The Civium World Model

Spatial and Semantic Issues in Pervasive Computing

Dominic Widdows, Peter Lucas, David Holstius, Michael Higgins

Technical Report MAYA–07013

MAYA Design, Inc.
Pittsburgh
USA

June 15, 2007

Abstract

The increasing pervasiveness of spatial information in small, personal devices such as GPS-enabled cellphones poses new opportunities and challenges. Such devices can act as sensors and transmitters of geographic data, and increasingly, as personalized information assistants, containing data specifically relevant to the user and accessible through small interfaces. The availability of such capacities on small, standalone devices, combined with the fragility of many mobile information networks, leads to considerable information research challenges. This paper addresses some of the challenges encountered when trying to design a rich geographic information experience for users of sporadically connected, real-time networks. Scalable and localizable solutions involve the efficient inference of semantic conclusions from sensor data, and the use of these semantic features to enable the users of small devices to reliably retrieve and navigate related information. This paper attempts to outline several aspects of the design of a collaborative distributed GIS application. A recurring theme is the interplay between (continuous) geospatial measurements, and the extraction and modelling of (discrete) semantic knowledge about the world.

1 Introduction

Many processes in human information gathering involve the perception of streams of relatively continuous stimuli, and the extraction and recognition of relatively discrete semantic features. Aural examples include the recognition of distinct phonological features in spoken language, which leads to the problem of speech recognition. Visual examples include the recognition of spatial regions in our immediate environment that are indicative of individual substances. This enables us to recognize distinct objects in our environment (such as a computer on a tabletop) and to begin to reason about our immediate spatial context. The idea that semantic cognition involves the "quantizing" of continuous streams of sensory input was proposed at least as early as Cherry (1957), and the theory that the senses are perceptual systems devoted to the recognition of persistent real-world objects was most famously expounded by Gibson (1966). The interplay between geometric spaces and semantic concepts has received recent attention in the work of Gärdenfors (2000), Widdows (2004), and others.

This paper describes an adaptation of these principles to the challenge of creating a large-scale, distributed spatial information system. Given the in-

creasing availability and compatibility of geospatial data from many sources, it is feasible to imagine hugely detailed models of the world in information space Gelertner (1992). At the same time, it is clear that most information devices in a pervasive mobile network will only house small fragments of such a comprehensive model. GIS workstations on stable, highly connected networks, will continue to be used in many projects, but it has already become clear that such a standalone conception of geographic data processing will become but one part of an approach that enables "GIS for Everyone" Lucas (2003). As part of this program, we have developed an interactive Geobrowser, based upon a scalable peer-to-peer information architecture called the VIA Repository.

The main contribution of this paper is to describe research and implementation of a distributed spatial information model called the "Civium World Model", and the data extraction and indexing tools that are necessary to create and use the model in a pervasive network. As a particular example, we outline the steps used to create a model that can execute an optimized and economical "What region am I in?" point-in-polygon query for any part of the world in a peer-to-peer network.

The creation of the model is as part of an ongoing research agenda whose goal is the creation of a global Information Commons network. It turns out that these ambitious but comparatively simple goals implicate the development of sophisticated tools for distributed data extraction, collaboration, and classification. Techniques and systems described in this paper include:

- The VIA Repository network, a peer-to-peer architecture for universal information sharing and collaboration.
- The representation of interrelated geometric objects in such an architecture. The representation needs to use recursive properties for the inclusion of ever more detailed data in certain specific areas, with appropriate distribution of data gathering and authority.
- Automatic and user-guided tools for the assimilation and rationalization of incoming geospatial data. These involve the recognition of significant topological features, the aid the transition from geometric to semantic information.
- Rapid stream-based compression and spatial indexing algorithms for classification and pattern extraction.

- Usage of this model to compute an effective "Where am I?" query that maps continuous measurement data to one of a discrete set of modelled outcomes.

Part of our interest in this problem is as a particularly well-understood form of the semantic mapping problem, where a new datum is mapped to one of a set of predefined or learned outcomes. Examples of such problems include word sense disambiguation Stevenson (2003) and discrimination Schütze (1998). In GIS, deciding which geometric subdivision a particular point is inside is a related problem. One of the simplifications that makes the geometric situation more tractable than linguistic cases is that our 'ontology' of known things in the world that can be chosen as outcomes is well-defined and non-overlapping.

Solving these challenges will lead to great improvements in the richness of information available in tactical ad hoc networks. By creating an information architecture that small devices can use, we hope to dramatically improve the effectiveness of information gathering and knowledge extraction in pervasive applications. As well as improving critical information infrastructure for application domains such as military collaboration and disaster management Jie Xu (2006), we hope to encourage the development of personal systems that behave intelligently in situations with rich spatial context and limited network resources.

2 U-forms and the VIA Repository Network

The spatial information system we describe in this paper is layered on top of a "universal database" architecture that has been developed in our research program over several years Lucas and Senn (2002); Lucas et al. (2005).

The basic abstraction of the system is an abstract data type called the *u-form*. A u-form is simply a bundle of name-value pairs associated with a universally-unique identifier (UUID).

U-forms have the following properties:

- A *UUID* is a sequence of bytes that, within acceptable engineering tolerances, is assumed to be unique in the Universe.
- In addition to the UUID, a u-form comprises a set of attribute name/value pairs.

$\sim$ 013a64ab30588c11d6b09a4cf30b756185

`name`	Pittsburgh
`country`	US
`state`	Pennsylvania
`latitude`	40.44
`longitude`	-79.996

Table 1: Simple u-form representing the city of Pittsburgh, showing the UUID and a few attributes and values.

- Each attribute name/value pair forms a 2-tuple comprising a single attribute name and a single value.
- Each attribute name is a text string of arbitrary length. It must be unique within the u-form.
- Each value is a sequence of bytes of arbitrary length.

The above is the entirety of the definition of the u-form datatype. A basic example of u-form could be the representation of the city of Pittsburgh given in Table 1.

In practice, the values of the `country` and `state` attributes are *relations*, i.e., UUID references to other u-forms, from which the name would be drawn to create a user-readable rendering of the Pittsburgh u-form. For performance and reliability, values from such related u-forms can be cached as intrinsic attributes on the Pittsburgh u-form (e.g., as `country_name`), leading to standard design tradeoffs (space for time, quality of service for quality of data).

U-forms were invented as part of the Visage projects Roth et al. (1996), and the use of u-forms as a universal datatype is known as the Visage Information Architecture (VIA). A store of u-forms is known as a VIA repository, and many VIA repositories are linked together in the peer-to-peer VIA repository network.

The design of u-forms was influenced by many mounting demands on modern information systems. Because the world is evolving and many information systems need to interoperate, recent years have stressed the need for extensible data representations (for example, flexible systems such as

XML have gained ground over traditional relational databases with fixed schemata). The use of uniquely-identified bundles of key-value pairs is, we believe, in the process of becoming a universally accepted data format. Other systems such as subject maps Park (2006) and topic maps Pepper (2000) can be viewed from this point of view, as can many of the clearer aspects of the Semantic Web Manola and Miller (2004) (we have deliberately avoided introducing any system wide ontology or type-system for u-forms). The need for universal information identifiers is recognized by the Universal Resource Locators of the Web. However, one of the main drawbacks of the URL system is that it is designed for a centralized client-server model of computing, which is known to impose increasing scalability problems in the pervasive computing era.

The design of u-forms fully embraces the need for location-independent object identifiers, in a similar spirit to that proposed by Leach et al. (2004) as an IETF standard. Location independent identifiers enable optimistic replication of u-forms to any location where they are useful. For many commonly used u-forms, this is hugely important for improving quality of service, especially for disconnected or sporadically connected devices. For example, the u-form for Pittsburgh described in Table 1 may one day be replicated to thousands of cellphones of people who live in or are visiting the city.

At the same time, having many copies of a u-form imposes the challenge of maintaining quality of data, since changes to a u-form need to be propagated to all replicas of that u-form. Much of our recent research has been devoted to this problem. At the most basic repository storage level, locating u-forms and trying to keep copies of u-forms up-to-date with one another is handled by artificial agents called *shepherds*. In our current system, shepherds will simply mark concurrently updated u-forms as conflicted, rather than trying to resolve conflicts. For building collaborative applications such as the collaborative GIS system designed in this paper, considerable efforts have been made in applications to enable users to collaborate fluently without needing to modify the same u-forms at the same time Higgins et al. (2006a). Other research has been devoted to providing rich indexing and query operations by building abstract data structures out of u-forms, using annotated relations Higgins et al. (2006b).

3 Geometric Data in U-forms

This section discusses the model we use to represent spatial information in u-forms. It describes at the same time an information architecture that is used by consumers of shape data, and a target architecture for the import and semantic extraction tools to aim towards.

There is considerable deliberate semantic reuse in the way we use physical objects to describe and demarcate parts of the Earth. For example, the Rio Grande begins in the San Juan Mountains of southern Colorado and follows 1,885-mile course before it empties into the Gulf of Mexico. For approximately two-thirds of its course, the river also forms the border between the United States of America and Mexico, and (consequently) also forms the border between the state of Texas to the north or the river, and the states of Chihuahua, Coahuila, Nuevo Leon, and Tamaulipas to the south of the river. It follows that the shape of the Rio Grande is part of the shape of the USA, Mexico, and a total of five states within these countries. Some political boundaries were declared to be synonymous with physical features before the physical features were even surveryed: for example, the border between France and Spain was agreed to be "the crest of the Pyrenees" before much of this area was accurately mapped.

Each of the countries and states mentioned above are represented by u-forms. As part of the Information Commons effort, we have collected and fused over 5 million geographic entities representing populated places and political divisions, in a resource called the Information Commons Gazetteer Lucas et al. (2006). Each of these is represented by a u-form, and the relations between these entities forms a complex and valuable information network. Many of these u-forms also have shapes. Source data for such shapes is available, for example, in the VMAP0 and more detailed datasets published by the US National Imagery and Mapping Agency and others. [1] In addition to populated places and shapes of political entities, global data on world shorelines is readily available, for example, in the Global Self-consistent, Hierarchical, High-resolution Shoreline Database (GSHHS) Wessel and Smith (1996). Since the coastlines in the GSHHS dataset are also borders of countries,[2] but the GSHHS data is much more detailed, we would ideally build a representation for which the coastlines of the countries (geopolitical shapes) as well as the continents and islands (geophysical

[1] See e.g., `http://earth-info.nga.mil/publications/vmap0.html` and others.

[2] In keeping with all other political maps, we do not draw the edge of territorial waters as political boundaries, but instead, the shape of a country ends at the water's edge.

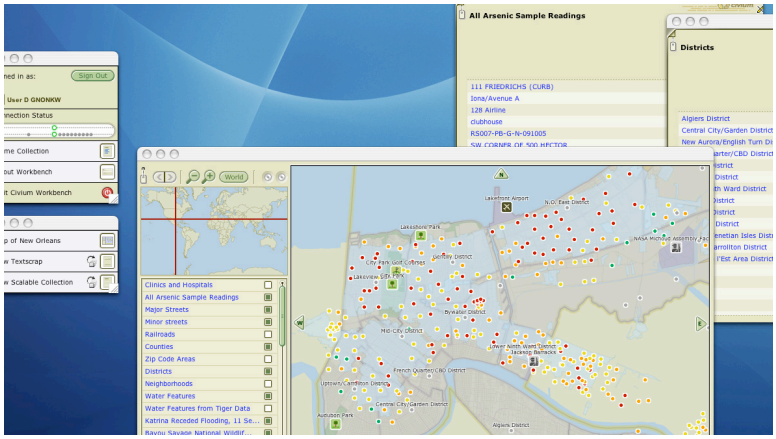

Figure 1: The Geobrowser, data from the Information Commons and the Greater New Orleans Non-Profit Knowledge works

shapes) all make use of the best available data.

Naturally, a list of vector points can easily be expressed as an attribute of a u-form, so geometric objects can be represented in u-forms. The design challenge in this domain is to do this in such a way that shapes can be suitably factored and composed to present a persistent and appealing user interface in a peer-to-peer network. An example of the Geobrowser interface (as used for rendering a variety of datasets during the New Orleans recovery effort) is shown in Figure 3. Basic requirements of the Geobrowser map interface include:

1. It should be possible to drag out a spatial object into its own frame, add it to a collection, and add comments, just as with other phenomena in Visage interfaces.

2. The user should see a basic outline map of the area of interest as soon as possible.

3. More detailed shape data should be rendered as the shape u-forms become available.

4. The user should be able to zoom in to obtain greater levels of detail in specific areas.

Requirement 1 is supported by representing the shape of each object in a Cartesian coordinate frame that is optimized for the object in question. (For example, with shapes on the earth's surface such as boundaries of landmasses and political subdivisions, the approximate centroid is located, and then the local east, local north, and outward normal are used as orthonormal x-, y- and z-axes.)

Requirements 2, 3 and 4 are met in the following way. A simplified outline shape is calculated using a line simplification algorithm which will be discussed in a later section. This outline shape then stores relations to more detailed line segments, along with information saying which points in the main shape should be replaced by the extra detail in the subshapes. An example of this approach to shape rendering is shown in Figure 2, which depicts the Supercontinent (Asian, African and European landmasses) with both the first level (roughest) and second level (slightly more detailed) decomposition. The first level points are contained in a single u-form, whereas the second level points are broken across 10 different u-forms of roughly similar length. In Figure 2, both levels are depicted together for explanatory purposes, though in the actual Geobrowser, the second level data actually replaces the first so that there is a single coastline. The benefit is that the user only needs to shepherd in the u-forms containing more detailed information for the parts of the world that are of interest. Once accessed, these local detail u-forms stay available the user for as long as they are wanted.

To support all of this functionality, shapes must be able to include one another recursively. Sometimes shapes are included without replacement (for example, adding an island to a continent in requirement 1), and sometimes subshapes replace parts of their parent shapes (meeting requirement 3). In both cases, a parent shape creates an annotated collection of child shapes, for each child shape giving:

1. The bounding box of the child shape in the parent's coordinate frame; and
2. The linear transformation used to map the child points into the parent frame.

In this way, shape u-forms act as containers for lists of geometric points, and as an index to subsidiary shapes. These subsidiary shapes may be topologically distinct, or may be more detailed parts of the main shape

Figure 2: Shape of the Supercontinent (orthogonal projection), showing first and second levels of detail combined.

itself. In this way, a shape u-form acts as an index to its own parts, and is described as a *self-indexing* structure.

A rendering algorithm proceeds by:

1. Testing the bounding boxes in turn to see whether the shape intersects with the user's field of view; and
2. Retrieving the child points, and using the linear transformation given to map the points to the parent frame, from which they are mapped to screen coordinates.

These mappings are all calculated using standard linear algebra as used in many computer graphics algorithms (see e.g., (Foley et al., 1990, §5.6)), and can be optimized significantly by hand-coding the floating point operations for our specific use case.

Our model is designed to give a more sophisticated GIS experience by taking into account the semantics of the way the world is described by humans. Once the model has been built, this gives an extremely effective way of sharing the model between many devices in the network, making maximum reuse of the best data from different publishers, and presenting an effective user interface that continues to work when the user is offline.

4 Geometry, Topology and Semantics

This section describes techniques for building the spatial model descibed in the previous section. As far as possible, out shape import tools try to make automatic judgements about the semantic nature of the shapes being added to the model. Sometimes user intervention is necessary, and here, our goal is to enable users to give as much guidance as possible to the geometric algorithms, with a minimum of effort and making maximum use of human intuition. It turns out that these goals can be met by focussing on the *topology* of the shapes. Once this is made clear to the system, the heavy computational geometry can be preformed reliably and effectively.

We begin with the following observation. A small child can be shown a map of the countries of the world, and can be asked to trace their finger round (say) the coastline of a large continent. The child will do this with ease. On the other hand, if we try to write a computer program which takes a large dataset of geographic shapes and outputs their external boundary,

we quickly find ourselves doing large amounts of computation, and often get brittle and unsatisfactory results. In a similar fashion, people tend to notice topological errors much more than merely geometric errors. (The importance of preserving topological consistency in shape simplification is also recognized by Wu and Marquez (2003).) If the shape of Great Britain is incorrect on a map, we may not notice. If the shape of Great Britain touches the shape of France, and therefore ceases to be an island, we know that something is wrong straight away. It turns out to be difficult to instill computers with such intuition.[3]

The problem of deciding which shapes are shared between different phenomena is greatly simplified in the case of subdivisions of a 2-dimensional space. By subdivisions, topologically speaking, we mean sure partitions into continuous components. Geometrically, these structures are more commonly referred to as 'tessellations.' They have particularly predictable properties which introduce guaranteed economy of description. One famous example of this is in the *four colour problem*: it was proposed in the 19th century that subdivisions in any political map could be drawn using only four different colours, in such a way that no two adjacent regions share the same colour, and this theorem was demonstrated (controversially) in recent years, as described by Wilson (2003).

The specific way we take advantage of the geometry of 2-dimensional subdivisions is al follows. Most (almost all) points in a subdivided space are in one and only one subdivision. (Points properly contained in a single polygon.) The remainder are border points. These include points on the boundary of two or more subdivisions, or points on the external boundary of the system under consideration. (For example, when subdividing the Earth one may think of the ocean as the "external boundary" or the system of political country shapes; or one may regard it as a subdivision in its own right, since both of these approaches leads to identical conclusions for the topological structure of the countries tehmselves.) Of all of the border points, only a very few touch three or more subdivisions. Examples of 4 subdivisions touching are so rare that they are sometimes noted tourist spots, e.g., the four corners monument where the states of New Mexico, Colorado, Arizona, and Utah meet.

These topological properties are what enables us to solve the border sharing problem. We identify points where three or more subdivions meet,

[3] At least, not without considerable optimization and simplification, which we are still investigating.

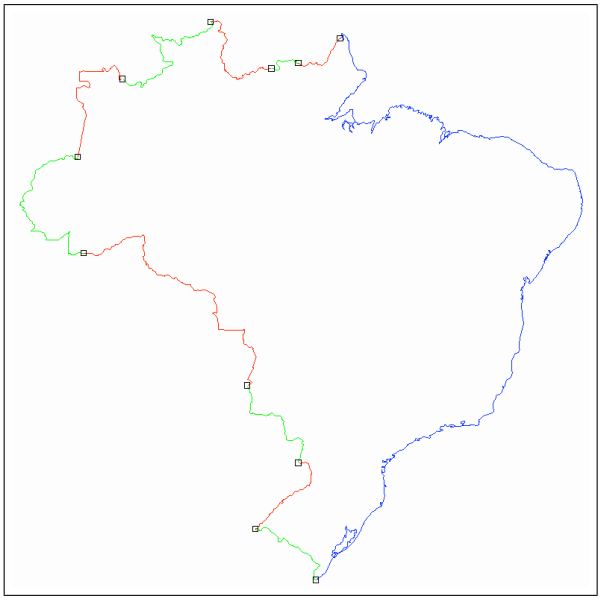

Figure 3: Shape of Brazil, showing points of topological interest (tripoints)

which are often called "tripoints". Shapes can then be decomposed into segments on each boundary that lie between these points. For example, the border of Brazil is depicted in Figure 3. This shows each of the points where three countries meet. These points therefore partition the border of Brazil into the border with the Atlantic Ocean, the border with Uruguay, the border with Argentina, etc. Understanding this structure is one of the keys to enabling border-sharing in GIS applications.

Those familiar with graph theory will find this approach particularly familiar. In the graph of county shapes, each border segment is a link between two neighbouring country nodes. In the tripoint representation, each border segment is a link joining two tropoints. These two points of view are *dual* representations to one another. In the first, the subdivisions are the nodes, the borders are edges, and the tripoints are faces. In the second, the tripoints are the nodes, the borders are still edges, and the subdivisions themselves are faces. These are topologically equivalent descriptions, and the whole model preserves toloplogical invariants such as the Euler characteristic (e.g., for countries sharing a single continent, the number of faces plues the number of vertices minus the number of edges will always equal one).

In particularly well managed datasets, this structures is already implicit. For example, in the VMAP0 dataset, the points on the boundaries of two countries have exactly the same latitude and longitudes in both shapes, whichever side of the border they are representing. This enables us to calculate graphs like the one in Figure 3 by simply analyzing the valence (total number of appearances) of each point over the dataset as a whole. The combinatoric details of this process are somewhat detailed in practice, but the general principles are quite simple, as follows:

- Points with a valence of three or more throughout the entire dataset are tripoints, that is, points of topological interest where three or more subdivisions meet.
- Points with a valence of one in the entire dataset are part of the external boundary of the total space, e.g., coastlines.
- Points with a valence of two in the entire dataset are usually on internal boundaries, unless their neighbouring points have valance one, in which case they are tripoints involving two subdivisions and the external space.

Of course, in most real life cases, such combinatoric techniques are not reliable due to measurement errors and the fact that data is collected and contributed by a number of independent publishers. However, the combinatoric principles arising from the surface topology are still extremely instructive. The key is still the identification of tripoints and the segments in between them, and when this cannot be done purely symbolically, it can be accomplished successfully using statistical methods or with user intervention. Due to space limitations, we do not discuss these algorithms and interfaces at length here, but details are available from the authors upon request.

An interesting practical consequence of this topological data inference was that the system "learned" that there are only six land masses on the Earth as represented by the VMAP0 dataset that are divided between more than two countries. These are the Supercontinent (mainland Asian / African / Europe), the American Continent, and the islands of Bornea, New Guinea, Hispaniola, and Ireland. (This list does not include Cyprus or Timor.)

5 Stream-Based, Linear Time Shape Simplification or *Cylinder Simplification*

Once the points of key topological interest have been identified, it is then possible to compute reduced representations of the spatial system being modelled in such a way that the significant features of the model are always preserved in the user interface. In keeping with our research goal of enabling fully distributed information and computation, the computation of the model to be represented in u-forms is performed using fully encapsulated components called *Infotrons*. This is part of a research methodology called *IDA*, which stands for *Information Devices Architecture*. Though less well documented than u-forms and the VIA Repository, IDA is based upon similar principles. Information processing machines should be created out of uniquely identified building blocks that can be instantiated on any device that has the necessary computational resources. This leads to a dataflow architecture in which infotrons communicate with one another asynchronously by sending discrete messages over specified channels. The code ("blueprint") used to create an infotron is represented in a u-form and disseminated around the network using the shepherds system. In this was, new algorithms can be created and widely used thoughout the network without developers ever needing to manually install new packages, set new path names, etc. We believe that such a design methodology is not only promising for code reuse, but also deliberately forces developers to be conscious of the computational resources their algorithms use, which is important if we are to systematically enfranchise smaller and smaller disconnected devices.

One important consequence of this approach is that it has encouraged us to develop a linear-time, steam-based algorithm for shape simplification. (This is probably the most significant direct contribution to computational geometry introduced in this paper.) The traditionally accepted method for shape simplification is the Douglas-Peucker simplification Douglas and Peucker (1973). Douglas-Peucker simplification of a polyline proceeds as follows. For each point in the polyline, its perpendicular distance to the straight line joining the two endpoints of the polyline is calculated. The "most significant point" is the internal point that maximizes this distance, and this point is added to the simplified representation. This divides the polyline into two segments, and the algorithm descends recursively. This process can be terminated either until a specified number of points has been selected, or until the maximum perpendicular distance is less than a given

simplification tolerance ε. The Douglas-Peucker algorithm usually runs in quadratic time, and due to the recursion, it can have significant memory requirements for maintaining a large stack.

After careful consideration, we decided that the ε-tolerance style of simplification is more appropriate for our needs than the n-best points style. This is for two reasons:

- There is a clear relationship between map resolution and tolerance. Information designers can use the idea of splification tolerance to determine optimal views at certain resolutions.
- It makes border sharing more reliable between shapes with different numbers of points. For example, if choosing 50 best points to represent the shape of Spain, and 50 best points to represent the shape of Portugal, the Portuguese representation will usually have a greater point density on the common border, because the common border is a greater proportion of the shape of Portugal than the shape of Spain. If instead, we use a fixed simplification tolerance ε for both shapes, we guarantee that if the unsimplified borders line up, the simplified borders will also line up.

We have prototyped and tested an algorithm that performs line ε-tolerance based line simplification in linear time with minimal space requirements. The algorithm is based upon the following intuition. Since we know ε in advance, we can contruct a notional cylinder of radius ε, and try to fit this cylinder over the shape to be simplified. As we stuff more points into the cylinder, the freedom of the cylinder to move around reduces, until eventually the cylinder gets stuck. When this happens, we "break off" the points contained in the cylinder as a new shape segment, and continue from where the break occurred. Instead of requiring a large stack to support the recursion, we require only a small heap to keep track of the points that are nearest to the walls of the current cylinder.

The algorithm is particularly simple in 2-dimensions, since the boundary of the notional cylinder is just a pair of parallel lines, and the heap only has to maintain the point most likely to escape 'from below', and the point most likely to escape 'from above'. Pseudocode for the 'from below' part of the algorithm is as follows:

```
# check to see if we've broken the bounds
if distance( line(current_point, start_point),
                          lo_bound) > tolerance:
    add_to_output(current_point)
    (optional) add_to_output(lo_bound)
make_new_segment(start_point = current_point)

# check to see if this point should be a new lo_bound
intercept = y_intercept( line(lo_bound, current_point),
                                          start_point)
if defined(lo_bound) and (intercept < 0):
if distance( line(lo_bound, current_point),
                     start_point, ) < tolerance:
    lo_bound = current_point
```

A comparison of traditional recursive Douglas-Peucker line simplification and linear cylinder simplification is shown in Figure 4. As expected, the cylinder algorithm is significantly faster (running time less than half), even on a comparatively small file. The recursive algorithm guarantess an optimal solution to the problem, and produces a correct simplification to a tolerance of 150km using only 45 points. The cylinder simplifaction algorithm produces slightly more, 51 points, in making the same guarantee. This gives information designers useful tradeoffs to consider — a more optimal output can be achieved at higher computational cost.

We note in passing that such tradeoffs are typical when considering compression algorithms, and shape simplification can be seen as a kind of compression. We believe that cylinder simplification has some typically useful properties, in particular, depending only on the local region it is considering, and never needing to iterate over the whole shape. This enables one-the-fly simplification, as used in other domains (e.g., it would be strange to design a video compression algorithm that relied on finding the frame that deviated most significantly from the expected trajectory from the first frame to the last).

The algorithm we have used is so far 2-dimensional, 3-dimensional robustness being guaranteed simply by padding the ε-tolerance enough to make sure that the less-varying z-coordinate does not introduce too much extra deviation. We believe it is possible to create a much more correct n-dimensional version by considering the set of cones in n-space that pass through a given point, ordered by containment. This has been left for future research, though it poses some very interesting pure mathematical questions.

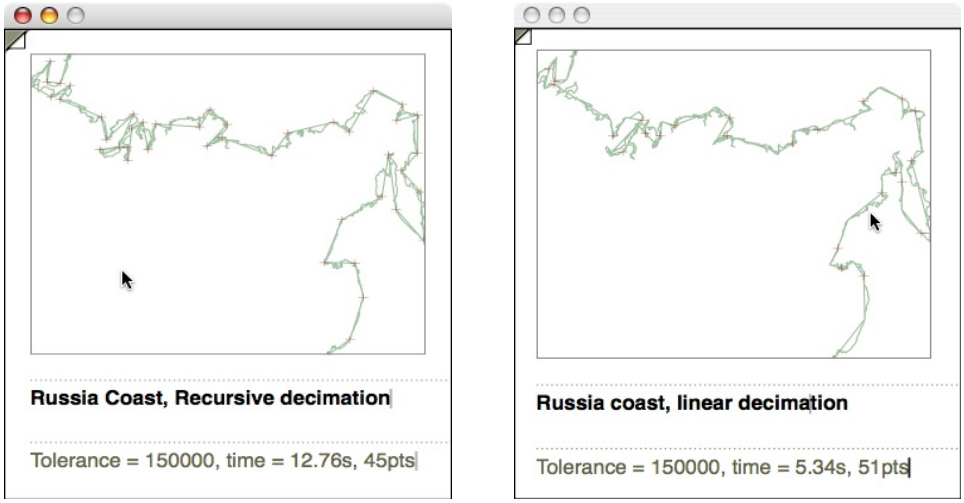

Figure 4: Comparison of the output and running time of line simplification algorithms

6 Where Am I? Optimized Location Quantization

We are finally in a position to address an interesting "semantic quantization" problem, as suggested in the introduction to this paper. Given a subdivision of Earth's surface, and the coordinates of a point on this surface, which subdivision contains the point? This is a very pure example of semantic quantization — the input is a continuous measurement (given, for example, by a GPS unit). The output is one of a discrete list of subdivisions such as countries. In the sense of quantum logic (Widdows, 2004, Ch 7), the countries are the pure states of the system, and the quantization problem is to select the correct pure state from a given measurement of coordinates.

The most naive approach would be to run a point-in-polygon test for each of the subdivisions. This would require large I/O and much computation, since conducting a Boolean point-in-polygon test is traditionally an $O(n)$ operation, or at best, $O(\log(n))$ for convex polygons Haines (1994), where n is the number of vertices in the polygon. Recommended optimizations for point in polygon tests involve (for example) dividing the line segments into quadrants and finding integer versions of the necessary arithmetic Hormann and Agathos (2001).

For our problem, a better approach is to begin by filtering out all countries whose bounding box does not contain the point. Better still, one would use a spatial index such as an R-tree Guttman (1984) to retrieve the nearby bounding boxes before beginning to filter. Such a process can be effectively distributed in u-forms so that the index (and the dataset of coutries) does not need to be contained in any single venue Higgins et al. (2006b).

By combining such a distributed R-tree index with the factored shape representation presented in this paper, we can do still better. Using standard $O(n)$ methods, it is easy to write an algorithm that (i.) tests whether a test point is inside or outside the simplified polygon, and (ii.) measures the minimum distance from the test point to the simplified polygon. If this distance is greater than ε, then the result also holds for the fully detailed polygon. Otherwise, the correct answer is currently too close to call, and u-forms containing more detailed points for the nearby segments are requested and the test is repeated.

While this algorithm produces appreciably faster results, formal analysis of this algorithm's complexity is complicated, because it depends to a large extent on the choice of appropriate levels of simplification, which itself depends on number of points, point density, average curvature, and the performance / detail tradeoff. In general, we are confident that many of the designs presented here will be of great use in future years as demands and user expectations increase. It should be perfectly possible to get off an aeroplane anywhere in the world, and have your GPS-enabled cellphone immediately register not only your new latitude and longitude, but also what country, province, city, and even census block you have arrived in. This must be accomplished without the cellphone having to store information about other areas of similar detail, without it needing to obtain the most detailed geographic data available (since this may be very large), but at the same time, giving the user enough local geographic detail to draw useful sketch maps and give directions. Our design makes all of this possible in a persistent, reliable fashion.

More generally, these issues are typical of those that will arise as computational devices learn to behave more like conscious cognitive agents. Logical agents should behave in robust ways given naturally limited resources. Finite resources include computational power and time, and also the data required to solve a problem Gabbay and Woods (2001). It is especially important with distributed information systems to design solutions that can answer a user's questions with the smallest possible bandwidth requirements and the maximum flexibility in terms of *where* computation is per-

formed. One should try to solve the problem at hand with available information, instead of asking for complete information before beginning to look for a solution.

7 Conclusion

We have presented a comprehensive design for the creation and maintenance of a geospatial model that makes deliberate reuse of geometric data. This is motivated by semantic considerations, because in many cases we know by definition that parts of different shapes should coincide. Topological and combinatoric models can be used as a fertile middle-ground to bridge between the semantic and the geometric representations, and sometimes the semantics implied by a dataset of geometric information can be inferred automatically, or failing that, with minimal supervision. Using such a model, one can create an extremely effective distributed geospatial information system, that is both resource-conscious and robust in the face of poor connectivity. Our research project is currently building the tools necessary to make this model widely available, and we believe that our research makes effective use of many techniques that will become invaluable as information becomes better integrated and computing devices become further disseminated.

References

Cherry, C. (1957). *On Human Communication*. MIT Press.

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

Foley, J. D., van Dam, A., Feiner, S. K., and Hughes, J. F. (1990). *Computer Graphics*. Addison Wesley.

Gabbay, D. M. and Woods, J. (2001). The New Logic. *Journal of the Interest Group in Pure and Applied Logics*, 9 (2):157–190.

Gärdenfors, P. (2000). *Conceptual Spaces: The Geometry of Thought*. Bradford Books MIT Press.

Gelertner, D. (1992). *Mirror Worlds: Or the Day Software Puts the Universe in a Shoebox...How It Will Happen and What It Will Mean*. Oxford University Press.

Gibson, J. J. (1966). *The Senses Considered as Perceptual Systems*. Houghton Mifflin, Boston.

Guttman, A. (1984). R-trees: a dynamic index structure for spatial searching. In *Proceedings of SIGMOD*, pages 45–47.

Haines, E. (1994). Point in polygon strategies. In Heckbert, P., editor, *Graphics Gems*, volume IV, pages 24–46. Academic Press.

Higgins, M., Roth, S., Senn, J., Lucas, P., and Widdows, D. (2006a). Managing distributed collaboration in a peer-to-peer network. 14th International Conference on Cooperative Information Systems (CoopIS 2006).

Higgins, M., Widdows, D., Balasubramanya, M., Lucas, P., and Holstius, D. (2006b). Shepherdable indexes and persistent search services for mobile users. In *8th International Symposium on Distributed Objects and Applications (DOA 2006)*, Montpellier, France.

Hormann, K. and Agathos, A. (2001). The point in polygon problem for arbitrary polygons. *Computational Geometry*, 20(3):131–144.

Jie Xu, A. L. (2006). Replication-aware query processing in large-scale distributed information systems.

Leach, P., Mealling, M., and R.Salz (2004). A UUID URN namespace. Technical report, The Internet Society. Current draft, awaiting approval.

Lucas, P. (2003). Civium: A geographic information system for everyone, the Information Commons, and the Universal Database. In *Vision Plus 10*, Lech/Arlberg, Austria.

Lucas, P., Balasubramanya, M., Widdows, D., and Higgins, M. (2006). The Information Commons Gazetteer: A public resource of populated places and worldwide administrative divisions. In *Fifth International Conference on Language Resources and Evaluation (LREC 2006)*, Genoa, Italy.

Lucas, P. and Senn, J. (2002). Toward the Universal Database: U-forms and the VIA Repository. Technical Report MTR02001, MAYA Design.

Lucas, P., Senn, J., and Widdows, D. (2005). Distributed knowledge representation using universal identity and replication. Technical Report MAYA-05007, MAYA Design.

Manola, F. and Miller, E. (2004). RDF primer.

Park, J. (2006). Promiscuous semantic federation: Semantic desktops meet web 2.0. In *Semantic Desktop Workshop at ISWC06*.

Pepper, S. (2000). The TAO of topic maps - finding the way in the age of infoglut.

Roth, S., Lucas, P., Senn, J., Gomberg, C., Burks, M., Stroffolino, P., Kolojejchick, J., and Dunmire, C. (1996). Visage: A user interface environment for exploring information. In *Proceedings of Information Visualization*, pages 3–12, San Francisco. IEEE.

Schütze, H. (1998). Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–124.

Stevenson, M. (2003). *Word Sense Disambiguation: The Case for Combining Knowledge Sources*. CSLI Publications, Stanford, CA.

Wessel, P. and Smith, W. H. F. (1996). A global self-consistent, hierarchical, high-resolution shoreline database. *J. Geophys. Res.*, 101(B4):8741–8743.

Widdows, D. (2004). *Geometry and Meaning*. CSLI publications, Stanford, California.

Wilson, R. (2003). *Four Colors Suffice: How the Map Problem Was Solved*. Princeton University Press.

Wu, S.-T. and Marquez, M. (2003). A non-self-intersection Douglas-Peucker algorithm. In *Computer Graphics and Image Processing (SIBGRAPI 2003)*, pages 60–66.