest known position but $w_{(i,j),l} = 0$ for all the others [13].

*Inverse distance weighting*, as the name suggests, sets $w_{(i,j),l}$ to an inverse power of distance between $(i, j)$ and the known position $l$, usually square [10]. *Kriging* is a geostatistical approach, in which all control point data are involved in finding optimal values of the general weighting function $w(s)$ for a known point distant $s$ from the unknown position. The main assumption here is that the covariance between two elevations depends solely on the distance between the positions [7].

Another popular approach involves fitting splines. In this case, first-order and even second-order continuity are explicitly enforced, thereby having an additional advantage of ensuring the slope of the surface is smooth.

*Compressed sensing* is a technique that has existed since 1970's, recently rediscovered by David Donoho [1] where it is possible to reconstruct a sparse signal from a set of samples whose size is so small that the Nyquist-Shannon sampling theorem would tell the opposite. The difficulty with the classical methods is that minimizing the $L_2$ norm is feasible but often leads to poor results while minimizing the $L_0$ norm makes the problem NP-hard. Donoho proved that the $L_1$ norm is equivalent to the $L_0$ norm, which makes it possible to solve the problem using linear programming.

However, in most cases such interpolating the known points is not necessary because of the measurement imprecision. *Approximation*, which allows relaxation from the measured values, allows better overall prediction results and much smoother surfaces. De-correlating elevation fluctuations from the density of known positions is vital because the way data are collected does not imply anything about the roughness of the terrain.

*Trend surface analysis* is a representative technique used for surface approximation. It involves specifying a general form of a mathematical function at the beginning. This is the trend which is expected to represent a large scale systematic change that extends from one map edge to the other. Then we fit the function with the sample data aiming to minimize least squares, a process also known as regression. A review of the technique can be found from Wren [17].

Numerous methods have been proposed to compute slope from a DEM. The slope can be calculated in one of the following ways: either by using trigonometry or by using differential geometry [16]. We pick the Zevenbergen-Thorne method [19], a differential geometry based method, which offers good results. It first computes the gradient in the horizontal and vertical directions separately by taking the difference of neighbors in that direction. Then we take cross product of the two to get the slope normal.

Compression of raster DEM data is supported by most GIS software packages. GRASS simply uses run length encoding, which is lossless, meaning it doesn't compress that much. ArcGIS offers several options: LZ77(lossless), JPEG(lossy) and JPEG2000(lossy).

# 3. BASIC ODETLAP

## 3.1 ODETLAP Framework

ODETLAP [2, 4, 11, 18] operating on an elevation grid where only a few points are known, computes a surface over the entire grid. ODETLAP, for Overdetermined Laplacian Partial Differential Equation, as an extension of a Laplacian PDE

$$\frac{\delta^2 z}{\delta x^2} + \frac{\delta^2 z}{\delta y^2} = 0 \tag{2}$$

to an overdetermined system of equations. Each point in the elevation matrix induces an equation setting it to the average of its neighbors.

$$z_{i,j} = (z_{i-1,j} + z_{i+1,j} + z_{i,j-1} + z_{i,j+1})/4 \tag{3}$$

(For border points, only the two or three neighbors that exist are used. This biasses the slope towards zero at the border, but that is insignificant in this context.) In addition to that, each point whose elevation is known induces another equation, an *exact equation*.

$$z_{i,j} = h_{i,j} \tag{4}$$

where $h_{i,j}$ is the given elevation at that point. For each unknown point, we create only an averaging equation. Therefore, if $k$ of the $n^2$ points have known elevations, there are $n^2 + k$ equations for the $n^2$ unknown variables. Even for a point of known elevation, we consider its actual elevation to be unknown and solve for it. Obviously, the resulting elevation will be close to the reported elevation, but will not be exactly it, because of the influence of nearby known points. Because the known elevations are only approximate, that is appropriate.

When there are more equations than unknowns, and the system of equations is inconsistent, we optimize for a least squares solution. Expressing the inconsistent system as $Ax = b$, the error is $e = Ax - b$, and we want to find an $x$ to minimize $||e||$. Mathematically, that occurs at $x = (A^t A)^{-1} A^t b$. However, that formula is never used because it is slower and uses more space, especially for sparse systems such as we have here, than any of various Matlab algorithms [12].

In this least squares solution, multiplying both sides of an equation by a constant will scale that equation's error and change its relative importance in the solution. We multiply all our averaging equations by a parameter $R$, which trades off accuracy and smoothness. A larger $R$ causes the solution to be smoother, but to be farther from the known points. The basic ODETLAP has only this one parameter, which gives it great conceptual simplicity. Figure 1 shows the impact of $R$ on the resultant surface.

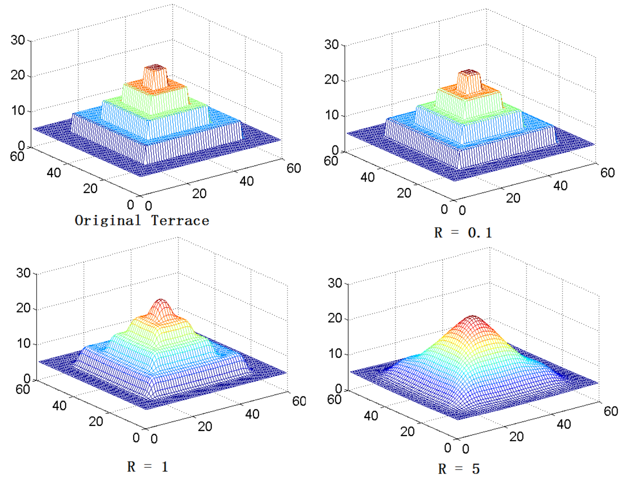


**Figure 1: ODETLAP approximation of nested squares**

ODETLAP has the following advantages. First, it can handle continuous contour lines of elevation, while allowing them to have gaps. Second, in addition to broken contour lines, isolated points can also be handled well, it is even possible to produce a surface that infers mountain tops inside innermost contours. Third, ODETLAP could enforce slope continuity across contours and generally shows no visible indication of input contours, i.e. no generated terraces. Figure 1 shows a surface interpolated over four nested square contours, an example chosen to be particularly difficult to interpolate. With a mean error of 4.7%, there is no evidence of terracing on the silhouette, which is the place where it would be most visible. Also the local max is nicely inferred.

The cost is that ODETLAP uses more resources, both memory and time. Its running time depends first on the cube of the number of rows in the DEM, and second on the number of known points. For an $n \times n$ dataset with $k$ known points, the running time is of $\Theta(n^3 + k)$. Solving a $600 \times 600$ dataset with 23 630 known points takes about 13 CPU-minutes and 1.1GB of writable memory on a 2.8GHz Lenovo Thinkpad W700 with 4GB of main memory running Ubuntu 9.04 Linux and 64 bit Matlab R2009a. For larger datasets, we can partition the problem, solve the pieces, and smoothly merge the solution pieces [11].

## 3.2 ODETLAP based DEM Compression

ODETLAP can be used as a lossy compression technique for the raster DEM data because once we obtained a point set $S$, we can generate an approximation of the original raster DEM using ODETLAP. With a carefully chosen $S$, we can produce a terrain with higher elevation and slope accuracy.

The ODETLAP compression algorithm's outline is shown in Figure 2 and the pseudo code is given in Algorithm 1.

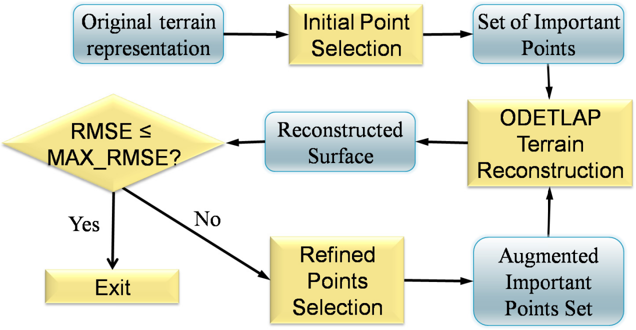


**Figure 2: ODETLAP DEM compression algorithm**

Starting with the original DEM, there are two point selection phases: Firstly, the initial point set S is built by selecting a regular grid over the original DEM and an initial approximation is computed using the equations 3 and 4. Given the reconstructed surface, a stopping condition based on an error measure is used. In practice, we have used the root-mean-square error(RMSE) as the stopping condition.

$$\text{RMSE} = \sqrt{\frac{\Sigma(Z_i - Z_t)^2}{N}}$$

$Z_i$ : Computed DEM elevation at a given point
$Z_t$ : True DEM elevation at a given point
$N$ : Total Number of points

If this condition is not satisfied, the second step is executed. In this step, k $\geq$ 1 points from the original terrain are selected by the method described in 3.3 and they are inserted in the existing point set S; this extended set is used by ODETLAP to compute a more refined approximation. As the algorithm proceeds, the total size of point set S increases and the overall error converges.

**Algorithm 1** ODETLAP DEM Compression

```
Require: $T$ = Original Terrain
Ensure: $S$ = Output Point Set,$RS$ = Restored Surface
  $S$ = InitSelection($T$)
  $RS$ = ODETLAP($S$)
  while RMSE($RS, T$) > $MAX\_RMSE$ do
    $S = S \cup$ Refine($T, RS$)
    $RS$ = ODETLAP($S$)
  end while
  return $S$
```

## 3.3 Refined point selection - Greedy algorithm

After the initial point set is obtained, ODETLAP is used to reconstruct the elevation matrix. This matrix has high error with respect to the original terrain, mostly due to the limited size of the initial point set. As shown in Figure 2, refined points selection is applied and a set of additional points is chosen and added to the existing points set S to form the augmented points set. The way we choose new points is greedy: we find a set of points with the greatest absolute vertical error. The size of the set in our experiments is intentionally kept small (10% or smaller) so that for a given total number of points, more iterations could be used to reduce the error as much as possible. This is actually a trade off between accuracy and computation time. The augmented set is then given to ODETLAP to reconstruct a more refined approximation. The newly obtained approximation is again examined with respect to the original terrain against our stopping condition, which is either relative RMSE: compute the RMSE of the approximation and check if its ratio against the RMSE of the first approximation is less than a predefined threshold or absolute RMSE: define a value for the maximum acceptable RMSE.

## 3.4 ODETLAP Slope Compression

Dan Tracy [3, 15] extended the ODETLAP by adding two equations

$$z_{i+1,j} - z_{i-1,j} = h_{i+1,j} - h_{i-1,j} \tag{5}$$

$$z_{i,j+1} - z_{i,j-1} = h_{i,j+1} - h_{i,j-1} \tag{6}$$

where the meanings of $z$ and $h$ remain consistent with that of equation 3 and 4. Slope equations are structured in this manner to match the way these quantities are used in the Zevenbergen-Thorne slope method. Similar to the parameter $R$ for regular ODETLAP, we use introduce a new parameter $RS$ to control the weight of slope equations. So now the overdetermined system contains these equations:

- $R \times [x_{i,j} = (x_{i+1,j} + x_{i-1,j} + x_{i,j+1} + x_{i,j-1})/4]$
- $x_{i,j} = h_{i,j}$
- $RS \times [z_{i+1,j} - z_{i-1,j} = h_{i+1,j} - h_{i-1,j}]$
- $RS \times [z_{i,j+1} - z_{i,j-1} = h_{i,j+1} - h_{i,j-1}]$

The ODETLAP slope compression algorithm is then a slight modification of Algorithm 1. Like before, we start with an initial guess. In each iteration, a certain number of points with greatest slope errors are picked nonrepetitively. Here we collect the points together with the gradient in $x$ and $y$ directions (called $\Delta_x$ and $\Delta_y$ separately). The terrain is then approximated using the new slope ODETLAP equations and the whole process is repeated until the desired accuracy is reached.

# 4. ENCODING

The whole ODETLAP DEM compression algorithm mainly does one thing: selecting points that would best represent the terrain in such a sense that when given to ODETLAP, would produce most accurate terrain. However, selecting points is not the end for compression, a more compact representation would be more economical and useful in practical applications. To improve the compression ratio, the points are compressed using the following strategies: the $(x, y, z)$ coordinates are split into $(x, y)$ pairs and $z$ alone. The former are compressed using an adaptive run length encoding method, described below in section 4.1. The z sequence is compressed using linear prediction and then by bzip2.

## 4.1 Run Length Encoding

The coordinates $(x, y)$ are different from $z$ because they distribute uniformly within the matrix dimension's values, for example from 1 through 400, while $z$ values follow some geographical distribution yet to be discovered. The run length encoding is a simple lossless compression technique which, instead of storing the actual values in the sequence, stores the value and the count of sequence containing the same data value. A run is just a consecutive sequence that contains the same data value in each element. Since the $(x, y)$ values correspond to positions in a matrix, we need to store only a binary bitmap where each location indicating whether the corresponding point exists in S or not. There is no need to store the original $(x, y)$ pairs. Thus, given a binary matrix of size $N \times N$, the method is the following: For each run length $L$, test if $L < 254$, then use one byte for it; if $254 \leq L < 510$, then use FE as a marker byte and use a second byte for $L - 254$; if $510 \leq L < 766$, then use FF as a marker byte and use a second byte for $L - 510$; lastly, if $L \geq 766$, then use FFFF as a two byte marker and use next two bytes for L.

For our test cases, we made some histograms of the run length which show that most runs are below 512, that means for most runs we need only 1 byte to store it. Here we assume all runs are shorter than 65535, which is a reasonable value for terrains of 400 × 400 resolution and point set $S$ with reasonable size. In the future we plan to use a facsimile compression algorithm for the $(x, y)$.

## 4.2 Encoding of elevation using prediction

Normally elevation data contains a high degree of correlation and that means we may predict the elevation value from its neighbors. The method of linear predication has been very successful in image processing [8] and the idea is very simple: to predict the vale of next element in the sequence, just use the last seen element and keep the correction by saving the difference between the two. The sequence z that we are going to compress contains elevation information in points selected by the ODETLAP algorithm.

In addition to linear prediction, we are considering other prediction modes which also exploit the correlation between points selected such as Minimum Spanning Tree(MST) to reorder the points. The hypothesis is that the MST formed by points selected by ODETLAP reflects certain terrain structure that can be used to exploit spatial correlation between the elevation values. Consider the ridge of the mountain for example, suppose points selected by ODETLAP has a bigger chance to fall on the ridges, and points on the same ridge would likely to have similar elevation values, which could help reduce the compressed size if we use the correct spatial based prediction scheme.

Encoding of slope pairs selected in section 3.4 is done similarly. The difference is that now we need to compress $(x, y, z, \Delta_x, \Delta_y)$ pairs. The first two are still done by run length encoding and the last three are compressed separately using prediction and bzip2.

# 5. RESULTS AND ANALYSIS

## 5.1 Test dataset

For this paper, our test dataset are six level 2 DTED DEMs. DTED level 2 has relatively higher resolution and accuracy comparing to DTED-0 and DTED-1. It has post spacing of one arc second (approximately 30 meters). The size of each 400 × 400 test dataset, at 2 bytes per point, is 320KB. The plots of our six dataset are given in Figure 3. Some statistics of the datasets are given in Table 5.1.

| Data | Elev Range | Elev $\sigma$ | Slope Range | Slope $\sigma$ |
|---|---|---|---|---|
| hill1 | 505 | 78.87 | 50.97 | 4.64 |
| hill2 | 745 | 134.4 | 49.82 | 7.87 |
| hill3 | 500 | 59.32 | 42.75 | 3.09 |
| mtn1 | 821 | 146.0 | 52.81 | 9.64 |
| mtn2 | 953 | 152.4 | 49.93 | 8.99 |
| mtn3 | 788 | 160.7 | 60.38 | 11.15 |

**Table 1: Some statistics of the test data**

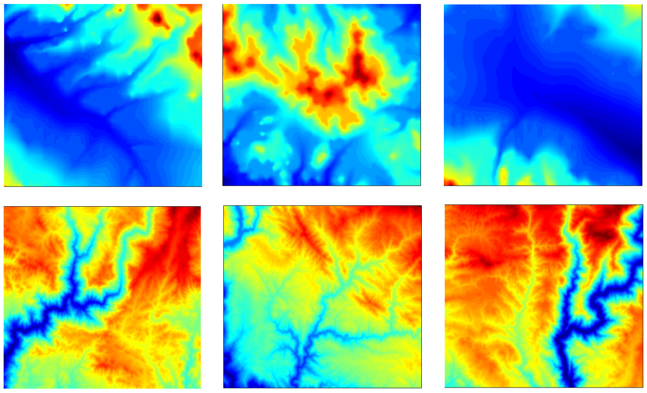


**Figure 3: Test 400×400 30-meter Level 2 DEM**

## 5.2 Evaluation criteria

Evaluating the reconstructed surface is nontrivial. We start with the most basic metric: maximum absolute and RMS elevation error. In Figure 4 we have the compressed file size versus the maximum and RMS elevation error of the reconstructed (decompressed) terrain. We could see the terrain is fairly accurate since the RMSE is well below 10 for a 1/10 compression ratio(recall the original binary size for a 400 × 400 DEM is 320KB). 10 meters of accuracy is already below the accuracy of DTED-2, which is 18 meters of absolute vertical accuracy [5]. It means some of the fine detail that we are losing are possibly just random noise but it is always good to have a compression technique that works well in a high resolution setting since more accurate data are becoming available in the near future [5].

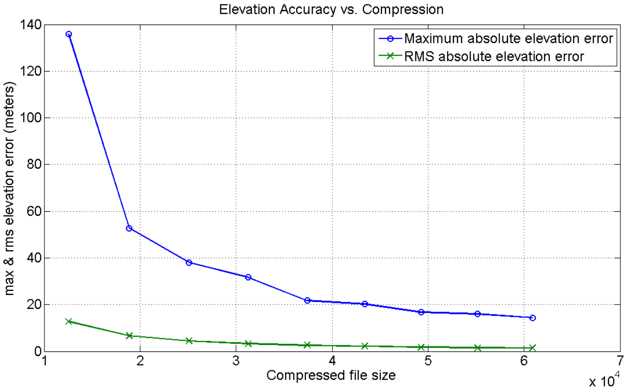


**Figure 4: Elevation error vs. compressed file size.**

Although we always compute statistics, our real metric here is, "does it look good?". Even elevation changes smaller than the known precision are unacceptable if the slopes have artifacts. Accurate slopes are also important for computing erosions, watersheds and for path planning. In Figure 5 we compare the elevation and slope from two stages of our compression and the original DEM. The top row are the elevation visualization and bottom row are the slope visualization. The first column has the results after the first iteration of Algorithm 1, which has a compression ratio of 1:25.3(12645 Bytes); the second column corresponds to the second iteration, where the compression ratio is 1:16.9(18975 Bytes). We could also find their corresponding maximum and RMS error in Figures 4 and 6. From the figures, we could see that the first iteration preserves the general trend of both slope and elevation, but lost some of the high resolution detail because of the high compression ratio. However, after more points are included in the second iteration, a lot more details are restored and most of the important features are captured.

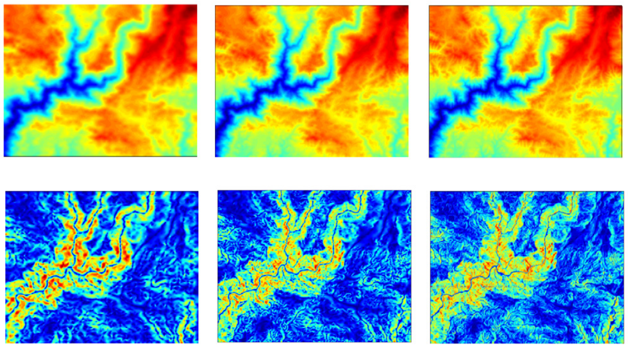


**Figure 5: Reconstructed terrain and slope at different compression levels. From left to right: after first iteration, after second iteration and the original DEM; Top row: elevation visualization, bottom row: slope visualization.**

Accuracy of elevation implies nothing about accuracy of slope. (It was the realization of counter-intuitive properties like this that motivated the formalization of calculus into the new field of mathematical analysis in the 19th century.) In Figure 6, we have the compressed file size versus the slope error. Considering the 30 meters spacing of DTED-2 DEM data, we could see RMS error of our reconstructed terrain is pretty accurate(RMSE of 5 degrees for 1:10 compression).

# 6. CONCLUSION AND FUTURE WORK

In this paper we described a new algorithm to lossily compress the terrain. The focus is on how to achieve higher compression ratio while maintaining the elevation and slope accuracy. ODETLAP framework is very useful for reconstructing terrain surface from very sparse and unevenly distributed elevation data. We made several modifications to the original ODETLAP system so that, without explicitly storing slope values, we can achieve higher slope accuracy in the output terrain. We also discussed run length encoding, linear prediction and minimum spanning tree as ways to encode the intermediate data so that the compression ratio could be higher. The broader impact of ODETLAP

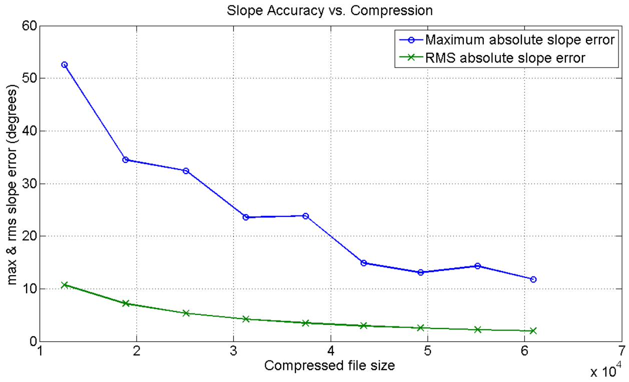


**Figure 6: Slope error vs. compressed file size.**

is that it provides a foundation for a class of customizable compression methods, adaptable to the currently important evaluation metric. Within the ODETLAP framework, we can extend different partial differential equations (PDEs), such as the thin plate PDE. We can add new modules such as the aforementioned Minimum Spanning Tree compression for the *z*. JPEG2000 sometimes performs better than the current version of ODETLAP. However as this is a work-in-progress, we expect the former's limited potential caused by its lack of extensibility soon to render it obsolete for compressing terrain data.

One future direction of our research is the extension of ODETLAP to preserve monotonic terrain structures besides slope. An example would be the terrain compression that preserves shorelines and cliffs. There are two ways to achieve this: either by explicitly storing the shoreline/cliff structure along with terrain itself or by postprocessing the compressed terrain produced by ODELTAP to generate new shorelines/cliff that match the original. The relevance of this extension to the main theme of this paper is that both strive to maintain some important terrain features throughout the terrain compression process. By effectively preserving these features, the compressed terrain becomes more useful and meaningful for the related geographical applications.

## 7. ACKNOWLEDGEMENT

This research was partially supported by NSF grant CMMI-0835762. We thank Professor Barbara Cutler, Professor Badrinath Roysam, Tsz-Yam Lau and You Li for valuable discussions on terrain representation and compression.

## 8. REFERENCES

[1] D. L. Donoho. Compressed sensing. *IEEE Trans. Inform. Theory*, 52(4):1289–1306, April 2006.

[2] W. R. Franklin, M. Inanc, Z. Xie, D. M. Tracy, B. Cutler, and M. V. A. Andrade. Smugglers and border guards: the geostar project at RPI. In *15th ACM International Symposium on Advances in Geographic Information Systems*, Seattle, WA, 2007.

[3] W. R. Franklin, D. M. Tracy, M. Andrade, J. Muckell, M. Inanc, Z. Xie, and B. Cutler. Slope accuracy and path planning on compressed terrain. In *Symposium on Spatial Data Handling*, Montpelier FR, 2008.

[4] M. B. Gousie and W. R. Franklin. Augmenting grid-based contours to improve thin plate DEM generation. *Journal of Photogrammetry and Remote Sensing*, 71(1):69–79, 2005.

[5] B. Heady, G. Kroenung, and C. Rodarmel. High resolution elevation data(HRE) specification overview. In *ASPRS/MAPPS 2009 Conference*, San Antonio, Texas, 2009.

[6] K. H. Jones. A comparison of algorithms used to compute hill slope as a property of the DEM. *Computers & Geosciences*, 24(4):315 – 323, 1998.

[7] D. G. Krige. A statistical approach to some mine valuations and allied problems at the Witwatersrand. Master's thesis, University of Witwatersrand, 1951.

[8] P. Maragos, R. Schafer, and R. Mersereau. Two-dimensional linear prediction and its application to adaptive predictive coding of images. *Acoustics, Speech, and Signal Processing*, 32:1213–1229, 1984.

[9] L. D. Raaflaub and M. J. Collins. The effect of error in gridded digital elevation models on the estimation of topographic parameters. *Environmental Modelling & Software*, 21(5):710–732, May 2006.

[10] D. Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 ACM National Conference*, pages 517 – 524, 1968.

[11] J. Stookey, Z. Xie, B. Cutler, W. R. Franklin, D. Tracy, and M. V. Andrade. Parallel ODETLAP for terrain compression and reconstruction. In *16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 2008.

[12] The Mathworks, Inc. Matlab - mldivide, mrdivide. `http://www.mathworks.com/access/helpdesk/help/techdoc/ref/mldivide.html`, (retrieved 6/15/2009), June 2009.

[13] A. H. Thiessen. Precipitation averages for large areas. *Monthly Weather Review*, 39(7):1082–1084, 1911.

[14] W. Tobler. A computer movie simulating urban growth in the detroit region. *Economic Geography*, 46(2):234–240, 1970.

[15] D. Tracy. *Path planning and slope representation on compressed terrain*. PhD in Computer Science, Rensselaer Polytechnic Institute, 2009.

[16] S. D. Warren, M. G. Hohmann, K. Auerswald, and H. Mitasova. An evaluation of methods to determine slope using digital elevation data. *CATENA*, 58(3):215 – 233, 2004.

[17] A. E. Wren. Trend surface analysis - a review. *Canadian Journal of Exploration Geophysics*, 9(1):39–44, 1973.

[18] Z. Xie, W. R. Franklin, B. Cutler, M. A. Andrade, M. Inanc, and D. M. Tracy. Surface compression using over-determined laplacian approximation. In *Proc. of SPIE Vol. 6697 Advanced Signal Processing Algorithms, Architectures, and Implementations XVII*, San Diego CA, August 2007. paper 6697-15.

[19] L. W. Zevenbergen and C. R. Thorne. Quantitative analysis of land surface topography. *Earth Surface Processes and Landforms*, 12:47–56, 1987.

# PhD Showcase: A Model for Progressive Transmission of Spatial Data Based on Shape Complexity

PhD Student: Fangli Ying
Department of Computer Science
NUI Maynooth
Co. Kildare. Ireland
fying@cs.nuim.ie

PhD Supervisor: Peter Mooney[*]
Environmental Research Center
Environmental Protection Agency
Clonskeagh,Dublin 14. Ireland
p.mooney@epa.ie

PhD Supervisor: Padraig Corcoran
Department of Computer Science
NUI Maynooth
Co. Kildare. Ireland
padraigc@cs.nuim.ie

PhD Supervisor: Adam C. Winstanley
Department of Computer Science
NUI Maynooth
Co. Kildare. Ireland
adamw@cs.nuim.ie

## ABSTRACT

Due to the limited bandwidth available to mobile devices transmitting large amount of geographic data over the Internet to these devices is challenging. Such data is often high-resolution vector data and is far too detailed with respect to most location-based services (LBS) user requirements. A less detailed version may be sent prior to the complete dataset using a progressive transmission strategy. Progressive transmission is generally performed by transmitting a series of independent pre-computed representations of the original dataset at increasing levels of detail where the transitions between these levels are not necessarily smooth. A model is proposed in this paper for selective progressive transmission which will provide smoother transmission over increasing levels of detail. We define criteria for the comparison of similarity between the progressive states of the vector-data based on shape complexity of the polygon features. This allows development of a real-time strategy for the progressive transmission of vector data over the Internet to mobile devices.

## Categories and Subject Descriptors

H.2.8 [**Database Applications** ]: Spatial databases and GIS; H.3.5 [**Online Information Services**]: Data sharing

## General Terms

Human Factors

## Keywords

Progressive Transmission, Vector data, Shape Complexity OpenStreetMap

## 1. INTRODUCTION

Making publicly available geographic datasets available for users to view, download and analyse over the Internet is now an important topic in web-GIS and Location-based Services (LBS). However the increasing amounts of data coupled with sometimes slow communication links to Internet-enabled mobile devices means transmitting such large amounts data over the Internet is often difficult. For example when a user with a mobile device is attempting to download and visualize a large amount of spatial data there are a number of user interaction issues which include the problem of displaying a large amount of spatial data on a small screen and the need to prevent the user from having to wait a long time to receive the full map representation. Progressive transmission is a promising technique to address these practical problems in these situations. This PhD in Computer Science commenced in October 2009. This paper describes the current state of progress of the research work. The aim of the reesarch work is to improve the user experience for users of mobile devices accessing Location-based Services (LBS) when downloading and visualising spatial data on these small-screen devices. Some practical examples of location-based applications for our proposed model include: tourist maps of wildlife areas; environmental monitoring maps; and pedestrian navigation. Our paper is organised as follows. In section 2 we give an overview of related work on progressive transmission both in GIS and other computing disciplines. To give an overview of the types of OpenStreetMap data used in the development of this model we describe the processing of OpenStreetMap XML data in section 3. The detailed description of our model for progres-

[*]Also at: Dept of Comp Sci, NUI Maynooth, Ireland.

The SIGSPATIAL Special, Volume 2, Number 3, November 2010.

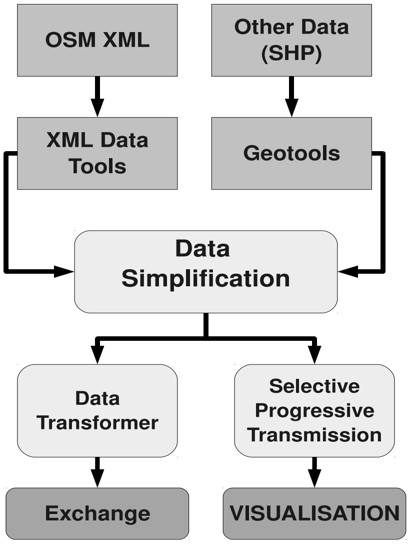


Figure 1: A flowchart of components in this selective progressive transmission

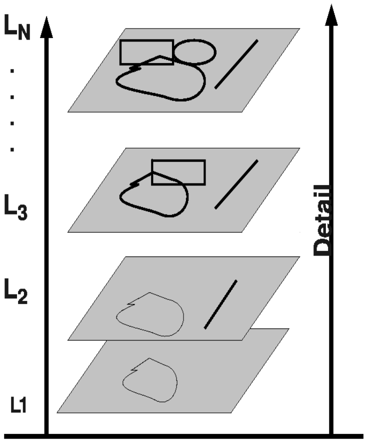


Figure 2: A hypothetical example of selective progressive transmission where level $L_N$ is the original highest level of detail and $L_1$ is the most simplified version of the original $L_N$

sive transmission based on shape complexity begins in section 4 with a detailed description of polygon simplification with emphasis on shape preservation. Selective progressive transmission based on shape complexity rules is described in section 5. The paper closes with section 6 where we provide some initial results of implementation of our model, discussion of the development of the model, and our plan for immediate and long-term future work in this area.

## 2. OVERVIEW OF RELATED RESEARCH

Progressive transmission usually provides a number of precomputed levels of representation of a spatial dataset which can be delivered quickly to meet the time requirements of users. However these pre-computed levels are computed offline and may not be updated regularly. Given the size of some spatial datasets it may take a large amount of time to update these levels. Several authors have highlighted problems with this approach with Jones and Ware [6] stating that these multiple representations may differ markedly in their degree of generalization while the size limitations of mobile devices make it all the more desirable that the level of generalisation to be adapted flexibly to meet the needs of individual users. The use of progressive transmission of spatial data would be much more flexible if it could be performed in real-time and be adapted to the current user's needs and spatial location. Unfortunately there has been very little work carried out on the progressive transmission of spatial data. It is necessary to look to the domain of computer graphics where similar problems also arise in the Levels of Detail (LOD) approximation for the progressive transmission of high detailed geometric models. Lounsbery *et al* [12] proposed the concept of multi-resolution analysis to surfaces of arbitrary topological type. Since real-time switching between LOD for the meshes representing these geometric models may lead to perceptible "popping" effect the goal is to construct a progressive transmission model which has smooth visual transitions between meshes at different resolutions. Eck *et al* [4] describes a series of efficient strategies for progressive transmission of meshes and using selective refinement to optimize the LOD representations. In related work Hoppe [5] proposed an efficient algorithm for selective refinement for incrementally adapting the mesh refinement in order to reduce "popping" effects. This approach is also invertible whereby the progressively transmitted levels of meshes can deconstructed for any level of resolution. In GIS progressive transmission in remains a challenging problem because of the intrinsic complexity of map generalization [1]. One of the most challenging aspects of this part of progressive transmission is topologically consistent generalization methods. The standard method for progressive transmission is to generalized data into a series level of details for incrementally delivery (Lehto). Most readers will be familiar with this concept from the use of map-tile based web mapping such as Google Maps or Bing Maps. While this standard method for progressive tranmission offers multiple-resolution views of the spatial data the user experience can be effective by the similar problem of "popping" mentioned above. This "popping" in progressive transmission of spatial data occurs because of the difference of representations among the pre-computed levels for the given spatial datasets. The naive solution would be to precompute enough levels of representation such that there is a smooth visual transition between levels. However this is completely impractical because for the more levels of resolution required the greater the storage requirements on the server side would be. This would also result in the same data being sent multiple times. The concept of "popping" is best illustrated from the example in Figure 2 where the distance between successive levels is too great (like that between $L_2$ and $L_3$) and information appears to "pop" into the current representation. Map generalization techniques are generally performed in progressive transmission for generating a coarser representation of the data before the remainder of data is incrementally transmitted. There are several solutions for automated line generalizations which have already been defined with respect to several constraints. For example some methods of line simplification were developed to ensure topological consistency [14]. Other methods are intended to preserve shape characteristics [16].

# 3. PROCESSING OSM-XML DATA

Before we describe how to process the OSM-XML we give a brief overview of the use-case scenario for progressive transmission. A user selects an area from a web-based map displaying OpenStreetMap data. Using the OSM API the OSM XML data corresponding to the area selected is downloaded in real-time. Using the Stax XML Java toolkit the OSM XML is processed. The geographic objects (points, polygons, polylines) are then bound to Java objects. The spatial data then undergoes simplification from the highest level of detail ($L_n$) to the most simplified version ($L_1$). The order in which the objects are simplified is maintained using a set of arrays which hold the Java objects containing the spatial data. When simplification is completed the data is then progressively transmitted to the mobile device by progressively transmiting the data out to the mobile device from version $L_1$ to $L_n$. As described in the flowchart in figure 1 the data transformer component can also deliver *packages* of the spatial data at any of the levels of simplification from $L_1$ to $L_n$ in OSM XML format or other vector data formats. OpenStreetMap XML (OSM-XML) is one of several formats in which the raw OSM geographic data is made publicly available for download. Most LBS enabled mobile devices are still unable to handle vector data delivered in XML-based formats such as OSM XML. The problem is compounded by the fact that very often the OSM XML can represent a very large geographical area and/or contains a very large number of geographic objects. OSM XML contains point, line and polygon features. Every spatial attribute (or tag) corresponding to each feature is included in the OSM-XML. Very often the size of OSM-XML files corresponding to very small geographical areas in locations where OpenStreetMap coverage is very good can be several megabytes in size. As illustrated in Figure 1 in section 1 the OSM-XML is downloaded in real-time and processed using an Open Source XML data processing framework called Stax which is suitable for processing XML data using streaming. Stax allows the stream-based processing of the OSM XML files to the mobile device in real-time. The OSM API `http://wiki.openstreetmap.org/wiki/API_v0.6` is used for on-the-fly XML data capture. We can introduce constaints upon the size and extent of the geographical data which the user can select for download and visualisation.

Considering that most LBS-enabled devices have limited storage and computing ability the optimized streaming-based XML API such of Stax is alternative approach for real time pre-processing OSM-XML. Unlike traditional tree-based tools or pure streaming-based tools, such as JDOM or SAX, for XML processing Stax has many advantages. These advantages include the availability of cursor level access to the XML data and effecient memory management techniques. The advantage of Stax is that we can easily move the cursor pointer forward, skip to any specified geographic feature in the XML, and finally extract the OSM data efficiently without prohibitive memory consumption. The software implementation of our model for progressive transmission is written in Java. This Java-based approach means that our application will run on most Java-enabled mobile devices. Moreover, when presented with raw geographic data in some vector data formats, such as ESRI shapefile, the Java Geotools library, is a very useful tool for transforming and processing these data formats. Currently we are using the OpenStreetMap (OSM) database as a case study dataset. However with the use of the Java GeoTools library the model for progressive transmission described in this paper can access any well known vector data format. Figure 1 shows a flowchart of our proposed model for selective progressive transmission.

# 4. SHAPE SIMPLIFICATION WHILE PRESERVING POLYGON CONTOUR CHARACTERISTICS

The screen displays on LBS-enabled devices are usually small and of relatively low resolution. This means that visualisation is not comparable to a map presentation in a GIS or within a desktop application [15]. It is most important to preserve contour shape without an over-represented of detail. Most polygons can be generalised and simplified as they have nodes which can be removed without adversely affecting the overall shape of the polygon and how the shape is interpreted by the human visual system [9, 10]. Informally a polygon can undergo simplification if the removal of a subset of the polygon nodes can be performed without affecting the overall shape of the polygon to such an extent that it is unrecognisable from its original form. Only insignificant vertices can be considered for removal during simplification. Latecki et al. [11] proposed the following metric $K$ which determines the significance of each vertex to the overall shape of the polygon in question. Suppose for some vertex $s$ in the polygon $p$ with incident edges on $s$ called $s_1$ and $s_2$ then the $K$ metric for significance is given by:

$$K(s_1, s_2) = \frac{\beta(s_1, s_2)\, l(s_1)\, l(s_2)}{l(s_1) + l(s_2)}. \qquad (1)$$

Where $l$ is the length function normalized with respect to the total contour length of the polygon, and $\beta(s_1, s_2)$ is the turning angle at the vertex in question. Informally this metric will determine vertices with a greater turning angle and adjacent edges of a greater length as being most significant. The effectiveness of this metric at determining vertex significance is demonstrated by Figure 3 where polygon vertices are highlighted in two polygons $A$ and $B$. The removal of the vertex circled in polygon $B$ in Figure 3 would dramatically alter the overall shape of the contour in question. This node receives a more significant corresponding $K$ value. On the other hand the vertices circled in polygon $A$ could be removed without any significant changes to the overall shape of the contour of polygon $A$ and are assigned an insignificant $K$ value very close to zero.

## 4.1 Simplification of Polygon Shapes

For a set of polygons $P$ the following algorithm is applied to each polygon $p$ in the set. The establishment of the $\lambda$ parameter (by trail and error) allows the simplification of all polygons in $P$ to a similar resolution. Over-represented polygons will undergo more steps of simplification than other most suitably represented polygons. To establish the overall significance $KS$ of removing nodes from a given polygon $p$ the following steps are performed:

1. For each polygon node with adjacent edges $i$ and $j$, determine its corresponding significance by evaluating $K(i, j)$.
2. Calculate $Kmean$ which represents the mean of all $K$ over all polygon nodes; that is $Kmean = \frac{\sum K(i,j)}{N}$ where $N$ is the number of nodes in the polygon $p$.
3. If $Kmean > \lambda$ where $\lambda$ is a predefined threshold then this polygon $p$ is not simplified further as the vertices are all significant. The polygon $p$ is a candidate for direct delivery. Otherwise go to the next step.

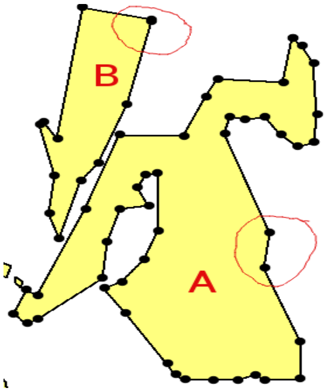


Figure 3: Example of vertex significance for two polygons

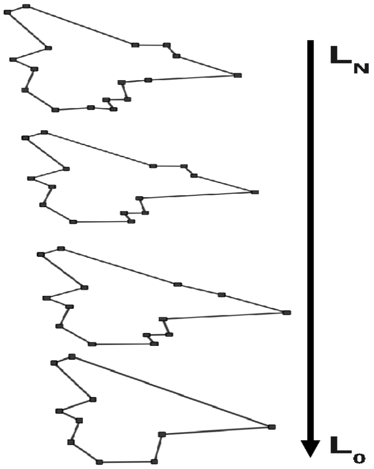


Figure 4: Node Reduction Example

4. Extract the node $n_{i,j}$ in polygon $p$ with minimum $K(i,j)$. This node is removed from the polygon $p$ and placed on top of the node stack.

5. Update the polygon $p$ data structure - node $n(i,j)$ has been removed. The incident nodes $x$ and $y$ to $n$ through edges $i$ and $j$ become connected directly by a new edge $ij$. Recalculate $K$ for both $x$ and $y$ nodes. Recalculate the $Kmean$ value. Go to step 3.

The illustration in Figure 4 shows a simple example of this algorithm applied to a single polygon. The polygon is presented at full resolution at level $L_N$. At each level from $L_N$ down to $L_0$ the lowest level of simplification the nodes with the minimum $K(i,j)$ are removed. At level $L_0$ the $Kmean > \lambda$ and the simplification of this polygon is stopped. Each polygon $p$ in the set of polygons $P$ has a corresponding node stack. This is an array data structure which holds the node objects in sorted order. Each node object $n$ has the following attributes: $K$ value, integers to indicate the nodes that it is connected to in $p$, an index $r$ which indicates its position in the polygon $p$, and a Point object which holds its geographical coordinates. In the Java implementation of this model we have implemented polygons and nodes as `Comparable` objects. These objects can be stored and managed by Java's Collections Framework which offers several efficient libraries for sorting arrays and lists of `Comparable` objects.

### 4.2 Progressive Transmission of Contour Preserved Shapes: Implementation

As described in Section 4.1 the selected polygons are simplified according to rules which preserve the shapes of the polygons. To complete the progressive transmission of the selected polygons to the user's mobile device the process of Section 4.1 (and illustrated in Figure 4) must be reversed. The most simplified representation of the selected polygons are stored at level $L_0$. These polygons are delivered to the mobile device and visualised on the screen using a mapping framework such as OpenLayers. The process of progressive transmission from $L_0$ to full resolution at $L_N$ works by progressively selecting and transmitting nodes from the polygon node stacks. At each iteration $T$ most significant nodes are popped off the node stacks and transmitted. If $T = 1$ then one node is popped off. This node is the node with largest $K(i,j)$ amongst the node stacks of all polygons in $P$. For $T > 1$ the $T$ largest $K(i,j)$ values are popped off the node stacks. When these $T$ nodes are transmitted to the mobile device Javascript/AJAX functionality in the visualisation module updates the map display by updating the polygons to include the newly arrived $T$ nodes. The progressive transmission of the polygons is completed when all of the node stacks are empty or if the user decides to request visualisation of a different geographical area.

One of the problems with this approach is that there is no consideration given to the overall shape complexity of the polygons selected. The $T$ nodes which are progressively transmitted on each level $L_i$ are taken from the node stacks. In the next section we described an enhanced model of selective progressive transmission which takes the shape complexity of the polygons into consideration before transmitting any nodes. Broadly speaking the polygon(s) at $L_i$ which are most dissimiliar to their corresponding full resolution representation at $L_N$ are selected and nodes belonging to these polygons are selectively transmitted.

## 5. SELECTIVE PROGRESSIVE TRANSMISSION BASED ON SHAPE COMPLEXITY

In this section we describe an enhanced version of the selective progressive transmission strategy proposed in Section 4. This version of the selective progressive transmission strategy is based on using rules generated from the shape complexity of the polygons selected by the user. By using shape complexity to calculate the dissimilarity of the polygon(s) at $L_i$ to their corresponding full resolution representation at $L_N$ the polygon(s) which should have nodes added are selected. This builds upon work by Joshi *et. al* [7] who describe a dissimilarity function that can be used in state-of-the-art spatial clustering algorithms. This results in clusters of polygons that are more compact in terms of spatial contiguity. The concept of spatial contiguity is very important in spatial data clustering and visualisation [13]. The dissimilarity function for the shape complexity of a polygons proposed by Joshi *et. al* [7] measures the distance between $n$ scalar spatial attributes of the polygons. The distance can be measured using standard the Euclidean distance metric. If $n = 2$ then this is just 2 dimensional cartesian space. There have been many characteristics proposed to describe the structural shape complexity and characteristics of a polygon object. Brinkoff *et .al* [2] provide a description of the most popular shape complexity measures for spatial data polygons. In other work [19, 18] the authors use a small subset of the measures described by Brinkoff *et .al.* These are described as follows:

- Normalised Area Ratio ($AR$) - the area difference between the area of the polygon $q$ ($A(q)$) object and its convex hull ($A(C(q))$) expressed as $\frac{A(C(q))-A(q)}{A(C(q))}$. An AR value of 1.0 indicates that the convex hull perfectly fits the polygon. As the value approaches 0.0 this indicates an increasingly "spiky" polygon where the convex hull is much larger than the polygon itself.

- Circularity $C$ - an expression of the compactness of the polygon object $q$ - $\frac{4\pi * A(q)}{P(q)^2}$ where $P(q)$ is the perimeter of the polygon object. A circularity value of 1.0 indicates a perfect circle. As the value approaches 0.0, it indicates an increasingly elongated polygon.

### 5.1 Progressive Transmission based on Shape Complexity: Algorithm

This method begins by calculating the Normalised Area Ratio ($AR$) and the Cicularity $C$ of each of the polygons $P$ in the geographical area chosen by the user. The polygons are stored at full resolution at $L_N$. The values of $AR$ and $C$ are stored as scalar attributes of the polygon objects. The establishment of the $\lambda$ parameter is set as before. The process described in Section 4.1 is used to simplify the polygon shapes. When the simplification is completed (that is all polygons in $P$ have reached step 3 in the process from Section 4.1) the $AR$ and $C$ are computed again for all of the simplified polygon objects at $L_0$. Now the selective progressive tranmission can begin to deliver the polygon objects back to the user device for visualisation. In the next section we describe how this algorithm is implemented. A dissimilarity measure $D(L_0^p, L_N^p)$ between a polygon $p$ at the final level of simplifcation $L_0^p$ and the same polygon $p$ at full resolution representation $L_N^p$ is given by:

$$D(L_i^p, L_N^p) = \sqrt{(AR^p_{L_i} - AR^p_{L_N})^2 + (C^p_{L_i} - C^p_{L_N})^2} \qquad (2)$$

The measure $D(L_0^p, L_N^p)$ allows the selective progressive transmission to add more node details to polygons which are very dissimiliar (within the parameters of $Kmean > \lambda$). The measure $D(L_0^p, L_N^p)$ is easily computed. In Ying *et. al* [19, 18] the authors show that the combination of the scalar attributes of $AR$ and $C$ can cluster spatial polygons into two distinct clusters: one with polygons with high $Kmean$ (complex shapes) value the other with low $Kmean$ value (simple shapes).

### 5.2 Progressive Transmission based on Shape Complexity: Implementation

The set of simplified polygon objects at $L_0$ must be progressively transmitted to the user. The next step in the progressive transmission of the vector data to the user is to select which nodes are transmitted next. The process is as follows.

- At $L_0$ the $AR_{L_0}$ and $C_{L_0}$ are computed for all of the simplified polygon objects at $L_0$. For each of the polygons $p$ we calculate the dissimilarity measure $D(L_0^p, L_N^p)$ between the complexity of polygon $p$ at $L_0^p$ and $L_N^p$.

- The $T$ most significant nodes are popped off the node stack for the polygon $p$ with the largest dissimilarity measure $D(L_0^p, L_N^p)$. If $T = 1$ then one node is popped off. This node is the node with largest $K(i, j)$ amongst the node stacks of all polygons in $P$. For $T > 1$ the $T$ largest $K(i, j)$ values are popped off the node stacks. When these $T$ nodes are transmitted to the mobile device Javascript/AJAX functionality in the visualisation module updates the map display by updating the polygons to include the newly arrived $T$ nodes.

- The steps 1 and 2 are repeated for subsequent levels $L_k$ to $L_N$. The parameter $T$ can be adjusted to control the number of levels from $L_k$ to $L_N$

The dissimilarity measure $D(L_i^p, L_N^p)$ could be extended to use additional scalar shape complexity attributes such as those proposed by Brinkhoff *et al*[2]: notches (number of large turning angles in the polygon), polygon convexity measurement, or perimeter ratio.

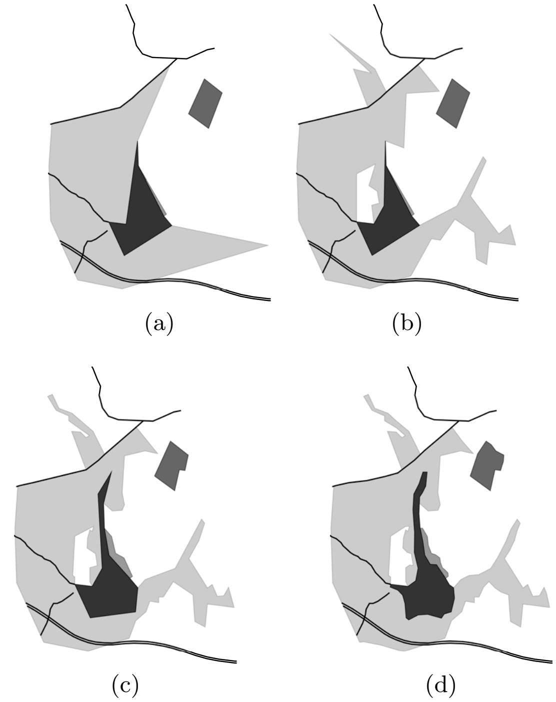
## 6. DISCUSSION AND CONCLUSIONS

The initial work of an early stage PhD in Computer Science is described. A model for progressive transmission of spatial data based on shape complexity has been proposed in this paper. The target area of implementation is in the delivery of spatial data to mobile devices accessing Location Based Services (LBS). As described in the literature review of Section 2 progressive transmission of GIS data is a difficult problem. Our model aims to tackle some of these problems most notably the issue of providing a smooth transmission between *data levels* or representations. Currently our model only simplifies polygons. Polylines are transmitted without simplication. We are currently working on including progressive transmission of polylines also using the Doughas Peucker algorithm for polyline simplification [3]. An issue for further work is comparing our progressive transmission model with compression techniques for spatial data. Several vector-data compressive techniques have been proposed recently namely variable-resolution compression [17] and algorithms for lossy vector data compression [8]. Both techniques demonstrate a feasible and efficient solutions for the compression of vector data, are able to achieve good compression ratios and maintains the main shape characteristics of the spatial objects within the compressed vector data.

Figure 5a to 5d displays some sample steps in the progressive transmission of a set of polygons. The simplified representation of the polygons in question is displayed in Figure 5a and contains 38 polygon vertices in total. Detail is progressively added to this representation in Figure 5b and Figure 5c which contain 70 and 97 polygon vertices respectively. The progressive transmission process completes when the entire data set has been received and integrated as shown in Figure 5d. This final map contains a total of 182 polygon vertices. This example shows the very initial results of the proposed progressive transmission strategy. There are a number of potential practical implementations of progressive transmission of vector data to mobile devices. For example in environmental monitoring a user can move quickly through an area and may stop to make samples or measurements. Not all high resolution data is required at all times and progressively more detailed data can be delivered and visualised depending on the user's requirements.

## Acknowledgments

Fangli Ying is funded by Irish Universities Association and China Scholarship Council. The Environmental Protection Agency provides funding to Dr. Peter Mooney grant (2008-FS-DM-14-S4). Research presented in this paper was also funded by a Strategic Research Cluster grant (07/SRC/I1168) by Science Foundation Ireland under the National Development Plan. The authors gratefully acknowledge these supports.



(a) (b)

(c) (d)

Figure 5: A sample of steps in the progressive transmission of a set of polygons are shown.

## 7. REFERENCES

[1] M. Bertolotto and M. Egenhofer. Progressive transmission of vector map data over the world wide web. *GeoInformatica*, 5(4):345–373, 2001.

[2] T. Brinkhoff, H. Kriegel, and R. Schneider. Comparison of approximations of complex objects used for approximation-based query processing in spatial database systems. In *The Ninth IEEE International Conference on Data Engineering*, Vienna, Austria, Apr. 1993.

[3] D. H. Douglas and T. K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *The Canadian Cartographer*, 10(2):112–122, Dec. 1973.

[4] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle. Multiresolution analysis of arbitrary meshes. In R. Cook, editor, *SIGGRAPH 95 Conference Proceedings*, Annual Conference Series, pages 173–182. ACM SIGGRAPH, Addison Wesley, Aug. 1995. held in Los Angeles, California, 06-11 August 1995.

[5] H. Hoppe. Progressive meshes. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 99–108, New York, NY, USA, 1996. ACM.

[6] C. B. Jones and J. M. Ware. Map generalization in the web age. *International Journal of Geographical Information Science*, 19(8-9):859–870, 2005.

[7] D. Joshi, A. Samal, and L.-K. Soh. A dissimilarity function for clustering geospatial polygons. In *GIS '09: Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 384–387, New York, NY, USA, 2009. ACM.

[8] A. Kolesnikov. Optimal algorithm for lossy vector data compression. In M. Kamel and A. Campilho, editors, *Image Analysis and Recognition*, volume 4633 of *Lecture Notes in Computer Science*, pages 761–771. Springer Berlin / Heidelberg, 2007.

[9] L. Latecki and R. Lakamper. Polygon evolution by vertex deletion. pages 398–409, 1999.

[10] L. J. Latecki and R. Lakaemper. Contour-based shape similarity. *Lecture Notes in Computer Science*, 1614:617–630, 1999.

[11] L. J. Latecki and R. Lakmper. Convexity rule for shape decomposition based on discrete contour evolution. *Computer Vision and Image Understanding*, 73(3):441 – 454, 1999.

[12] M. Lounsbery, T. DeRose, and J. Warren. Multiresolution analysis for surfaces of arbitrary topological type. *ACM Transactions on Graphics. Also: Technical Report, 93-10-05b, Dept. of Comp. Sci., Univ. of Washington*, 1997.

[13] R. T. Ng and J. Han. Clarans: A method for clustering objects for spatial data mining. *IEEE Trans. on Knowl. and Data Eng.*, 14(5):1003–1016, 2002.

[14] S. Z. P. M. van der Poorten and C. B. Jones. Topologically-consistent map generalisation procedures and multi-scale spatial databases, 2002.

[15] M. Sester and C. Brenner. Continuous generalization for visualization on small mobile devices. In P. Fisher, editor, *11th International Symposium on Spatial Data Handling*, Developments in Spatial Data Handling, pages 355–368. Springer Berlin Heidelberg, August 2005.

[16] Z. Wang and J.-C. Muller. Line generalization based on analysis of shape characteristics. *Cartography and Geographic Information Science*, pages pp. 3–15(13), January 1998.

[17] B. Yang, R. S. Purves, and R. Weibel. Variable-resolution compression of vector data. *Geoinformatica*, 12(3):357–376, 2008.

[18] F. Ying, P. Mooney, and P. Corcoran. Using shape complexity to guide simplification of geospatial data for use in location-based services. In G. Gartner and M. LIU, editors, *The 7th International Symposium on LBS & TeleCartography*, page (To appear), Hidleberg, Germany, September 2010. Springer - Lecture Notes in Computer Science.

[19] F. Ying, P. Mooney, P. Corcoran, and A. Winstanley. Polygon processing on openstreetmap xml data. In M. Haklay, J. Morely, and H. Rahemtulla, editors, *Proceedings of the GIS Research UK 18th Annual Conference*, pages 149–154, London, England, 2010. University College London.

# PhD Showcase: Applications of Graph Algorithms in GIS

PhD Student: Stéphanie Vanhove
Department of Applied Mathematics and
Computer Science
Ghent University
Krijgslaan 281 - S9
9000 Ghent
Belgium
stephanie.vanhove@ugent.be

PhD Supervisor: Veerle Fack
Department of Applied Mathematics and
Computer Science
Ghent University
Krijgslaan 281 - S9
9000 Ghent
Belgium
veerle.fack@ugent.be

## ABSTRACT

This paper describes ongoing PhD research on applications of graph algorithms in Geographical Information Systems. Many GIS problems can be translated into a graph problem, especially in the domain of routing in road networks. Our research aims to evaluate and develop efficient methods for different variants of the routing problem.

Standard existing shortest path algorithms are not always suited for use in road networks, e.g. in a realistic situation forbidden turns and turn penalties need to be taken into account. An experimental evaluation of different methods for this purpose is presented.

Another interesting problem is the generation of alternative routes. This can be modelled as a $k$ shortest paths problem, where a ranking of $k$ paths is desired rather than only the shortest path itself. A new heuristic approach for generating alternative routes is presented and evaluated.

## Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*Routing and Layout*; E.1 [**Data Structures**]: Graphs and Networks; G.2.2 [**Discrete Mathematics**]: Graph Theory—*Graph algorithms, Network problems*

## General Terms

Algorithms, Experimentation, Performance

## Keywords

Graphs, road networks, routing, turn penalties, alternative paths, $k$ shortest paths, heuristics

## 1. INTRODUCTION

Road networks can easily be modelled as a graph where nodes represent intersections and dead ends and arcs represent directed road segments. Arc weights usually represent either distances or travel times. The recent popularity of route planners and navigation systems has renewed the interest in the applications of graph algorithms to road networks, especially routing algorithms. However, there are some additional requirements for these applications. On the one hand, the used models must represent the real world as realistically as possible. On the other hand, the algorithms must be very fast, since users prefer short query times and servers need to answer many queries. Even though the shortest path problem is a classic problem in graph theory, the existing standard algorithms such as the algorithm of Dijkstra [2] cannot always immediately be used for route planning. One example is the presence of forbidden turns in road networks. Standard shortest path algorithms do not take this into account at all, even though it would be unacceptable if a navigation system instructs a driver to take an illegal turn. Moreover, turns can imply additional waiting times, e.g. at stoplights, another issue which is not handled by standard shortest path algorithms. Figure 1 shows two situations where a standard shortest path algorithm would calculate either an incorrect or suboptimal route. Different methods have been proposed for this problem, but up to now it has remained unclear how well these algorithms perform on real-life road networks and how these algorithms compete with each other. A study of these algorithms is presented in Section 2. Another interesting variant of the shortest path problem is the generation of alternative routes. This is very

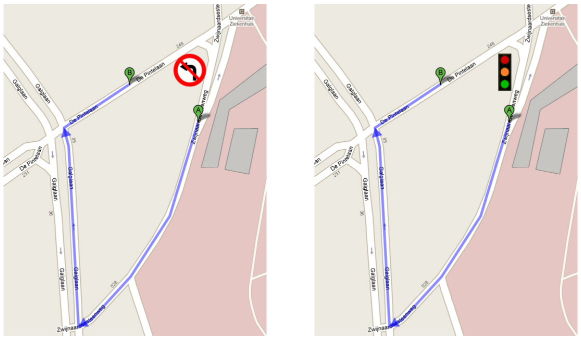


**Figure 1: Two situations where a standard shortest path algorithm would take the obvious left turn to go from A to B. In reality however, the best route is the detour shown in the pictures because of either a forbidden turn (left) or a long waiting time at stoplights (right).**

The SIGSPATIAL Special, Volume 2, Number 3, November 2010.

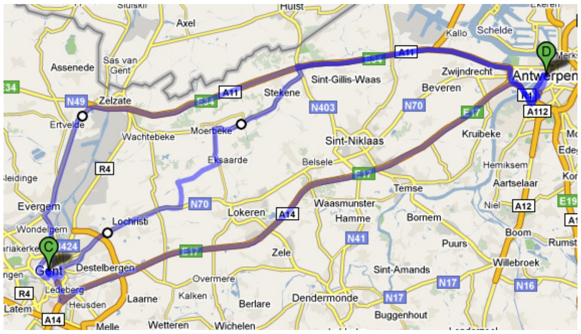


**Figure 2: Alternative routes from Ghent to Antwerp in Google Maps.**

commonly used, as can be seen in Figure 2. In graph theory, the problem of calculating a ranking of shortest paths is called the $k$ *shortest paths* problem, where $k$ is the number of paths to be calculated. Not only can this be useful for generating alternative routes, but $k$ shortest paths algorithms also serve as a basis for methods for optimizing multiple parameters. Such methods (e.g. Mooney et al. [8]) can be used to find e.g. a fast but scenic route. The most scenic route can be chosen from a ranking of the $k$ shortest paths. Similarly, a set of dissimilar paths can be selected from a ranking of $k$ shortest paths. This can be used for generating dissimilar routes for the transportation of hazardous materials in order to spread the risk. Such a method is presented by Dell'Olmo et al. [1]. However, algorithms for the $k$ shortest paths problem tend to be very time-consuming. This is a major issue in interactive routing applications. On the other hand, in routing applications obtaining a *good* solution very fast can be more interesting than obtaining the *exact* solution a lot slower. In Section 3 we present a new heuristic which calculates an approximation of the $k$ shortest paths with results of good quality much faster than the exact algorithm. Section 4 outlines the possibilities for future work.

## 2. TURN RESTRICTIONS

There are two kinds of turn restrictions in road networks: turns can either be forbidden (*turn prohibitions*) or imply an additional cost (*turn costs*). In our examples we will assume that U-turns are always forbidden. The next sections describe the different methods for the shortest path problem with turn restrictions and an evaluation of these methods.

### 2.1 Modelling turns

#### Direct method

Gutiérrez and Medaglia [5] present an adaptation of the algorithm of Dijkstra which takes turn prohibitions into account. We will call this method the *direct method* since it operates directly on the original graph, unlike the other considered methods. While the algorithm of Dijkstra assigns labels to nodes, the direct method assigns labels to arcs. Also, for every transition from one arc to another, a check is performed to make sure that the turn is not prohibited. Since the graph itself does not model turn prohibitions, this information is stored separately in a data structure outside the graph. Turn costs are not considered in their work. We have adapted the direct method in order to consider turn costs as well. Further details of this method are omitted for space reasons.

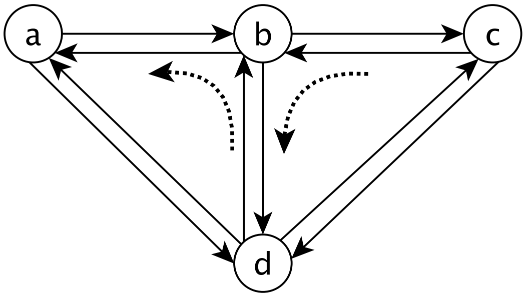


**Figure 3: Graph with forbidden turns. Dashed arrows indicate forbidden turns: it is legal to move from arc $(d, b)$ to $(b, c)$ but it is illegal to move from arc $(d, b)$ to $(b, a)$.**

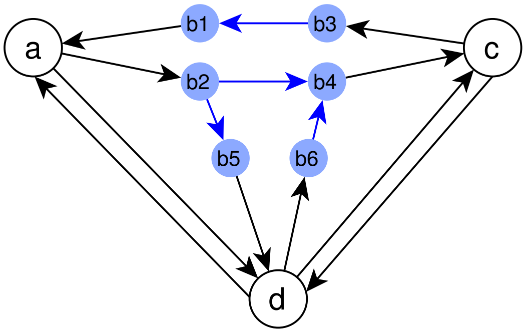


**Figure 4: Node splitting: transformed graph for the graph in Figure 3. Node $b$ is split into 6 nodes: $b_1$ .. $b_6$, each representing an arc incident with node $b$. Arcs between the split nodes represent legal turns. Split nodes and the arcs between them are shown in blue.**

#### Node splitting

Kirkby and Potts [7] and Speičys et al [10] present another method called *node splitting*. This method requires a graph transformation. Every node in the graph with a turn cost or turn prohibition is split into several nodes: one for every incoming or outgoing arc. Then, for every legal turn, an arc is added between the two nodes representing the arcs of the turn. The weight of this new arc is the turn cost, which can be zero or more. Illegal turns can no longer be taken in the graph since there is no arc connecting the corresponding nodes. Figure 3 shows a graph with two forbidden turns. Figure 4 shows its transformed graph. The main advantage of this method is that any standard shortest path algorithm, e.g. the algorithm of Dijkstra can be applied to the transformed graph, so no new algorithm is needed.

#### Line graph

The *line graph* method is another graph transforming method presented by Winter [11]. While the node splitting method

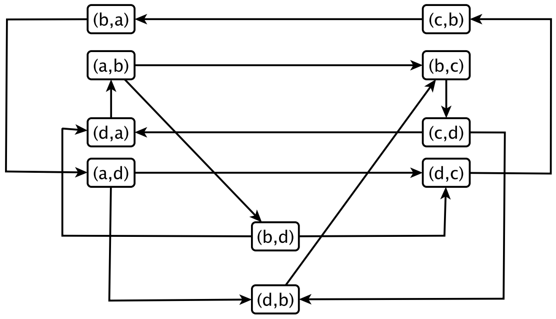


**Figure 5: Line graph: transformed graph for the graph in Figure 3. Every node in the line graph represents an arc in the original graph. Arcs in the line graph represent legal turns in the original graph.**

only affects those nodes with turn costs or turn prohibitions, the line graph method always transforms the entire graph. Nodes in the transformed graph represent arcs in the original graph. Arcs in the transformed graph represent legal turns in the original graph. Arc weights in the transformed graph represent arcs weights in the original graph as well as turn costs. The transformed graph is called the line graph. Figure 5 shows the transformed graph for the graph in Figure 3. Just like the node splitting method, this methods allows running any standard shortest path algorithm on the transformed graph.

## 2.2 Experimental evaluation

We performed several experiments aiming to evaluate the methods mentioned above on real-life road networks. Of course query time is important, so time measurements were performed. However, memory usage is important too. Two of the three methods require a graph transformation, which can possibly result in a much larger graph, while the direct method needs extra memory to store the information on turn restrictions separately. Therefore, memory usage was measured as well for the three methods. All algorithms were implemented, compiled and executed in Java version 1.6.0_03. All tests were run on an Intel dual core 2.13 GHz machine with 2 Gigabyte RAM running Linux. In the next paragraphs we make a distinction between turn prohibitions and turn costs, since different test data were used.

### *Turn prohibitions*

For turn prohibitions, the experiments were performed on road networks provided by Navteq [9] with real-world turn prohibitions. The size of these road networks is in a range bounded by 39,883 (Luxembourg) nodes and 1,017,242 nodes (The Netherlands). As can be expected, only a small fraction of the turns is forbidden (less than 1%). The results can be seen in Figure 6. All results are ratios compared to the algorithm of Dijkstra. E.g. if the memory usage ratio is 3, then the method needs 3 times more memory than the original graph representation without turn restrictions. If the query time ratio is 3, then the query time for this method is 3 times the query time of the algorithm of Dijkstra. The top chart shows that the direct method and node splitting take about the same amount of memory, while the line graph method has a much higher memory usage. When looking at

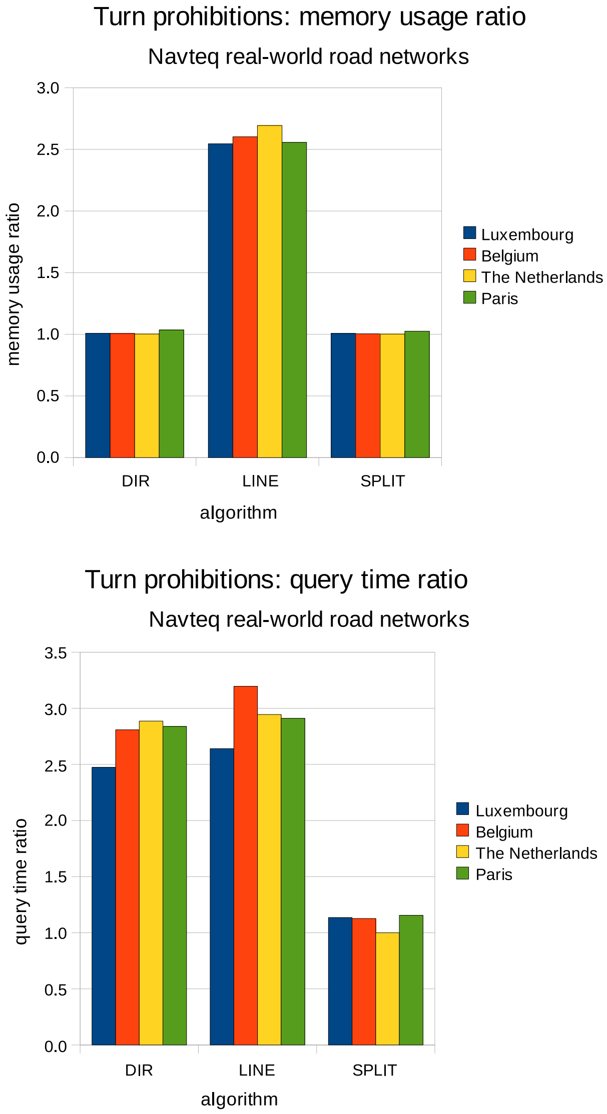
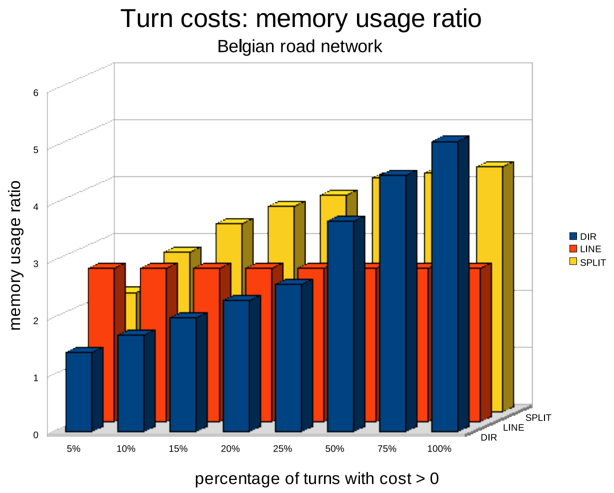


**Figure 6: Turn prohibitions: results for Navteq real-world road networks. Memory usage ratio (top) and average query time ratio (bottom) are shown for 4 different road networks. Three methods are compared: the direct method (DIR), line graph (LINE) and node splitting (SPLIT).**

the time measurements in the chart in the bottom however, the node splitting method is clearly the fastest. Hence, we can conclude that the node splitting method performs best on realistic road networks with turn prohibitions.

### *Turn costs*

To the best of our knowledge, no real-life data with turn costs are available at this moment. To overcome this obstacle, real-world road networks were used but the turn costs were added randomly. The road networks are provided by the University of Karlsruhe in the DIMACS format [3] and represent the road networks for different European countries. In this abstract, results for the Belgian road network



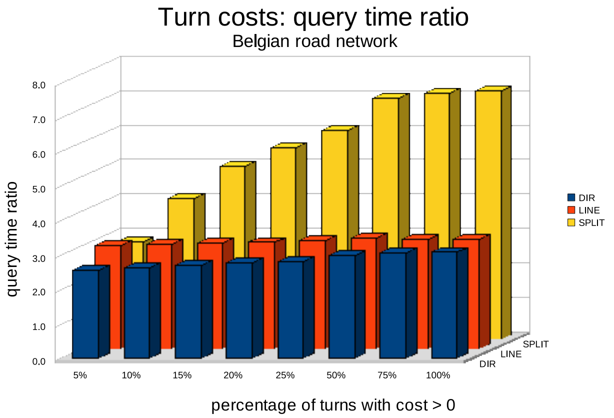


**Figure 7: Turn costs: results for the Belgian road network. Memory usage ratio (top) and average query time ratio (bottom) for different percentages of turn costs are shown. Three methods are compared: the direct method (DIR), line graph (LINE) and node splitting (SPLIT). The graph has 458,403 nodes, 1,085,076 arcs and 2,922,504 turns. The original graph size is 57.72 MB. The average query time for a standard Dijkstra algorithm is 254.49 ms.**

are shown, but results are similar for the other countries. In theory, when applying turn costs to a road network, every turn has its own cost associated to it. However, in a real-life situation, it is very likely that a data provider does not provide a turn cost for every turn in the network, since visiting every turn would be an extremely expensive and time-consuming task. A data provider would probably focus initially on the busiest roads and intersections and possibly keep the less important roads for a later phase. The data could e.g. contain turn costs for 5% of the turns. For this reason, different percentages of available turn costs are considered in the experiments, namely 5%, 10%, 15%, 20%, 25%, 50%, 75% and 100%. It should be noted that 100% available turn costs can still be realistic if the turn costs are calculated automatically, e.g. based on the angle. The results can be seen in Figure 7. The results are again ratios as explained in the previous section. The direct method and line graph method show very similar query times, which also seem to be independent of the percentage of turn costs. This can be expected since these methods transform the entire graph or perform no graph transformation at all, respectively, so the number of nodes in the final graph is independent of the percentage of turn costs for both methods. The node splitting method is never faster than the other two methods. On the other hand, memory usage seems to be independent of the percentage of turn costs for the line graph method but not for the direct method. This can be explained by the fact that the line graph always transforms the entire graph and doesn't store any additional information, while the direct method keeps the original graph but needs to store additional information for every turn cost. As can be seen in the chart, this leads to increasing memory usage for higher percentages of turn costs. The direct method appears to be more memory-efficient for lower percentages of turn costs. For higher percentages however, the line graph method is more memory-efficient than the direct method. So we can conclude that the direct method is best suited for graphs with fewer turn costs (up to about 25%) while the line graph performs better for graphs with many turn costs.

# 3. ALTERNATIVE ROUTES

## 3.1 K shortest paths

The second problem we discuss in this paper is the generation of alternative paths. We will assume that loops in the paths are forbidden, a natural assumption in road networks. For this problem - $k$ shortest paths without loops - an influential algorithm was proposed by Yen [12], which was the basis for many of the currently known algorithms (e.g. Hershberger et al. [6], Gotthilf and Lewenstein [4]). However, as mentioned in the introduction, routing applications are very time-critical and the existing algorithms tend to be too slow for this purpose. Therefore, we developed a heuristic approach which does not aim to find an exact solution but is much faster than the exact algorithms while the results are still of good quality. In the next sections we describe the general principle (*deviation path algorithms*) on which most algorithms are based, present our heuristic approach and report the results.

## 3.2 Deviation path algorithms

Our heuristic is based on the algorithm of Yen [12]. Both are examples of *deviation path algorithms*, which are based on the fact that any shortest path in the ranking always deviates at some point from a path previously found. A path can either immediately deviate from a path from the start node, or coincide with a path up to some node and then deviate from it. Figure 8 shows all possible deviations from a path with 5 nodes (note that a path can deviate from another path to join it again later).

Deviation path algorithms start by calculating the shortest path using any shortest path algorithm. In our work the algorithm of Dijkstra is used for this purpose. This shortest path is then added to a collection C (usually a priority queue). Then the algorithm fetches the shortest path $P$ from $C$ in every iteration, adds it to a list $L$ containing the ranking of shortest paths found so far, and calculates deviations from $P$ which are added to $C$. The algorithm is

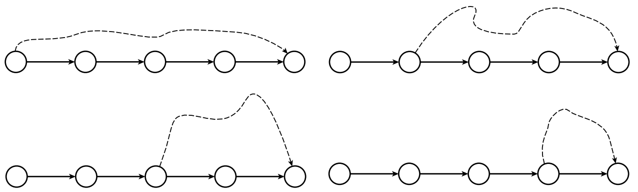


**Figure 8: All possible deviations (dashed lines) from a path (solid lines) with 5 nodes.**

finished after $k$ iterations. Algorithm 1 outlines this general principle, which is shared by all deviation path algorithms, who then differ in their method for calculating deviations.

```
Algorithm 1 Deviation path algorithms
Require: graph $G$, number of shortest paths $k$, source $s$, target $t$
Ensure: sorted collection $L$ of $k$ shortest paths
$P \leftarrow$ calculate shortest path from $s$ to $t$
add $P$ to a collection $C$
for $i$ from 1 to $k$ do
    $P \leftarrow$ shortest path in $C$
    remove $P$ from $C$
    add $P$ to $L$
    calculate deviations and add them to $C$ {algorithms differ here}
end for
```

### 3.3 Our approach

The method for calculating deviations used by our approach is similar to the method used in Yen's algorithm. The algorithm of Yen forbids every arc $(v_i, v_{i+1})$ on a path $P$ from $s$ to $t$ one by one, and calculates the new shortest path $P'$ from $v_i$ to $t$. In this calculation, all the nodes on $P$ preceding $v_i$ are also forbidden. A new path from $s$ to $t$ is then created by appending $P'$ to the subpath of $P$ from $s$ to $v_i$. This results in a very large amount of time-consuming shortest path calculations. Our heuristic aims to speed up the calculation of these shortest paths. Instead of performing so many shortest path calculations, the shortest paths are retrieved from precomputed information. The heuristic uses a backward shortest path tree $T$ which is precomputed and thus computed only once. The shortest path from any node in the graph to the target node $t$ can be looked up in $T$ very fast. Instead of actually computing the shortest path from a node $v_i$ to $t$, the heuristic calculates deviations by concatenating every outgoing arc $(v_i, x)$ from $v_i$ with $x \neq v_{i+1}$ to the shortest path from $x$ to $t$ fetched in $T$. Of course, the possibility exists that this path is no longer valid in the graph since some nodes and arcs have been forbidden in the meantime. This needs to be checked before creating the full $s - t$ path and adding it to $C$. Figure 9 illustrates this idea.

*Complexity*

When the algorithm of Dijkstra is used for shortest path calculations, Yen's algorithm has a time complexity of $O(kn(m + n \log n))$, with $n$ the number of nodes and $m$ the number of arcs in the graph. Since road networks are sparse, it can be assumed that $m = O(n)$, resulting in a time complexity

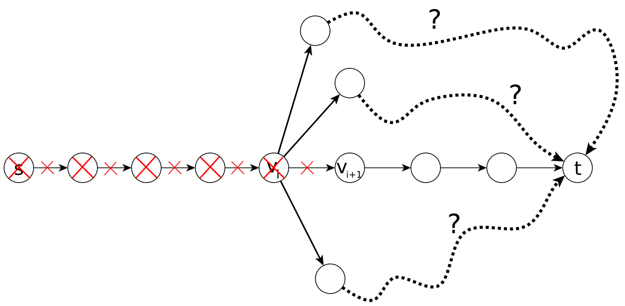


**Figure 9: How the heuristic works. Solid lines indicate the current path $P$ from $s$ to $t$. Red crosses indicate forbidden nodes and arcs. A detour is necessary from $v_i$ to $t$. Dashed lines indicate other outgoing arcs from $v_i$. Dotted lines indicate paths from these neighbours to $t$, which can immediately be looked up in the shortest path tree $T$. These paths are not allowed to pass through already forbidden nodes or arcs.**

of $O(kn^2 \log n)$ for the algorithm of Yen. Our heuristic reduces this time complexity to $O(kn^2)$ (details omitted for space reasons). Even though the complexities only differ by a logarithmic factor, the heuristic performs much better in practice. In the time complexity of the algorithm of Yen, a factor $O(n \log n)$ is attributed to shortest path calculations which can involve (almost) the entire graph. On the other hand, the heuristic does not perform these shortest path calculations, and the $O(n^2)$ factor is limited to iterating over the found paths. Although theoretically a path *can* have a length of $n$, this upper bound is never reached in practice, resulting in much faster running times. The results presented in the following section clearly confirm this statement.

### 3.4 Results

The heuristic was tested on several road networks. Figure 10 shows the results for the Navteq Belgian road network with $k = 1{,}000$. The results are similar for other road networks provided by Navteq and by the University of Karlsruhe. Three different parameters were tested. The first parameter (shown in the first chart of Figure 10) is the ranking of the $k^{th}$ path found by the heuristic in an exact ranking of shortest paths. E.g. for $k = 1{,}000$, if the ranking of the 1,000th path found by the heuristic is 1,006, then the heuristic has missed 6 paths. The results show that the heuristic often misses very few paths or even no paths at all. In some other cases, more paths are missed, but always within acceptable bounds, as can be seen from the results for the second parameter in the second chart of Figure 10. This shows the weight increase for the $k^{th}$ path found by the heuristic. E.g. if the weight increase is 0.80%, then the $k^{th}$ shortest path found by the heuristic is 0.80% longer than the exact $k^{th}$ shortest path. The results show that this value is always below 1%. A weight increase of less than 1% is neglectable in a road network, making the results of the heuristic very useful in practice. A third important parameter is query time, since of course a heuristic calculating an approximation is only useful if it is significantly faster than the exact algorithm. The speedup of our heuristic compared to the exact algorithm of Yen can be seen in the third chart. The re-

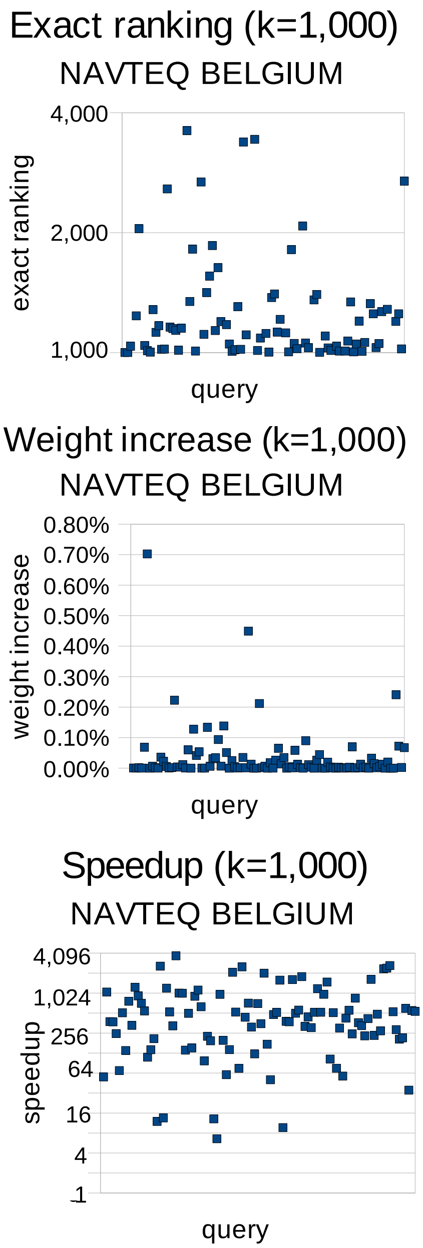


**Figure 10: Results for the Navteq Belgium road network (564,477 nodes and 1,300,765 arcs) with $k$ = 1,000. The exact ranking of the $k^{th}$ path found by the heuristic, the percentual weight increase and the speedup are shown, for 100 random queries.**

sults show that the heuristic is always faster than the exact algorithm, often hundreds or even thousands times faster. This is a very significant advantage of the heuristic for use in interactive routing applications.

## 4. FUTURE WORK

In the future we aim to further optimize our heuristic for the $k$ shortest paths problem. Even though the heuristic performs well in most cases, there is currently still a small number of cases where the heuristic misses a substantial number of paths. We aim to further reduce this number or, ideally, eliminate these outliers. As mentioned in Section 1, $k$ shortest paths algorithms can also serve as a basis for generating dissimilar paths. This can e.g. be interesting for the generation of alternative routes which can be used when weather conditions are not favorable on the usual route. In this case the alternative routes should coincide as little as possible with the first route, otherwise the alternative routes will suffer from the same weather conditions. We aim to develop a new method for this problem based on our heuristic for the $k$ shortest paths problem. Eventually, it would be interesting to include turn restrictions in our methods for $k$ shortest paths and dissimilar paths.

## 5. ACKNOWLEDGEMENT

Stéphanie Vanhove is supported by a research grant of the Research Foundation - Flanders.

## 6. REFERENCES

[1] P. Dell'Olmo, M. Gentili, and A. Scozzari. Finding dissimilar routes for the transportation of hazardous materials. In *Proceedings of the 13th Mini-EURO Conference on Handling Uncertainty in the Analysis of Traffic and Transportation Systems.*, pages 785–788, 2002.

[2] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[3] DIMACS. 9th dimacs implementation challenge on shortest paths. *http://www.dis.uniroma1.it/∼challenge9/*, 2005.

[4] Z. Gotthilf and M. Lewenstein. Improved algorithms for the $k$ shortest paths and the replacement paths problems. *Information Processing Letters*, 109:352–355, 2009.

[5] E. Gutiérrez and A. L. Medaglia. Labeling algorithm for the shortest path problem with turn prohibitions with application to large-scale road networks. *Annals OR*, 157(1):169–182, 2008.

[6] J. Hershberger, M. Maxel, and S. Suri. Finding the $k$ shortest simple paths: a new algorithm and its implementation. *ACM Trans. Algorithms*, 3(4):45, 2007.

[7] R. F. Kirby and R. B. Potts. The minimum route problem for networks with turn penalties and prohibitions. *Transportation Research*, 3:397–408, 1969.

[8] P. Mooney and A. Winstanley. An evolutionary algorithm for multicriteria path optimization problems. *International Journal of Geographical Information Science*, 20:401–423, 2006.

[9] NAVTEQ. Navteq network for developers. *http://www.nn4d.com/*, 2007.

[10] L. Speičys, C. S. Jensen, and A. Kligys. Computational data modeling for network-constrained moving objects. *GIS '03: Proceedings of the 11th ACM international symposium on Advances in geographic information systems*, pages 118–125, 2003.

[11] S. Winter. Modeling costs of turns in route planning. *Geoinformatica*, 6(4):345–361, 2002.

[12] J. Y. Yen. Finding the $k$ shortest loopless paths in a network. *Management Science*, 17(11):712–716, 1971.

# UBICOMP 2011
## Beijing ,China

13th ACM International Conference on Ubiquitous Computing
September 17-21, 2011

13th International Conference on Ubiquitous Computing (Ubicomp 2011) will be held in Beijing, China on September 17-21, 2011. Ubicomp 2011 welcomes original, high-quality research contributions that advance the state of the art in the design, development, deployment, evaluation and understanding of ubiquitous computing systems and their applications. Ubicomp is an interdisciplinary field that includes technologies that bridge the digital and physical worlds, systems and applications that incorporate such technologies, infrastructures that support them, human activities and experiences these technologies facilitate, and conceptual overviews that help us understand – or challenge our understanding of – the impact of these technologies.

The Ubicomp conference is the premier international venue in which novel results in these areas are presented and discussed among leading researchers, designers, developers and practitioners in this field. Ubicomp 2011 will include a highly selective single-track program of full papers and notes. Relevant topic areas for full papers and notes include, but are not limited to:

- devices & techniques – descriptions of the design, architecture, usage and evaluation of devices and techniques that create valuable new capabilities for ubiquitous computing
- systems & infrastructures – descriptions of the design, architecture, deployment and evaluation of systems and infrastructures that support ubiquitous computing
- applications – descriptions of the design and/or study of applications that leverage Ubicomp devices and systems
- methodologies & tools – new methods and tools applied to studying or building Ubicomp systems and applications
- theories & models – critical analysis or organizing theory with clear relevance to the design or study of Ubicomp systems
- experiences – empirical investigations of the use of new or existing Ubicomp technologies with clear relevance to the design and deployment of future Ubicomp systems

Ubicomp 2011 encourages full papers and notes that reflect the breadth and scope of Ubicomp research, including conceptual development, empirical investigations, technological advances, user experiences, and more. Although it is expected that papers will focus on one or a small number of the aforementioned areas, authors should write for the broader Ubicomp audience, and make clear how the work contributes to the Ubicomp field as a whole.

For more information about Ubicomp 2011, please check:

**http://www.ubicomp.org/ubicomp2011**

Brought to you by:

acm

sigmobile

SIGCHI

acm SIGSPATIAL

locally organized byTsinghua University

清華大学 Tsinghua University

**Important Dates**:

- Paper submission: Apr. 15, 2011
- Notification: Jun. 18, 2011

**General Co-Chairs**

Yuanchun Shi, Tsinghua University, China

James Landay, University of Washington, USA

**Program Co-Chairs**

Xing Xie, Microsoft Research Asia, China

Yvonne Rogers, Open University, UK

Don Patterson, UC Irvine, USA

For inquiries, please contact:
chairs2011@ubicomp.org

# join today!

# SIGSPATIAL & ACM

www.sigspatial.org www.acm.org

The **ACM Special Interest Group on Spatial Information** (SIGSPATIAL) addresses issues related to the acquisition, management, and processing of spatially-related information with a focus on algorithmic, geometric, and visual considerations. The scope includes, but is not limited to, geographic information systems (GIS).

The **Association for Computing Machinery** (ACM) is an educational and scientific computing society which works to advance computing as a science and a profession. Benefits include subscriptions to *Communications of the ACM*, *MemberNet*, *TechNews* and *CareerNews*, plus full access to the *Guide to Computing Literature*, full and unlimited access to thousands of online courses and books, discounts on conferences and the option to subscribe to the ACM Digital Library.

| | |
|---|---|
| ❑ SIGSPATIAL (ACM Member) | $ 15 |
| ❑ SIGSPATIAL (ACM Student Member & Non-ACM Student Member) | $ 6 |
| ❑ SIGSPATIAL (Non-ACM Member) | $ 15 |
| ❑ ACM Professional Membership ($99) & SIGSPATIAL ($15) | $114 |
| ❑ ACM Professional Membership ($99) & SIGSPATIAL ($15) & ACM Digital Library ($99) | $213 |
| ❑ ACM Student Membership ($19) & SIGSPATIAL ($6) | $ 25 |
| ❑ Expedited Air for *Communications of the ACM* (outside N. America) | $ 39 |

## payment information

Name ______________________________

ACM Member # ______________________________

Mailing Address ______________________________

______________________________

City/State/Province ______________________________

ZIP/Postal Code/Country ______________________________

Email ______________________________

Fax ______________________________

**Mailing List Restriction**
ACM occasionally makes its mailing list available to computer-related organizations, educational institutions and sister societies. All email addresses remain strictly confidential. Check one of the following if you wish to restrict the use of your name:

❑ ACM announcements only
❑ ACM and other sister society announcements
❑ ACM subscription and renewal notices only

Credit Card Type: ❑ AMEX ❑ VISA ❑ MC

Credit Card # ______________________________

Exp. Date ______________________________

Signature ______________________________

Make check or money order payable to ACM, Inc

ACM accepts U.S. dollars or equivalent in foreign currency. Prices include surface delivery charge. Expedited Air Service, which is a partial air freight delivery service, is available outside North America. Contact ACM for more information.

**Questions? Contact:**
ACM Headquarters
2 Penn Plaza, Suite 701
New York, NY 10121-0701
voice: 212-626-0500
fax: 212-944-1318
email: acmhelp@acm.org

**Remit to:**
**ACM**
**General Post Office**
**P.O. Box 30777**
**New York, NY 10087-0777**

**SIGAPP29**

www.acm.org/joinsigs

Association for Computing Machinery

*Advancing Computing as a Science & Profession*

The SIGSPATIAL Special