Route Visualization using Detail Lenses

Pushpak Karnick, *Student Member, IEEE,* David Cline, Stefan Jeschke, Anshuman Razdan, *Member, IEEE,* and Peter Wonka, *Member, IEEE,*

Abstract—We present a method designed to address some limitations of typical route map displays of driving directions. The main goal of our system is to generate a printable version of a route map that shows the overview and detail views of the route within a single, consistent visual frame. Our proposed visualization provides a more intuitive spatial context than a simple list of turns. We present a novel multi-focus technique to achieve this goal, where the foci are defined by points-of-interest (POI) along the route. A detail lens that encapsulates the POI at a finer geospatial scale is created for each focus. The lenses are laid out on the map to avoid occlusion with the route and each other, and to optimally utilize the free space around the route. We define a set of layout metrics to evaluate the quality of a lens layout for a given route map visualization. We compare standard lens layout methods to our proposed method and demonstrate the effectiveness of our method in generating aesthetically pleasing layouts. Finally, we perform a user study to evaluate the effectiveness of our layout choices.

Index Terms—Route Visualization, Map Visualization, Overview and Detail Techniques

1 Introduction

A *route map* is a special purpose map that displays a travel path along with directions for following it (see Fig. 1). The path itself may describe a simple cross-town trip or a journey spanning multiple countries. Numerous systems exist that can provide turn by turn directions for following a route. These can roughly be divided into two categories: those that provide a real-time animated view of the path during travel, and those that create static route maps. This paper addresses the problem of creating static route maps suitable for print media.

Typically, a route map provides an overview of the complete route along with textual directions for following it. This type of route map is fairly effective, but not necessarily ideal. Consider the scenario of a traveler printing a route map before starting a trip. It is unlikely that the traveler will be willing to memorize (or even read) the route directions before starting the trip. At the same time, the directions will be difficult to read while driving. This is partly because a list of textual directions contains few visual cues and the traveler must scan the list visually to find the current location and its corresponding directions for every such query.

In this paper we address the route map problem by providing detail lenses along the route that mirror the information contained in the textual directions. We start with a typical route visualization that shows a high level map view of the route along with driving directions in a second window. To this we add path highlighting and detail lenses for points of interest (POI) along the path. These detail views provide a local context for each decision that must be made while navigating a route. In other words, the spatial layout of the lenses on the page provides a natural visual indexing system for the route directions.

The main contributions of this paper are:

- A web-based automated system for generation and layout of detail lenses over a route map.
- A two-step optimization algorithm for placing the detail lenses on the map with the help of extensible layout metrics.

We demonstrate with the results of a user study that our layout strategy produces aesthetically pleasing and visually coherent lens layouts that have less visual clutter than other layout strategies, and that it forms an effective means of route visualization. In the user study, we evaluate our design choices, as well as compare our method with related method of inset placement on maps.

- *P. Karnick is with the Department of Computer Science, Arizona State University, Tempe. Email:pushpak@asu.edu*
- *D. Cline is with the Department of Computer Science, Arizona State University, Tempe. Email: clinedav@gmail.com*
- *S. Jeschke contributed to this paper while working at Arizona State University, Tempe. Email: jeschke@cg.tuwien.ac.at*
- *A. Razdan is with the Department of Engineering, Arizona State University at Polytechnic Campus, Mesa. Email: ar@asu.edu*
- *P. Wonka is with the Department of Computer Science, Arizona State University, Tempe. Email:pwonka@gmail.com*

2 Related Work

Route Maps are one of the most common forms of geospatial data in use [5]. Route visualization is related to a number of topics, including cartographic methods for map labeling, human cognition, overview+detail methods, and route mapping. This section describes related work in these areas.

The problem of placing detail lenses on a map can be thought of as a special case of map labeling. General map labeling attempts to place a number of feature labels on a static map while avoiding overlap and adhering to prescribed aesthetic criteria (see [18], [17], [4], [15], [19] for details).

Because of the computational difficulty of optimal label placement in the general case, most approaches use heuristics such as simulated annealing and gradient descent [14], [15].

Although the output of our method is a static route map, our work is related to a number of interactive labeling algorithms. Daiches and Yap [8] address the problem of placing labels on a pannable and zoomable map. The method works by creating static label placements and assigning priorities to each of the labels. During interaction, the labels are displayed in order according to priority, omitting labels that overlap with labels of higher priority.

Other interactive labeling systems attempt to handle more complicated changes than panning and zooming. Götzelmann, Hartmann, and Strothotte [25] present a real-time algorithm to place both internal and external labels on 3D visualizations. Their system uses several heuristics with adjustable weights to control the layout, employing special “layout agents” to optimize the layout and maintain coherence between frames.

Fekete and Pliasant [16] generate dynamic “excentric labels” for objects within a user-specified neighborhood. The dynamic labels allow the user to inspect the contents of a small neighborhood without zooming in to it, making interaction more efficient.

Bekos and Kaufmann [9] explore the concept of *boundary labels*, in which labels are placed around a rectangle containing points of interest, and the POIs are associated with their respective labels via connecting lines called *leaders*. This is similar in many ways to our approach, but we consider metrics specific to route map visualization when computing lens layouts.

Also of interest in the context of route visualization is the use of mobile devices such as PDAs and cell phones to display location-specific information [12], [23] and study navigation patterns [13]. These methods serve as a good starting point for our application. Our implementation is web-based, and hence could be ported to handheld devices that support web browsers by incorporating interaction behaviors as suggested in the above papers.

Our proposed approach of providing detail lenses over an overview map is most closely related to the problems of inset placement on maps [11], [22], [7]. While the high-level goal of our proposed approach and the above methods is similar - displaying overview and detail data over a geospatial domain - there are significant differences in the type of data that we propose to work on. The existing approaches are geared towards determining what portions of the map can be better described with a detail view. There may be one or more detailed insets on a single map; however, the individual inset regions may not have an explicit relationship with one another. The placement of these insets is defined by the boundaries of the Regions-of-interest (ROIs) that they are enclosed in. In our case, the entire route is the ROI, marked with specially designated points (POIs) along the route. Detail lenses have to be created for every POI without respect to the POI density in their vicinity. The detail lenses cannot be placed in an arbitrary manner since the POIs have a strict temporal ordering. Thus, detail lenses of nearby POIs should also have the corresponding proximity to one another.

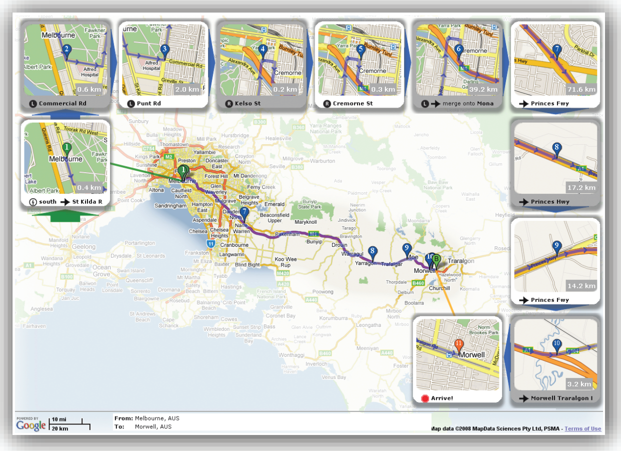


Fig. 1: A route map with detail lenses created using our system.

There is a wealth of previous work for the evaluation of verbal directions and non-verbal symbols in the case of navigation or route maps. The seminal work by Allen [6] defines in detail the subprocesses involved in transfer of route directions between individuals, and their constituent elements (landmarks, segments/pathways, choice points/turns). Lovelace et al. [21] perform a qualitative assessment of route directions using the standard elements mentioned above. Tversky and Lee [26] compare descriptions and compact depictions of route directions with the help of a standardized ‘toolkit’ of common phrases and their corresponding pictorial representations. Their study finds semantic commonalities between route directions that describe a particular route, and corresponding route maps, which show an imprecise sketch of the same route pictorially. Recently, Klippel and Montello [20] explored the conceptualization of route directions employing linguistic as well as non-linguistic categorization.

Agarwala [3], [2] creates route maps that are similar to those that might be drawn by a human. The route is distorted and simplified to highlight important features and make the route map more readable. Our work shares many of the same goals, but we avoid distortion and attempt to provide more near-route context.

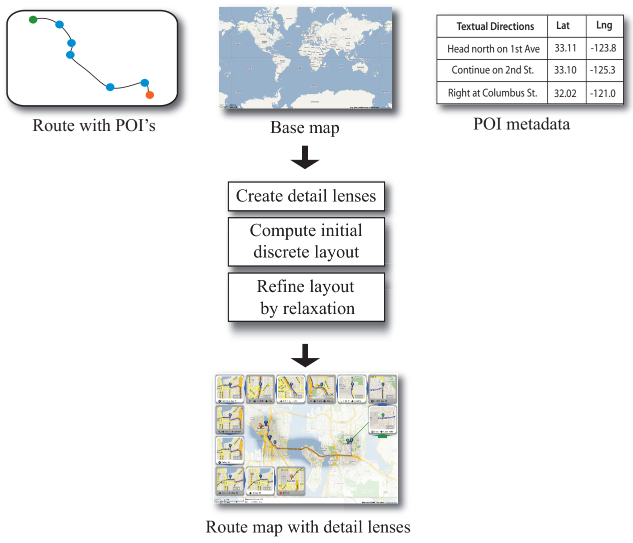
Recently, Google Maps (maps.google.com) has also started including detail lens views for the printable version of their maps. However, these views are placed next to the textual direction list, and further increase the number of pages to be printed. Also, in some cases (for example, a map from Boston, MA to Philadelphia, PA) the source and destination are not visible in the overview at all. Our method addresses both issues by rescaling the map such that it is always visible, and placing the lenses on the same page(s) as the route overview.

3 System Overview

Our route visualization system provides route navigation details in the form of a route map augmented with *detail lenses* for points of interest along the route path. Inputs to the system include a *base map* at multiple geospatial scales, a polyline *route* that the travelever wants to follow and *points of interest* (POI) along the route with textual directions or other *metadata* for each POI (see Fig. 2). In our prototype, the POIs and their associated metadata are provided by Google Maps as part of their route directions.

Given the inputs for a route, the layout engine creates detail lenses for the POIs and places them around the route as navigation aids. Lenses include a close-up of the decision context, along with an abbreviated version of the textual directions. We place the lenses on the map border based on layout metrics that encourage visual continuity between the route and the lenses. The layout metrics include terms for the distance between the POIs and the lenses as well as the relative position of the lenses. This allows our system to produce layouts designed specifically for route visualization rather than simply adopting standard map labeling norms.

Our layout algorithm proceeds in two phases. The first phase computes an optimal discrete layout using a backtracking approach. This is followed by a relaxation step that refines the discrete layout further to produce a continuous layout.


Route with POI's
Base map
POI metadata
Create detail lenses
Compute initial discrete layout
Refine layout by relaxation
Route map with detail lenses

Textual Directions	Lat	Lng
Head north on 1st Ave	33.11	-123.8
Continue on 2nd St.	33.10	-125.3
Right at Columbus St.	32.02	-121.0

Fig. 2: Schematic view of our system. Inputs include a base map, a route path, and POIs along the route with textual directions and other metadata for each POI. The output is a route map with detail lenses around the map border.

4 Detail Lens Design

This section describes the design of individual detail lenses while the following sections discuss how multiple lenses are placed. The basic function of the detail lenses is to provide the same information as the textual directions, while adding supplementary visual context for decisions made along the route. Ideally, the user should be able to recognize the information provided in a lens at a glance; thus, one goal is to provide sufficient information on the lenses to describe the POIs without any ambiguity.

As shown in Fig. 3, each lens in our system corresponds to a single POI, or direction in the instruction list. This design allows us to sidestep issues related to placing multiple directions on a single lens, and it has the benefit that the user never has to determine whether to linger on one lens for multiple route decisions. Typically a detail lens contains a close-up view of the route near the decision point, a number identifying which POI the lens represents, one line of abbreviated textual directions and the distance to the next POI along the route. Placing additional information on the lenses would be possible, but in our experience, this quickly leads to information overload. Even so, the information currently provided is almost always sufficient to follow the route without the need to refer to the direction list.

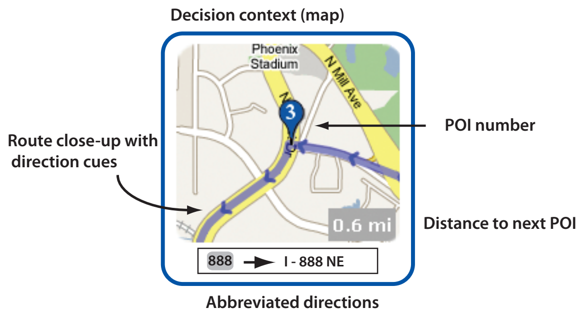


Fig. 3: A route detail lens provides the local context for each decision that should be made while traversing the route.

4.1 Abbreviated Directions

In our implementation, the Google Maps server provides textual directions for each POI along the route, but they are often too long to fit in the limited space on a lens. We attempt to solve this problem by replacing common phrases with the compact symbols shown in Fig. 4. We arrived at the list of common phrases by creating a large number of random routes and manually extracting common phrases from the textual directions. Additional phrases could very easily be incorporated into the system. In order to provide familiarity with real-world traffic symbols, our symbols are based on commonly observed highway symbols [1] where possible, or designed to match the meanings of the phrases that they replace. One advantage of using symbols instead of abbreviating the phrases is understandability by non-English speakers. By establishing a common symbolic code, the maps can be used by non-English speakers without resorting to translation of the text. We present the results of an informal evaluation of our symbols as a part of our user study in Section 8.

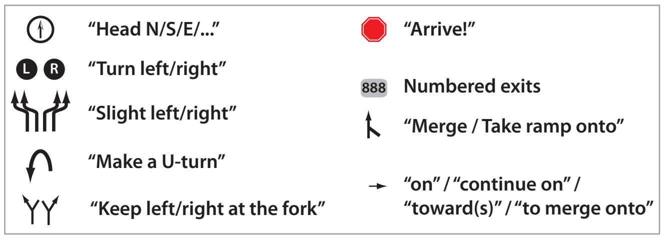


Fig. 4: Symbols used in our system and the phrases that they represent.

4.2 Lens Size

In addition to deciding on the lens content, an important design choice is the size of the lenses. If a lens is too small it does not adequately show the area around the decision point, lens elements become crowded, and there is often not enough space to fit the abbreviated directions. On the other hand, very large lenses take up too much space on a page, and may show extraneous details that are distracting rather than useful. Thus, setting a proper lens size involves a number of tradeoffs.

Some of our early layout attempts used multiple lens sizes on the same page, but this invariably lead to a cluttered appearance. Another idea was to scale all lenses equally so that they would fit on one page. This approach failed for routes with many POIs since the lenses quickly became too small to be useful. In the end we decided to use a fixed lens size, splitting the map into multiple pages if needed. The map scale inside the lenses was also fixed to be at the street level.

We conducted a user study to evaluate the preferred lens size. Users were presented with examples of maps with three lens sizes (a) small (20 lenses on a page), (b) medium (14 lenses on a page) and (c) large (10 lenses on a page). Fig. 5, shows the size of lenses used in our system.

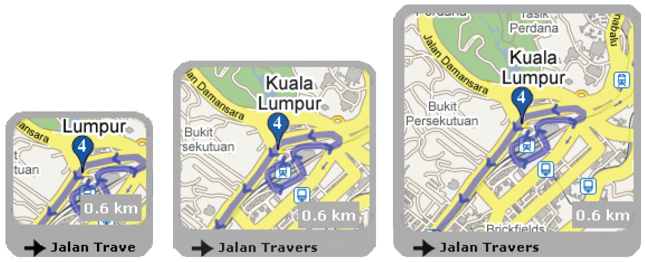


Fig. 5: Example of the lenses used in our system (middle), along with smaller (left) and larger (right) sizes for comparison. All examples are shown at about 60% scale.

5 Layout Principles and Metrics

5.1 Placement Principles

The goal of placing a number of detail lenses on a route map is in some ways more difficult than general map labeling. For example, standard map labeling techiques [18], [17], [15] would suggest placing the lenses as close to their POIs as possible without overlapping the route. This idea seems quite promising, but has a number of drawbacks. First, the search space for such a placement is large, making the algorithm to find an optimal placement slow. Additionally, rather than producing useful layouts as one might expect, the lens placements made by this policy tend to merge visually with the route, making the map look cluttered (see Fig. 6). Finally, since the lenses are not arranged in a particular order, the location of one lens does not naturally lead to the location of its successor. Based on these observations and a number of other attempted layout procedures, we developed a set of principles that good lens layouts should adhere to:

1) Lens placements should be as close as possible to their respective POIs.
2) Lenses should not occlude the route or each-other.
3) Lenses should not be placed too near the route to avoid visual clutter.
4) Consecutive lenses should be placed near one another to provide visual coherence in the layout.
5) The vector (direction) between consecutive lenses should be similar to the vector between their corresponding POIs.

These principles are guidelines for developing an "ideal" layout, but they do not have a precise mathematical definition. In addition, the goals stated in the above rules will often compete with each other for real-world examples. In section 5.2 we define a set of extensible metrics that capture the essence of the above rules in strict mathematical terms. Our metrics help balance the trade-offs between competing rules by the assignment of user-defined weights.

We address rule 3 by placing the lenses along the border of the page, thereby minimizing visual clutter, and address rule 2 by rescaling the route to fit within the border region defined by the lenses. Thus, we only need to derive metrics for rules 1, 4 and 5 to generate a good layout.

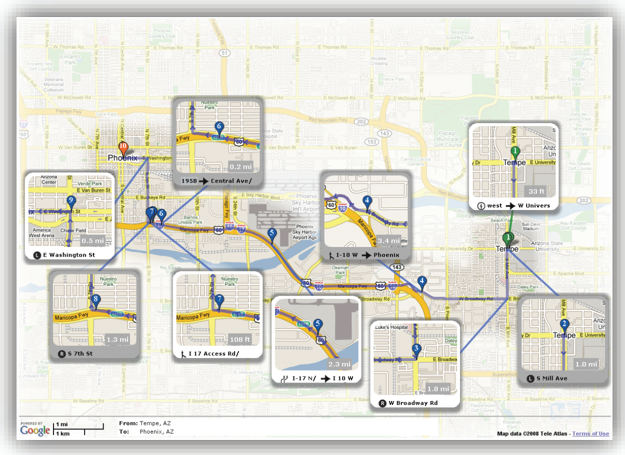


Fig. 6: Placing lenses using standard map labeling techniques results in visual clutter since the lenses tend to merge visually with the route, even if they do not occlude it.

5.2 Layout Metrics

Using the principles just described, we can now define a set of metrics to enable objective comparisons between different lens layout styles. We assume that in a valid layout all lenses are placed such that they do not intersect the route or each-other. Let L_i denote a lens corresponding to the i^{th} POI. Given a valid lens layout $\mathbf{L} = \{L_1 \ldots L_n\}$, we define three metrics to measure its quality: *lens distance* (C_{ld}), which measures the physical proximity of the lenses to their POIs (principle 1), *spatial coherency* (C_{sc}), which measures the distance between consecutive lenses (principle 4), and *visual coherency* (C_{vc}), which measures the similarity of directions between successive POIs and the corresponding lenses (principle 5). We define the overall quality of a layout $Q(\mathbf{L})$ as the weighted sum of the three metrics:

$$Q(\mathbf{L}) = \alpha C_{ld}(\mathbf{L}) + \beta C_{sc}(\mathbf{L}) + \gamma C_{vc}(\mathbf{L}), \qquad (1)$$

where lower values correspond to better layouts. We will now describe each of the metrics in detail.

Lens Distance

The *lens distance* metric, C_{ld} measures the physical proximity of the lenses to their POIs (corresponding to layout principle 1 in section 5.1):

$$C_{ld}(\mathbf{L}) = \sum_{i=1}^{n} \|L_i - P_i\|, \qquad (2)$$

where L_i is the center of the i^{th} lens, P_i is the location of the corresponding POI, n is the total number of POIs and $\|\cdot\|$ indicates the Euclidean distance between two points. This metric favors layouts which place lenses close to their POIs. Fig. 7 shows good and bad layouts according the lens distance metric.

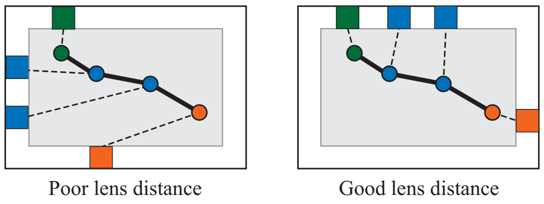


Fig. 7: The lens distance metric measures how close the lenses are to their POIs.

Spatial Coherency

The *spatial coherency* metric, C_{sc}, measures the proximity between successive lens placements (corresponding to layout principle 4 in section 5.1):

$$C_{sc}(\mathbf{L}) = \sum_{i=1}^{n-1} \|L_{i+1} - L_i\|. \qquad (3)$$

This metric favors placing the lenses next to one another in sequence, as shown in Fig. 8.

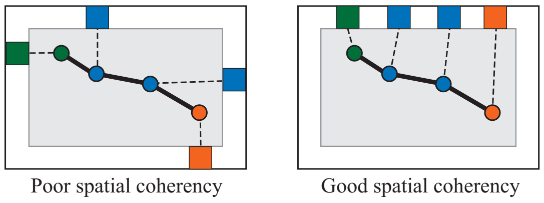


Fig. 8: The spatial coherency metric measures the distance between the lenses.

Visual Coherency

Visual coherency metric, C_{vc}, refers to the directional similarity between successive POIs and the corresponding lenses (corresponding to layout principle 5 in section 5.1). In other words, the vector between L_i and L_{i+1} should match the vector between P_i and P_{i+1}. Mathematically, the visual coherency can be defined as:

$$C_{vc}(\mathbf{L}) = \frac{w_l}{\pi} \sum_{i=1}^{n-1} \cos^{-1}\left(\frac{P_{i+1} - P_i}{\|P_{i+1} - P_i\|} \cdot \frac{L_{i+1} - L_i}{\|L_{i+1} - L_i\|} \right), \qquad (4)$$

where w_l is the lens width and $(\cdot)$ represents the dot product. Fig. 9 shows good and bad layouts from a visual coherency standpoint.

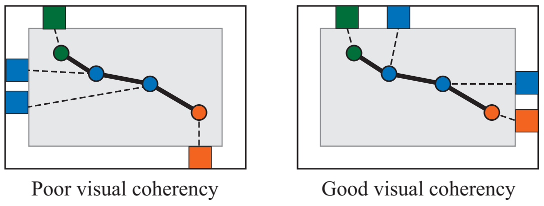


Fig. 9: The visual coherency metric measures how closely the direction between lenses matches the direction between their POIs.

Choosing α, β and γ Coefficients

Experimental results show that our algorithm is stable over a wide range of coefficient values. Fig. 10 demonstrates the effect of the individual layout metrics by assigning a weight of one to the metric being demonstrated, and zero to the others. Note that each of the metrics captures a desirable layout principle, but none of them is capable of defining a visually optimal layout in all cases. For the examples in this paper (see table in Fig. 14), we use $\alpha = 0.75, \beta = 0.15, \gamma = 0.1$ unless otherwise stated explicitly.

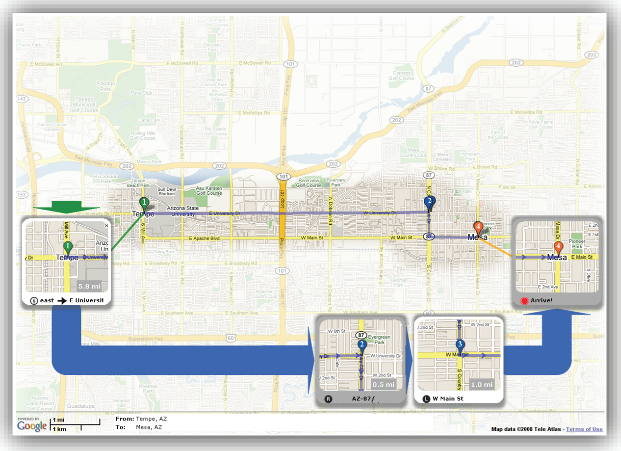
6 Producing an Optimal Lens Layout

The main objective of our layout engine is to produce optimal lens layouts based on the principles and metrics described in section 5. Allowing arbitrary lens placements leads to a large search space as well as cluttered layouts, but we can use several of the layout principles to restrict the search space and at the same time reduce visual clutter. First, based on the principle that lenses should not be placed too close to the route, we restrict lens placements to be on the map border. Furthermore, we place the lenses "in order" either clockwise or counter-clockwise around the border, based on the principle that consecutive lenses should be placed near each-other. Thus, the task becomes to find an optimal "in order" border layout for the lenses.

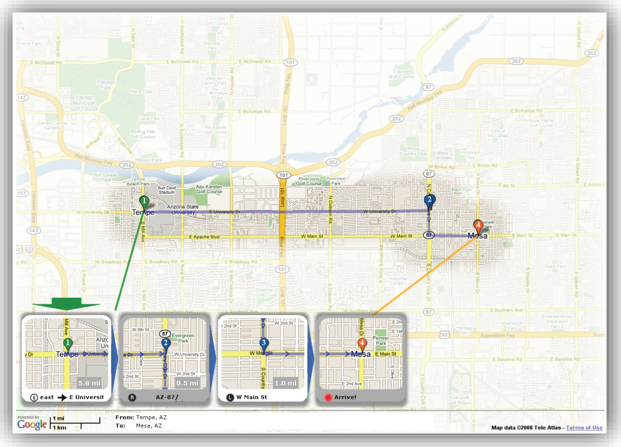
Even if layouts are restricted to lie on the map border, the search space of continuous lens placements is too large for an efficient direct solution. On the other hand, the search space of *discrete* border layouts is tractable, and the optimal discrete layout is likely to be a good approximation of the optimal continuous layout. This suggests a two-phase algorithm in which an initial step computes the optimal discrete layout, and a refinement step produces a continuous layout. Our algorithm employs this strategy, computing the optimal discrete layout by backtracking followed by a relaxation step to further refine it.

6.1 Optimal Discrete Layout.

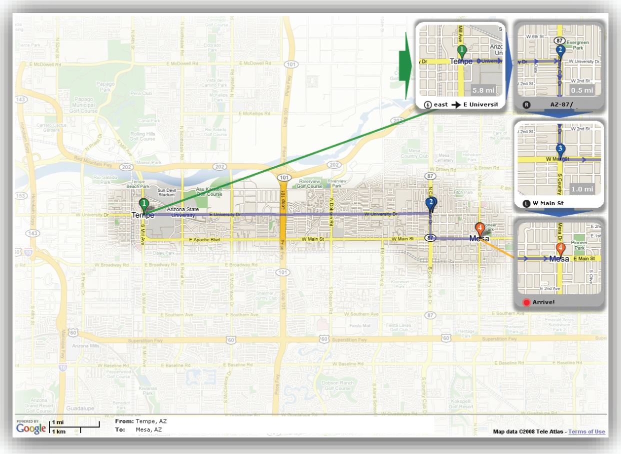
Our lens placement strategy starts by solving a discrete version of the border layout problem. Formally, the discrete border layout problem can be stated as follows: Given a route R with n points of interest, $\mathbf{P} = \{P_1 \dots P_n\}$, and m border positions $\mathbf{B} = \{B_1 \dots B_m\}$, find unique lens placements for each of the



(a) Lens Distance ($\alpha = 1, \beta = 0, \gamma = 0$)



(b) Spatial Coherency ($\alpha = 0, \beta = 1, \gamma = 0$)



(c) Visual Coherency ($\alpha = 0, \beta = 0, \gamma = 1$)

Fig. 10: A comparison of the layout metrics on a lens layout. (a) shows the layout of lenses with only the *Lens Distance* metric having a non-zero coefficient. The *Spatial Coherency* metric minimizes the inter-lens distance, without any regard for the distance from the POI (b). The *Visual Coherency* rewards configurations where successive lenses are placed in the same direction as their respective POIs (see lenses 1, 2 and 3 in (c)). For the above example: *Start:* Tempe, AZ, *Destination:* Mesa, AZ.

POIs, $\mathbf{L}^* = \{L_1 \dots L_n\} \subset \mathbf{B}$, such that the lenses are placed sequentially either clockwise or counter-clockwise around the map border, and $Q(\mathbf{L})$ is minimized:

$$\mathbf{L}^* = \{L_1 \dots L_n\} \subset \mathbf{B}, \ \operatorname{argmin} \ Q(\mathbf{L}). \tag{5}$$

An optimal solution to this problem can be found using backtracking, placing the lenses in turn around the map border. The full search tree contains

$$2mC_n^m = \frac{2mm!}{n!\,(m-n)!}$$

leaf nodes, including both clockwise and counter-clockwise orderings. Once again m is the number of border positions and n is the number of lenses. This results in a worst case scenario for a “medium” lens size, when 8 lenses are placed, of about 400,000 possible layouts.

Our backtracking algorithm uses a simple but effective bounding function to prune the search tree. This reduces runtime by about an order of magnitude while still finding the optimal discrete layout. For a node on level k of the search tree, the bounding function is

$$\widehat{Q}(k) = Q_p(k) + \alpha\widehat{C}_{ld}(k+1) + \beta\widehat{C}_{sc}(k+1), \tag{6}$$

where Q_p is the partial route cost so far:

$$Q_p(k) = Q(\mathbf{L_k} = \{L_1 \dots L_k\}),$$

$\widehat{C}_{ld}$ is the sum of the minimum possible distances between the remaining POIs and their lenses:

$$\widehat{C}_{ld}(k+1) = \sum_{i=k+1}^{n} \min_{j=1}^{m} \|P_i - B_j\|,$$

and $\widehat{C}_{sc}$ is the minimum spatial coherency between the remaining lenses, assuming that they are placed in adjacent slots:

$$\widehat{C}_{sc}(k+1) = (n-k)\|B_1 - B_0\|.$$

Note that we do not include a bounding value for visual coherency since its minimum value is zero. Our choice of pruning function ensures that the algorithm always chooses the branch with the least global cost and avoids local minima. Our optimization procedure does not optimize over the values of the coefficients themselves, but chooses an optimal layout for a specified set of α, β and γ values.

Efficiency Concerns

The backtracking solution just described is fast enough for our application, but if we want to apply the layout algorithm in a real-time setting, or place lenses on a larger map with more border locations, some optimization is in order. One simple change that often improves the backtracking performance is to change the search order, starting by placing the first lens as close as possible to its POI. Lower cost layouts are often found earlier with this search order, making the pruning more efficient.

A more aggressive optimization works by placing every other lens instead of placing consecutive lenses. A linear search can then be used to find the optimal locations of the in-between lenses. For example, instead of placing lens 1 and

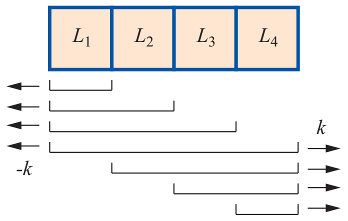


Fig. 11: Groups of touching lenses can either be moved as a unit, or split into two pieces during a relaxation step. The algorithm checks the cost of each of the $2n$ possible configurations, and moves the lenses to configuration with the lowest total cost.

then lens 2, the modified algorithm would place lens 1 and lens 3, and then place lens 2 in the optimal location between them. This procedure will still find the global minimum, since the cost associated with a particular node only depends on its location and the locations of its immediate neighbors. The modified procedure reduces the number of search tree leaf nodes to $2mC_{n-h}^{m-h}$ where $h = \lfloor (n-1)/2 \rfloor$, or about 40,000 in the worst case for the "medium" lens size setup.

We also note that a dynamic programming algorithm exists to find the optimal lens layout which has space complexity $O(m^2n^2)$ and time complexity $O(m^3n^2)$. We have not implemented it, however, as the backtracking is fast enough for our needs.

6.2 Refining the Placement by Relaxation

The lens positions generated by our backtracking method are discrete, so we refine them to a continuous layout using a relaxation procedure. The relaxation works by iteratively moving the lenses along the map border to improve the layout quality. For an isolated lens, an iteration of the relaxation procedure attempts to move the lens $\pm k$ pixels around the map border, keeping the position with the lowest Q value. Groups of lenses that touch each-other are slightly more complicated. In this case, the algorithm attempts to either (1) move the entire group of touching lenses $\pm k$ pixels around the border or (2) split the group in two and move one of the subgroups away from the other by k pixels, as shown in Fig. 11. Note that this does not change the computational complexity of the relaxation procedure; a group of n lenses must check only $2n$ positionings, the same as if the lenses were isolated. In any case, the relaxation is much faster than finding the optimal discrete layout, so adding relaxation allows us to achieve high quality continuous layouts with relatively little added cost. Fig. 18 show an example of a route visualization with and without the relaxation enabled.

6.3 Splitting the Lens Order

It is sometimes useful to split the order of the lenses. In other words, the order of the lenses is cut into two groups, and the order of the second group is reversed. Allowing a split in this manner can be beneficial, for example, when a single ordering wraps the lenses around the route. By splitting the lens order, the lenses can move in the same direction as the route on both sides of it, as shown in Fig. 12. Accounting for all possible single split orders increases the size of the search space to $2mnC_n^m$ (about 3.3 million nodes in the worst case with 16 border positions as above), but the backtracking search can still find the optimal route within a few seconds (see section 8 for timings). Fig. 19 shows an example of route splitting.

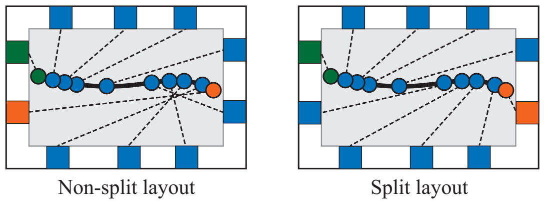


Fig. 12: Splitting the lens order allows the lens layout to follow the route more closely in some instances.

6.4 Multiple Page Layouts

For complicated routes, the number of detail lenses may exceed the capacity of a single page. In these cases, the layout engine splits the route into multiple pages and applies the layout algorithm to each page. We have found this solution to be more practical than shrinking the lenses to fit on one page, since very small lenses often do not provide enough information to be useful. The current implementation can place up to 14 lenses on the map border while leaving a mandatory gap of two tiles between the first and last lens.

6.5 Embellishments

In addition to the lenses and POI markers, the layout engine adds a number of embellishments to help the user navigate the route visually. These include route highlighting, multi-POI markers, and the replacement of leader lines with arrows.

Route Highlighting.

We highlight a route on the base map by drawing the route as a bold line, and by graying out parts of the map that are far away from the route.

Lens Highlighting.

We highlight the lenses with alternating white and gray backgrounds to enable a quick lookup while driving. We also show small blue arrows along the route on a lens to clearly specify the direction of travel around the specific POIs.

Muli-POI Markers

When multiple POI markers on the route overlap we combine them, using a special marker, 8, to indicate that multiple POIs are located in a single spot. This icon is labeled with the first POI number in the group. This helps to clear up confusion when multiple markers are placed directly on top of one another.

Replacing Leader Lines with Arrows

The standard way to connect the POIs with their lenses would be to use leader lines. However, early tests showed that a large number of leader lines can obscure route details, so we add leader lines only to the first and last lens on each page, or each group of lenses if the layout engine splits the lens order. The remaining leader lines are replaced by arrows between the lenses, as shown in Fig. 13. An arrow is also placed before the first lens so that the user can instantly see the starting point of the route. An arrow placed after the last lens on a page indicates that the visualization contains additional pages.

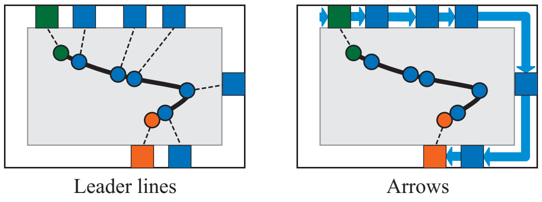


Fig. 13: Replacing leader lines by arrows makes the route less cluttered and easier to follow.

7 Implementation

Our system is built using a two-tier architecture. The front-end user interface is written in Javascript. It handles the interaction with Google Map objects. For efficiency purposes, the core algorithms of our method are implemented as a Java Applet and Java Servlet. Both modules communicate via the Netscape LiveConnect classes. The system is currently hosted on a 3.46GHz Intel Pentium D machine running Windows XP x64 Professional with 4GB of RAM.

To create a route visualization, the user types in a source and destination for the route. The system then retrieves an overview image of the route, rescaling the map extents if necessary to make space for the lenses around the map border. It then highlights the route and retrieves the map directions and corresponding POIs. Finally, the system retrieves the imagery for the lenses and positions them using backtracking and relaxation. Response times vary from around 3-4 seconds for simple cases to 11-12 seconds for longer, complicated routes, with the majority of the time being taken to retrieve the relevant imagery from Google Maps and to create embellishments. Fig. 14 provides timings for our algorithm on different route sizes.

8 Evaluation and Results

In this section, we present the results of an informal user study conducted with 18 participants. The users consisted of individuals from diverse backgrounds: 8 were students and 10 working professionals with at least a Master's degree. 13 subjects were male and 5 female. 16 currently owned a car or had owned one previously, while 12 had more than 6 years of driving experience in the US. 2 subjects with less than 6 years of driving experience did not have a valid US driver's license.

The study was conducted in two phases, the *learning* phase and the *evaluation* phase. In the learning phase, the participants familiarized themselves with route maps produced by our system. They were also tested on the interpretation of abbreviated symbols as used in our lens design. In the second phase, the participants were presented with twenty pairs of example maps that compared specific design choices. The individual design choices were: (a) the lens size, (b) the placement of leader lines, (c) split or no split in the lens order, (d) inset placement vs. border layout methods and (e) plain text vs. abbreviated symbols.

8.1 Learning Phase

In the learning phase, the participants were familiarized with our application, as well as being trained and tested on the interpretation of the symbols used in our maps. The training involved an understanding of a route map produced by our system (Start: Boston, MA; End: Philadelphia, PA). We explained the terminology used in our approach. In addition, users were presented a printable version of the of the same map as produced by Google Maps website to compare and contrast. Users were then asked to answer a series of questions about the map produced by our system. The questions included tasks such as identifying the start and end locations, identifying visual cues on the detail lenses, marking a POI and its corresponding lens, and identifying leader lines.

The symbol testing involved two steps (a) unassisted interpretation, and (b) assisted interpretation. In *unassisted interpretation*, the participants were not provided any external input or key to help the interpretation process. The participants were asked to quickly translate symbol sequences corresponding to directions that might be found on lenses. For example, the sequence

“🅛 **Lemon Street**”

translates to “Turn left on Lemon Street.” In the *assisted interpretation*, the users again translated the symbol sequences, but were provided with a key (Fig. 4).

We observed that users who had driving (or navigating) experience > 6 months, and were also familiar with Google Maps or similar online map service, needed very little assistance in interpreting the symbols correctly. Users who had no driving or navigating experience would often interpret the symbols incorrectly in the unassisted interpretation step. Once the key was made available, all users could readily interpret the symbols and 60% of users eventually preferred symbols over plain text by the end of the user study. This suggests that the process of learning our symbolic abbreviations does not place excessive demands on the end-user.

8.2 Evaluation Phase

The *evaluation phase* of the user study consisted of side-by-side comparison of pairs of maps by the participants. The participants were asked to indicate which of the two maps they would prefer to use as a navigation aid, and also specify the reason for the same. We designed two sets of twenty examples each. A participant was randomly assigned a set for evaluation. The examples were designed to test one design choice per example. We present the results of our user study below. The color scheme for the result graphics (pie charts) was chosen with the help of the *ColorBrewer* tool [10]. We use the “One-sample Z-test [24]” for the analysis of our test results. The analysis results are summarized in Fig. 15. For each category described below, the *null hypothesis* is stated as the following:

H_0 : *The specified design choice is not preferred over its competing choices.*

Start Location	Destination Location	Number of lenses	Total time *(secs)*	Layout time *(msecs)*	Nodes Processed
Tempe	Phoenix	5	4.28	46	4174
Gilbert	Mesa	8	7.16	2206	288897
Tempe	Dallas	11	6.57	737	86514
Yahoo Inc.	Google	11	5.68	262	25954
Melbourne	Morwell	13	5.70	143	16556
100, Rue Lord Byron, Paris	Rue Nicolas Appert, Paris	13	5.60	218	24488
Tempe	Boston	27	11.43	353	43697
Berlin	Warsaw	28	7.31	1131	133230
Madrid	Lisbon	29	9.52	100	13355

Fig. 14: Timing results for our layout algorithm: The first two columns specify the start and end locations of the route, respectively. The number of lenses for a route is shown in the third column. The fourth column shows the total time taken by our system (in seconds) to generate the layout (including the embellishments), and the fifth column shows the time required for our backtracking algorithm (in milliseconds) to generate an optimal solution. The last column shows the number of nodes processed by our backtracking algorithm. For all the above examples, $\alpha = 0.75, \beta = 0.15, \gamma = 0.1$. We use the "medium" lens size for these examples. Note that for the worst case example (with 8 lenses) our pruning criteria reduce the number of nodes actually traversed in the search tree from 3.3 million to just under 300 thousand.

We use $\alpha = 0.05$ as the critical value for rejecting H_0.

Lens Size: As Fig. 15 demonstrates, H_0 was rejected with a significant probability for the "medium" lens size. By observing the proportion of actual responses over the entire test set, we can conclude that the "medium" size lenses were preferred for the maps (see Fig. 16). The next preferred size was "large." Users who preferred the "small" lenses remarked that they were already familiar with the route and thus did not need as much detailed information about the POIs. It was interesting to note that in a few cases, the users preferred the "medium" lens size even though it was not presented in the examples. Such cases were categorized as the user having "no opinion" on the presented example (refer Fig. 16(b)). As the "medium" lens size was not referred to explicitly in such cases, they were excluded from the Z-test analysis. Thus, in terms of both readability and number of pages, the "medium" size lens was the preferred choice, and we use this size as the default lens size in our implementation.

Layout Method: H_0 was also rejected for the Border layout method (refer Fig. 15). In more than 85% of the examples, the users preferred our border layout over the traditional inset placement methods. The examples where users preferred the inset placement methods were routes with few lenses (< 10), and where the lenses did not overlap with the route or with other detail lenses. In instances where some lenses could not be placed due to overlap constraints, the same users chose the border layout as a much cleaner and clutter-free layout. Fig. 16(e) summarizes these results. By the Z-test results and the actual proportion of selections, we can conclude that the border layout method was chosen over the local layout by a statistically significant amount.

Leader Lines: Fig. 15 shows that H_0 was rejected for the option of first and last leader lines. The users preferred this option over the other two options. Having a leader line at regular intervals (in our examples, every third lens) was the next preferred option. When comparing the above two options against the option of having a leader line for all lenses, the users did not prefer the latter. The option of all leader lines was preferred only in cases where the lines did not intersect the route itself. The overall user feedback suggests that too many lines on the map led to a cluttered layout, whereas having fewer leader lines allowed for faster lookup when used with the lens highlighting (see Fig. 16).

Abbreviated Directions: The *null hypothesis* was not rejected for this category. Thus, we cannot conclusively state the user preference for plain text over our abbreviated symbols, based on the sample from the user study. Fig. 17 shows the relative distribution of votes for both the categories. 8 of the 18 users noted that once the user had learned the symbols, then it was very easy to follow the directions. An interesting result that emerged from this comparison was that a majority of male subjects preferred the symbols, while the female subjects were more inclined towards the plain text. A common suggestion among all users was to increase the number of lines of symbols visible on the detail lens. Several users also asked for the ability to specify user-defined fonts for the symbols.

Split in Lens Order: H_0 was rejected for the option of "No split" in the lens order. Comparing the respective "successful" sample counts, the "No split" option was the preferred choice in over 95% of the cases from the user study sample. In general, the split in the lens ordering was confusing for most users, and they preferred to have an unbroken view of the lens sequence.

All the design choices are available as GUI options in our implementation. While the user study helped in determining the default values of the design parameters, we provide the user full flexibility to switch to the parameter of his choice.

Design Choice	**Sample Size** n	**Hits** P	**Success Rate** P/n	**Z value** $(P-\mu_P)/\sigma_P$	**P(Z)**	**Conclusion**
Lens Size						
Small	70	17	0.2428	-4.302	8.4E-06	not preferred
Medium	66	48	0.7272	3.692	0.999	**preferred**
Large	64	31	0.4843	-0.25	0.401	not preferred
Lens Placement						
Local	81	11	0.1358	-6.555	2.77E-11	not preferred
Border	81	70	0.8641	6.555	0.999	**preferred**
Leader Lines						
First and last lenses	34	23	0.6764	2.057	0.980	**preferred**
Every 3rd lens	54	29	0.5370	0.544	0.706	not preferred
All lenses	72	25	0.3472	-2.592	0.004	not preferred
Directions						
Plain text	54	20	0.3703	-1.905	0.028	not preferred
Symbols	54	33	0.6111	1.6329	0.948	not preferred
Lens Order						
Split	44	2	0.0454	-6.030	8.18E-10	not preferred
Not split	44	42	0.9545	6.030	0.999	**preferred**

Fig. 15: Statistical significance of our user study results. The first column enumerates the design choices under consideration. The second column shows the sample size n, i.e. the total number of tests in which this design choice was evaluated for all the subjects. The third column shows the observed hits for the category, and the fourth column expresses the success rate as the ratio of "success" hits for that category, P, to the total sample size n. The fifth and sixth columns show the estimated Z value, and the probability of rejecting the *null hypothesis*, H_0. The last column states whether the particular design choice was preferred or not, based on the user study. The rejection cut-off value for each Z-test is 0.05 (Z_{CV} = 1.645). The null hypothesis is rejected if the observed Z value is greater than 1.645.

The direct head-to-head comparisons for individual design choices are given in Fig. 17. The user evaluation in our current study was an informal evaluation process. As part of our future work, we wish to evaluate the impact of graphical symbols and text in much more detail, using the guidelines and framework as suggested in [20].

Design Choice (A vs B)	**User choice A**	**User choice B**	**%(A)**	**%(B)**
Lens Size				
Small Vs Medium	5	30	16.6	83.3
Medium Vs Large	18	12	60	40
Small Vs Large	12	19	35.29	55.88
Inset Layout Vs Border Layout	11	70	13.41	85.36
Leader Lines				
First Vs Every 3rd	4	4	50	50
First Vs All	19	6	73.1	23.07
Every 3rd Vs All	25	19	54.34	41.3
Split Vs No split	2	42	4.54	95.45
Text Vs Symbols	20	33	37.03	61.11

Fig. 17: Head-to-head comparisons of various design choices in our user study.

8.3 User Feedback

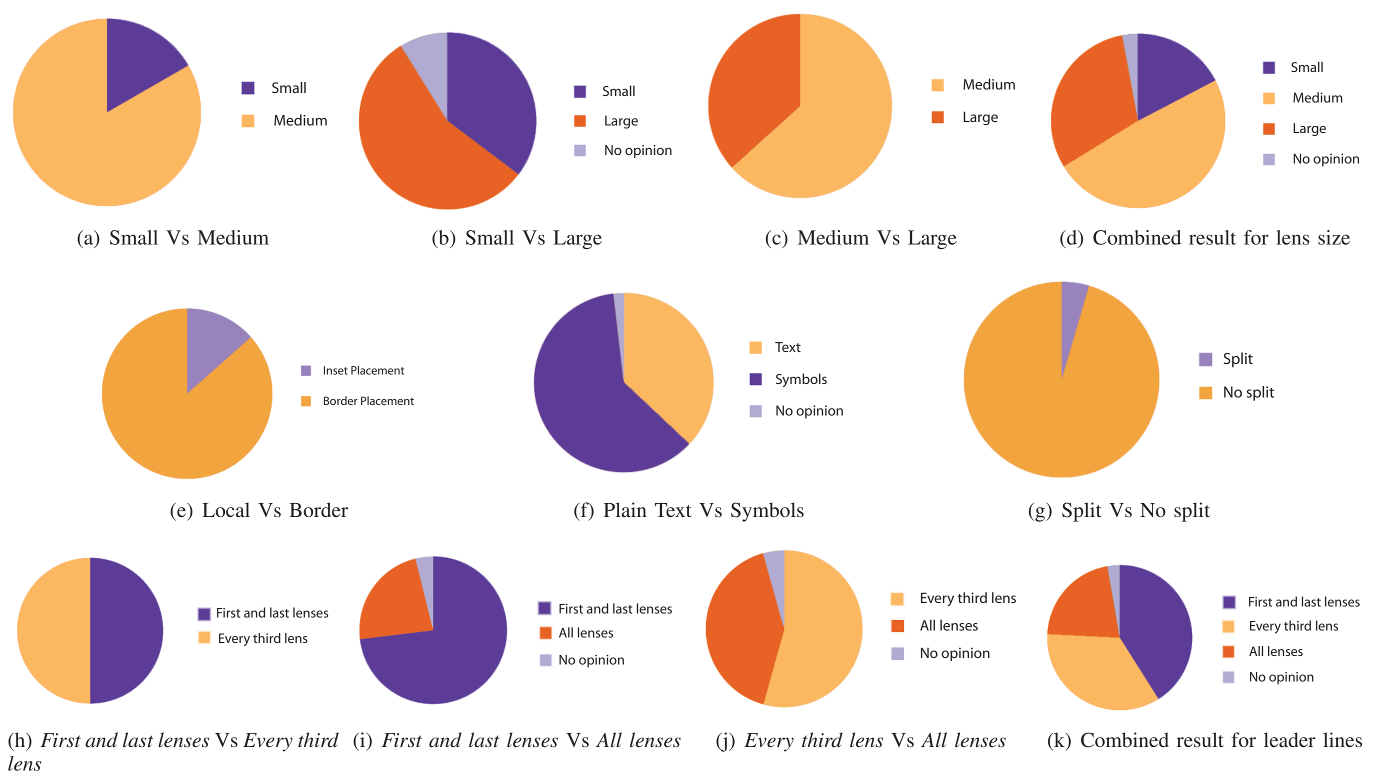
We asked the users to choose one (or more, if applicable) usability scenarios where they would use the maps produced by our system. 9 of the 18 participants said that they would use our maps exclusively, without any additional tools, while 11 said that they would use our maps in conjunction with existing route map services. The users were also asked to rate the system on a scale of 0 ("I don't see anything") to 5 ("The maps are crystal clear"). 7 users gave the system a rating of 5, while 11 gave it a rating of 4 ("The maps are useful but need improvements").

As the final step in the evaluation process, the users were asked to provide comments and/or suggestions to improve the usability of the system. We received several interesting suggestions with respect to usability and aesthetic value of our maps. Several users have asked for a display of temporal characteristics of the route as well. This included total time to traverse the route, as well as time to go from one lens to the next and the duration of partial routes in case of multiple pages. Some users wanted our system to be more interactive and include dynamic zoom-on-demand for the detail lenses. This leads to exciting possibilities of future work for interactive media like handheld devices or GPS systems.

8.4 Discussion of Alternate Layout Methods

The local placement method treats lens layout as a standard map labeling problem. This method closely follows the point labeling methods described in [18], [17], [15]. Our implementation follows a discrete search space approach, similar to [17], to find the best possible locations for the lenses that do not intersect the route. As can be seen in Fig. 6, for real-world scenarios the local layout method will often drop lenses and be quite cluttered.

Furthermore, due to the selective nature of the algorithm, there is no guarantee of visual or spatial coherency among the lenses that are displayed on the map. This makes local



(a) Small Vs Medium (b) Small Vs Large (c) Medium Vs Large (d) Combined result for lens size

(e) Local Vs Border (f) Plain Text Vs Symbols (g) Split Vs No split

(h) *First and last lenses* Vs *Every third lens* (i) *First and last lenses* Vs *All lenses* (j) *Every third lens* Vs *All lenses* (k) Combined result for leader lines

Fig. 16: User study results for the various design choices. See Fig. 17 for tabulated results.

placement a poor choice when the route has more than a small number of lenses, as borne out by the user study.

Though our method does not directly replace the LineDrive method proposed by Agrawala [3], it alleviates some of the shortcomings of their approach. Our approach combines visibility of individual turns in a complicated route, and also ensures that all the POIs and their corresponding turns are visible. We do not distort the route in any fashion and also provide the contextual information surrounding the route. Our method does not assume any familiarity with the environment, and the use of symbolic abbreviations is aimed to aid non-English speaking tourists. We "mask" extraneous information away from the route with our border layout. Only the region surrounding the route is highlighted to help users focus their attention on nearby landmarks.

The AAA (www.aaa.com) provides a TripTik map and directions service that enables the user to print customized directions. These directions are highly detailed and also reflect current information regarding road status, such as construction delays and detours, or road closure due to inclement weather. However, these maps are generally quite bulky to print out and carry along. A TripTik map from Seattle to San Diego with 14 POIs results in a 27 page document; compared to a 1 page map with 9 POIs using our method (from Google Maps).

9 Conclusion and Future Work

This paper presented a new method for visualizing route maps with multiple focus points. We formulated basic layout rules for such multiple foci maps to avoid clutter while supporting the visual indexing system of the user. This led to a design that augments a traditional route map with detail lenses that are optimally placed around the map border. We provide a user evaluation of the usability of our maps. The user study validates our design choices for the layout. The overall user feedback suggests that while the users are enthusiastic about our layout method, they would prefer to use it alongside existing route maps that they are familiar with. Thus, we can conclude that our lens design needs further refining before it can totally replace the existing methods.

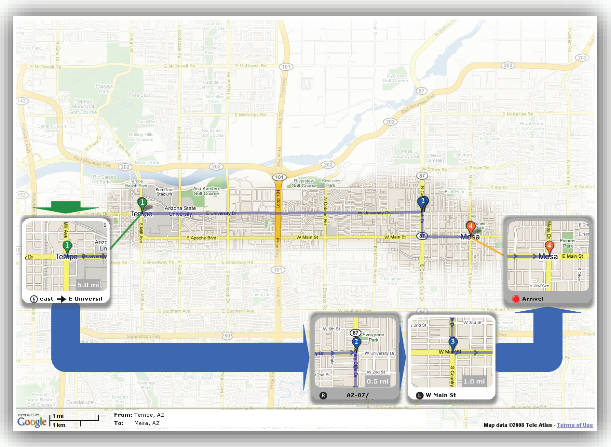
For example, the content of our lenses is currently limited to the imagery provided by Google Maps. The lens views could be improved by making sure that relevant street labels and other landmarks are always visible.

Currently, we truncate the directions if they exceed one line on the lens, with the direction list acting as a backup. It would be interesting to explore further shortening the directions or allowing more than one line of text on the lens in such cases.

We would also like to do a more rigorous study for the design of graphical and textual elements in the lenses. For example, what fonts and icon sizes are ideal for displaying information?

A design parameter that our current study does not address is finding the threshold beyond which the user experiences an "information overload," i.e. the user is unable to discern any meaningful information from the lens. A usability study that would determine this value is an important area for future research.

Though the authors have used the maps from our system on their journeys, potential future work involves testing the


(a) Discrete optimal solution

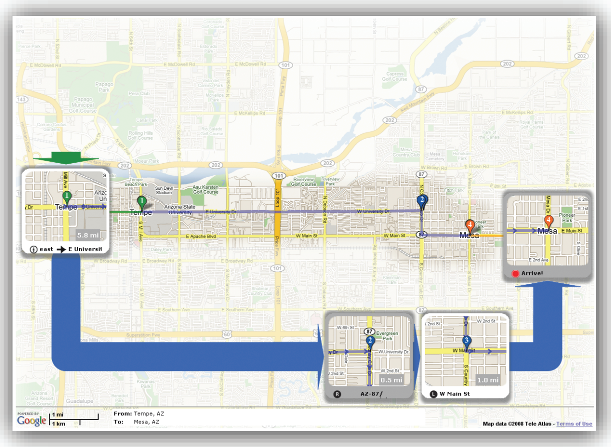
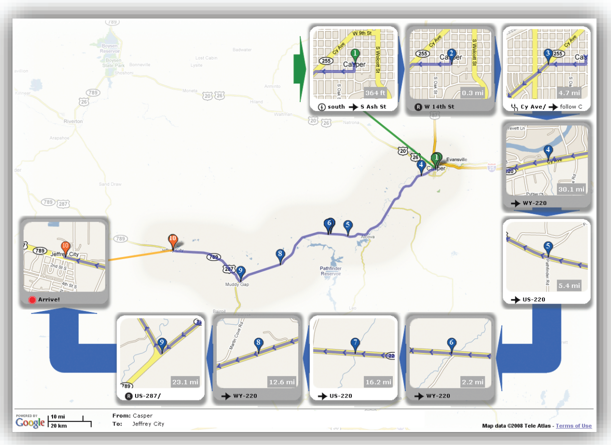

(b) After continuous relaxation

Fig. 18: Comparing the lens layout with the discrete solution, and after applying the relaxation step.


(a) Lens layout without split

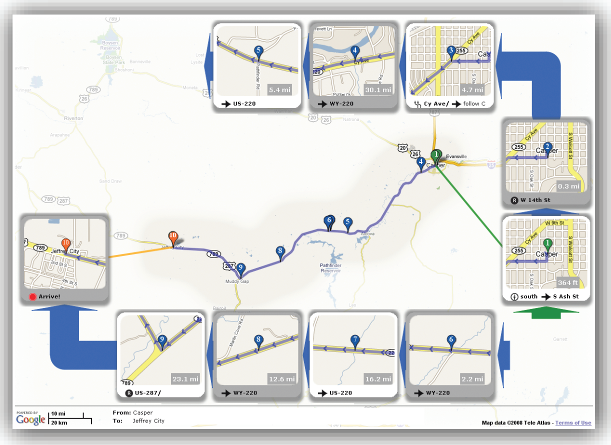

(b) Lens layout with split

Fig. 19: Comparing the lens layout without break in the sequence, and with break.

usability of the maps in a real-world situation with teams of users. Such a study would require extensive resources that are currently out-of-scope for this project, however.

An interesting extension to the current method would involve context and user-driven POI extraction, or suppression. A user may wish to omit many of the lenses for familiar parts of the route, or add lenses for intermediate destinations such as stores, rest areas, restaurants, gas stations, and landmarks. We are currently looking into ways to customize the route maps further based on user preferences and intermediate points along the route.

Acknowledgements

The authors would like to thank the participants of the user study for their help in refining our visualization design, and the anonymous reviewers for their insightful comments during the review process. The authors also acknowledge the support of the National Science Foundation (NSF) and the National Geospatial-Intelligence Agency (NGA).

References

[1] Manual on uniform traffic control devices. http://mutcd.fhwa.dot.gov/, 2004.

[2] M. Agrawala. *Visualizing route maps*. PhD thesis, Stanford, CA, USA, 2002.

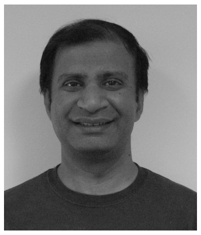
[3] M. Agrawala and C. Stolte. Rendering effective route maps: improving usability through generalization. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 241–249, New York, NY, USA, 2001. ACM.

[4] J. Ahn and H. Freeman. A program for automatic name placement. *Cartographica*, 21(2–3):101–109, 1984.

[5] J. R. Akerman, editor. *Cartographies of Travel and Navigation*. The University of Chicago Press, 2006.

[6] G. L. Allen. From knowledge to words to wayfinding: Issues in the production and comprehension of route directions. In *COSIT '97: Proceedings of the International Conference on Spatial Information Theory*, pages 363–372, London, UK, 1997. Springer-Verlag.

[7] C. Beard and A. M. Robbins. Scale determination and inset selection within a totally automated map production system. *Cartography and Geographic Information Systems*, 17(1):49–56, January 1990.
[8] K. Been, E. Daiches, and C. Yap. Dynamic map labeling. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):773–780, 2006.
[9] M. A. Bekos, M. Kaufmann, A. Symvonis, and A. Wolff. Boundary labeling: Models and efficient algorithms for rectangular maps. *Comput. Geom. Theory Appl.*, 36(3):215–236, 2007.
[10] C. A. Brewer. *ColorBrewer*. http://www.ColorBrewer.org, accessed May 04 2009.
[11] F. R. Broome, C. Beard, and A. Martinez. Automated map inset determination. Proceedings: Eighth International Symposium on Computer-Assisted Cartography, pages 466–470, 1987.
[12] L. Chittaro. Visualizing information on mobile devices. *Computer*, 39(3):40–45, 2006.
[13] L. Chittaro, R. Ranon, and L. Ieronutti. Vu-flow: A visualization tool for analyzing navigation in virtual environments. *Special Issue on Visual Analytics, IEEE Transactions on Visualization and Computer Graphics*, 12(6):1475–1485, November 2006.
[14] J. Christensen, J. Marks, and S. Shieber. An empirical study of algorithms for point-feature label placement. *ACM Transactions on Graphics*, 14(3):203–232, 1995.
[15] S. Edmondson, J. Christensen, J. Marks, and S. Shieber. A general cartographic labeling algorithm. *Cartographica*, 33(4):13–23, 1997.
[16] J.-D. Fekete and C. Plaisant. Excentric labeling: Dynamic neighborhood labeling for data visualization. In *CHI*, pages 512–519, 1999.
[17] S. A. Hirsch. An algorithm for automatic name placement around point data. *The American Cartographer*, 9(1):5–17, 1982.
[18] E. Imhof. Positioning names on maps. *The American Cartographer*, 2(2):128–144, 1975.
[19] K. A. K. Hartmann, T. Götzelmann and T. Strothotte. Metrics for functional and aesthetic label layouts. In F. Clöister, editor, *5th International Symposium on Smart Graphics (SG'05)*, pages 115–126. Springer Verlag, 2005.
[20] A. Klippel and D. R. Montello. *Spatial Information Theory*, volume 4736 of *Lecture Notes in Computer Science*, chapter Linguistic and Nonlinguistic Turn Direction Concepts, pages 354–372. Springer, Berlin / Heidelberg, August 2007.
[21] K. L. Lovelace, M. Hegarty, and D. R. Montello. Elements of good route directions in familiar and unfamiliar environments. In *COSIT '99: Proceedings of the International Conference on Spatial Information Theory*, pages 65–82, London, UK, 1999. Springer-Verlag.
[22] A. A. Martinez. Automated insetting: An expert component embedded in the census bureau's map production system. In *AUTO CARTO 9: Ninth International Symposium on Computer-Assisted Cartography*, pages 181–190, Falls Church, Virginia, 1989. American Society for Photogrammetry and Remote Sensing (ASPRS).
[23] D. C. Robbins, E. Cutrell, R. Sarin, and E. Horvitz. Zonezoom: map navigation for smartphones with recursive view segmentation. In *AVI '04: Proceedings of the working conference on Advanced visual interfaces*, pages 231–234, New York, NY, USA, 2004. ACM.
[24] R. S. Schulman. *Statistics in Plain English with Computer Applications*. Van Nostrand Reinhold, New York, 1 edition, 1992.
[25] K. H. T. Götzelmann and T. Strothotte. Agent-based annotation of interactive 3d visualizations. In A. K. P. O. A. Butz, B. Fisher, editor, *6th International Symposium on Smart Graphics 2006, LNCS 4073*, pages 24–35. Springer Verlag, 2006.
[26] B. Tversky and P. U. Lee. Pictorial and verbal tools for conveying routes. In *COSIT '99: Proceedings of the International Conference on Spatial Information Theory*, pages 51–64, London, UK, 1999. Springer-Verlag.



Pushpak Karnick received his BE in Computer Engineering from University of Pune, India in 1998 and is currently a Doctoral Candidate at Arizona State University, Tempe. His research interests include Geospatial Visualization Methods, Information Visualization, Procedural Modeling, and the application of Immersive Environments in Human-computer Interaction. He is a student member of the IEEE.

Stefan Jeschke received his Diploma and PhD in Computer Science from the University of Rostock in 2001 and 2005, respectively. He is currently a Post-Doc researcher at the Vienna University of Technology. His research interests include Image-based Real-time Rendering, Geometric Surface Details, Shadows and Global Illumination for Real-time Rendering, as well as Efficient Image and Volume Representations.

David Cline received an M.Sc. and Ph.D. in Computer Science from Brigham Young University in 1998 and 2007, respectively. He currently works at the Partnership for Research in Spatial Modeling (PRISM) lab at Arizona State University. His research interests include Global Illumination, Sampling, and GIS Visualization Methods.

Anshuman Razdan received his BS and MS in Mechanical Engineering and Ph D in Computer Science. He is an Associate Professor in the Department of Engineering and the Director of Advanced Technology Innovation Center (ATIC,) and the I^{3}DEA Lab (i3dea.asu.edu) at Arizona State University at Polytechnic campus. He has been a pioneer in computing based interdisciplinary collaboration and research at ASU. His research interests include Geometric Design, Computer Graphics, Document Exploitation and Geospatial visualization and analysis. He is the PI and a collaborator on several federal grants from agencies including NSF, NGA and NIH and DHS. He is a member of the IEEE.

Peter Wonka received the MS degree in urban planning and the doctorate in computer science from the Technical University of Vienna. He is currently with Arizona State University (ASU). Prior to coming to ASU, he was a postdoctorate researcher at the Georgia Institute of Technology for two years. His research interests include various topics in Computer Graphics, Visualization, and Image Processing. He is a member of the IEEE.