Gaze Map Matching: Mapping Eye Tracking Data to Geographic Vector Features

Peter Kiefer
Institute of Cartography and Geoinformation
ETH Zurich
Wolfgang-Pauli-Str. 15
CH-8093 Zurich, Switzerland
pekiefer@ethz.ch

Ioannis Giannopoulos
Institute of Cartography and Geoinformation
ETH Zurich
Wolfgang-Pauli-Str. 15
CH-8093 Zurich, Switzerland
igiannopoulos@ethz.ch

ABSTRACT

This paper introduces *gaze map matching* as the problem of algorithmically interpreting eye tracking data with respect to geographic vector features, such as a road network shown on a map. This differs from previous eye tracking studies which have not taken into account the underlying vector data of the cartographic map. The paper explores the challenges of gaze map matching and relates it to the (vehicle) map matching problem. We propose a gaze map matching algorithm based on a Hidden Markov Model, and compare its performance with two purely geometric algorithms. Two eye tracking data sets recorded during the visual inspection of 14 road network maps of varying realism and complexity are used for this evaluation.

Categories and Subject Descriptors

I.5 [**Pattern Recognition**]: Models; I.2.1 [**Artificial Intelligence**]: Applications and Expert Systems—*Cartography*

General Terms

Algorithms, Measurement

Keywords

Eye tracking, map matching, gaze-based assistance

1. INTRODUCTION

Eye trackers are used to measure the visual attention of individuals. They have a number of potentials for studying and improving the way humans interact with the interface of a geographic information system (GIS). On the one hand, they offer valuable data that can be exploited in usability studies. In the geovisualization community, for instance, gaze data analyses have been combined with standard usability metrics to evaluate the effectiveness and efficiency of interactive map interface designs [3, 4].

Another application of eye tracking consists in real-time map interaction, such as gaze-based zooming and panning [27], or using the user's gaze history as visual clue for better orientation on a map [9]. These gaze-based assistive technologies are specifically interesting for the support of geospatial tasks in scenarios where interaction possibilities are restricted, such as wayfinding with a mobile map [13].

In both, usability studies and gaze-based assistance, previous work on eye tracking for GIS has typically interpreted gaze with respect to few areas of interest (AOI) which were not originally in the underlying vector data. For instance, the analysis presented in [3] interprets the gaze data as a sequence between control elements of a GIS (map panel, layer selection panel, query results dialog, etc.).

In this paper we argue that, especially for the case of interactive maps where the map content may change frequently, an intelligent interpretation of gaze should take the current contents of the map into account: 'which map object, i. e., which road, river, lake, point of interest etc., was (or is) the user inspecting?'.

An algorithm that automatically answers this question would take a sequence of gaze points as an input and match these points with the vector features displayed on a map. To the authors' best knowledge, this problem of mapping gaze to geographic vector features (the *gaze map matching* problem) has never been addressed in literature before. From a geometric point of view, it is related to the problem of matching a sequence of vehicle position measurements to the road segments most probably travelled (known as 'map matching' in literature) [19]. Problems that make gaze map matching challenging include hardware inaccuracy, bad calibration, the permanent unintentional movements of the eye, and ambiguity resolution when a gaze point is in the vicinity of more than one map element.

The paper compares three gaze map matching algorithms: a naive algorithm, one working with a sliding window, and one probabilistic algorithm based on a Hidden Markov Model (HMM). A dataset of eye tracking data collected from two participants during the visual inspection of 14 road network maps of varying realism and complexity is used for the evaluation. For this inspection task, the HMM algorithm performed better than the naive algorithms. We argue that this is due to the possibility to parameterize the HMM to the task, whereas the other two algorithms are purely geometric.

The paper is structured as follows: section 2 gives a basic introduction to eye tracking, its applications for geo-spatial

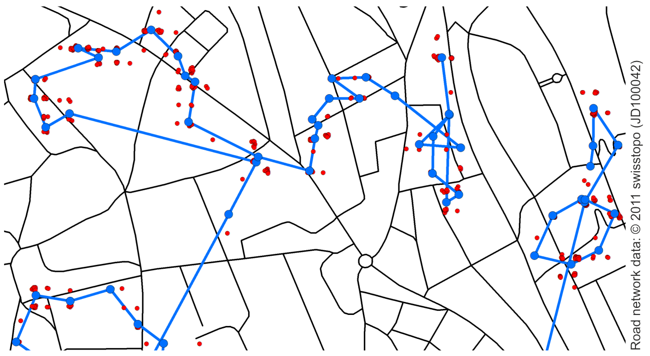

Figure 1: Gazes (red small dots), connected fixation sequence (blue large dots), and road network.

interfaces, and lists related work in the related areas of (vehicle) map matching and conflation. Section 3 introduces the gaze map matching problem and discusses its challenges and interdependence with the user's task. Algorithms for gaze map matching are described in section 4 and evaluated with a dataset reported on in section 5. A conclusion and outlook section closes the paper.

2. RELATED WORK

2.1 Introduction to Eye Tracking

Eye tracking, i. e., the process of measuring and analyzing where an individual is gazing at, is a well-established research method in a number of fields. Reading research, for instance, has been one of the first fields to use eye tracking in their studies [20].

There are several techniques eye trackers can build on to capture the eyes' movements. The most common one uses the principle of infrared corneal reflection. An infrared illuminator causes a reflection on the eye (or on both eyes for binocular devices), this reflection is captured by a camera, and the point of gaze is computed using image processing and a mathematical model of the eye [7].

Eye tracking data come as sequence of *gazes*, i. e., (x,y) coordinates in the reference frame of the system. For remote eye trackers calibrated to a computer display, the (x,y)-point represents screen coordinates. For head-mounted eye trackers, the gaze coordinates are measured w.r.t. a video of a front-facing environment camera.

An established approach in eye tracking is to pre-process gazes by grouping some of them to *fixations* (see Fig. 1). A fixation occurs when the eyes remain relatively still over a certain amount of time, which is the time when new information can be perceived. The time threshold neccessary in order to obtain new information is task-dependent. In our case, the fixation threshold was selected as 150 milliseconds after pre-tests. *Saccades* are very fast movements of the eyes and occur between fixations. Even when fixating to a certain location, the eyes are never still. Small saccades (so-called *mini-saccades*) can be measured continuously. Refer to [7] and [20] for introductory texts on eye tracking.

2.2 Eye Tracking in HCI and GIScience

Human-Computer-Interaction (HCI) uses eye tracking methodology in two ways: on the one hand, gaze serves as additional data for usability analyses. The data recorded during an empirical usability experiment are later analyzed in order to, for instance, improve the visual design of a graphical user interface.

On the other hand, eye trackers can also be accessed in real-time, thus serving as an additional input device. The user can trigger an interaction with the system using her gaze, which allows for easy, natural, and fast ways of interaction [11, 29]. Novel interaction concepts based on eye tracking are continuously evolving, using both, explicit and implicit interactions. According to Schmidt [24], explicit interaction refers to the intentional direct manipulation of a computer system by the user, whereas implicit interaction is defined as an action that is not primarily intended to be an interaction, but is registered by the system as such. An example for explicit interaction would be eye-typing [16]. An example for implicit interaction would be a system that pre-caches geo data depending on the map position gazed at [1]. Section 3.1.2 will develop a vision of a system that uses implicit interaction, building on top of gaze map matching.

In geographic information science (GIScience), eye trackers are mostly used for the purpose of evaluation. Early eye tracking studies in cartography had the goal of evaluating different map designs [26]. One important finding from this era is that gaze behavior on maps depends on the participant's task. More recent studies include the evaluation of interactive map interface designs [3, 4], or the usability of cartographic animations [18]. Other studies have used eye tracking to investigate spatial decision making in lab studies [32], and to perform real-world wayfinding studies in both, indoor [25] and outdoor [13] environments.

Gaze-based interaction with cartographic maps has been proposed on a content-independent level. For instance, [27] implement explicit gaze-based panning by automatically centering to the user's gaze position after a certain fixation duration. Another approach (GeoGazemarks, [9]) displays an aggregated gaze history to help the user recall those places on the map she has interacted with before. The map content, such as road networks, lakes, or points of interest have never been taken into consideration. One possible reason for the lack of gaze-based content-related interaction with maps could be the inaccuracy of eye trackers and the saccadic movement of the eye.

2.3 Conflation

Conflation refers to the combination of two geodata sets in order to create another data set that increases spatial accuracy and consistency. Starting in the 1980s, a number of methods for conflation have been developed [22]. In general, these methods can be categorized in vector to vector (closest to gaze map matching), vector to raster, and raster to raster data conflation [5]. Several techniques for vector to vector conflation exist [6, 30], some specialized for the conflation of two road networks [23].

A gaze track, connected with a line, and a map can be seen as two geodata sets to which vector to vector conflation could be applied. However, the saccadic movement of the eye during map perception produces very complex line strings which, in quite many parts, does not follow any linear features on the map (see Fig. 1). It can be doubted whether conflation would return acceptable results for gaze map matching.

2.4 Vehicle Map Matching

Literature uses the term 'map matching' for the problem of matching a sequence of (potentially inaccurate) vehicle position measurements to road segments of a street network [19]. In the scope of this paper, we refer to this problem as *vehicle map matching*. Position is typically measured with a Global Navigation Satellite System (GNSS). Some specialized algorithms consider other position methods, such as WiFi [31]. Besides purely geometric vehicle map matching (e.g., [28]), more advanced algorithms include topology and/or speed constraints into their reasoning (e.g., [15]). In the last years, probabilistic methods have gained some popularity, such as the HMM-based approach presented in [17] which is the basis of one of our algorithms in section 4.

Gaze map matching has a similar goal as vehicle map matching, with the decisive difference that, in general, topological and speed constraints do not apply to gaze data. Saccades from one part of the map to a completely different part may appear at any time. As an exception, extra knowledge can be utilized if we know the user is performing a task that somehow relates to, say, the topology of the network (refer to section 3.3). Even then, the task model never poses as strong restrictions as the physical restriction of a vehicle being forced to travel along the road segments. This excludes the transfer of most of the vehicle map matching methods to our problem.

3. GAZE MAP MATCHING

Gaze map matching aims at interpreting eye tracking data w.r.t. the contents of a cartographic map. The underlying vector data of the map are used to match the gaze sequence to the inspected geographic features, such as roads, rivers, or points of interest.

3.1 Application Scenarios

This paper focusses on the basic algorithmic problem of gaze map matching. Applications building on top of gaze map matching have not been implemented yet. To underline the benefits of gaze map matching, two potential applications are sketched in the following.

3.1.1 Usability Analyses

Daniel is a cartographer. He is designing an interactive online map that should help bicyclists find a safe, convenient, and fast route through the city of Zurich. With an eye tracking study he has collected gaze data from 10 participants. Daniel wonders whether his prototypical map design helps to clearly separate biking paths from major roads.

Current standard eye tracking analysis software allows to manually draw polygons (so-called areas of interest, AOI) in the reference system of the screen, or in a marker-defined reference system. Figure 2 shows an example screenshot of the D-Lab software[1] with six AOI around major roads on a paper map. The software can be used to compute statistics w.r.t. these AOI, such as the percentage of gazes that was inside one of them.

These statically defined AOI have obvious drawbacks: they do not work for interactive maps that allow for panning or zooming, such as the online map designed by Daniel. They are also ambiguous because they often overlap. Gaze

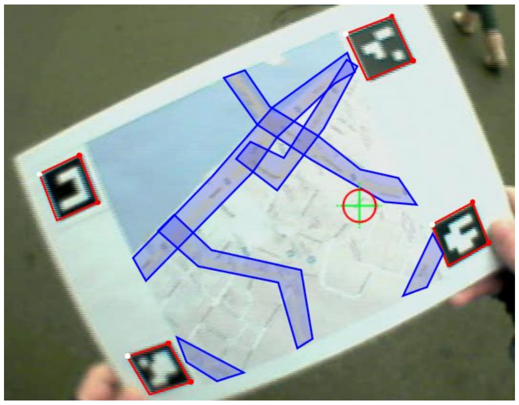

Figure 2: Manually defined areas of interest (AOI) buffering major roads on a paper map.

map matching, in contrast, would automatically use the geographic features displayed in the current map extent. No manual definition of AOI is required, and panning/zomming are automatically accounted for. An intelligent gaze map matching algorithm would also disambiguate at intersections.

3.1.2 Gaze-Based Assistance

Sandy is sitting on a London underground train. She is wearing her augmented reality glasses with integrated eye tracking. The glasses show a topological map of the underground network. Sandy starts to inspect the Waterloo line. After one second, the system recognizes her interest in that specific line and overlays according additional information, such as transfer options, and points of interest.

As mentioned in section 1, gaze-based assistance is specifically interesting in situations where interaction possibilities are restricted, which often applies to mobile situations. From a hardware perspective, eye tracking is becoming increasingly mobile: head-mounted mobile eye trackers are used in research already today [14], and it is only a small step for eye tracking to be integrated into mobile consumer products, such as smartphones[2] or augmented reality glasses[3].

Given this technological background, one can easily imagine systems like the one described in the example. Again, this requires algorithms that dynamically interpret gaze with respect to the currently displayed vector features. In contrast to ex-post usability analyses, gaze-based assistance requires gaze map matching algorithms that run incrementally, i. e., with an incomplete gaze sequence, and in real-time. We do not present incremental gaze map matching algorithms in this paper.

3.2 The Gaze Map Matching Problem

In the previous sections, gaze map matching was described as the problem of matching eye tracking data to the geographic vector features displayed on a map. Gazes and vector features are expressed in different coordinate systems and, for interactive maps, the projection between the two may change over time:

[1]`http://www.ergoneers.com/`

[2]e.g., Apple™'s patent application, `http://www.freepatentsonline.com/y2012/0036433.html` (June 29, 2012)

[3]e.g., the Google Glass™ project, `https://plus.google.com/111626127367496192147` (June 29, 2012)

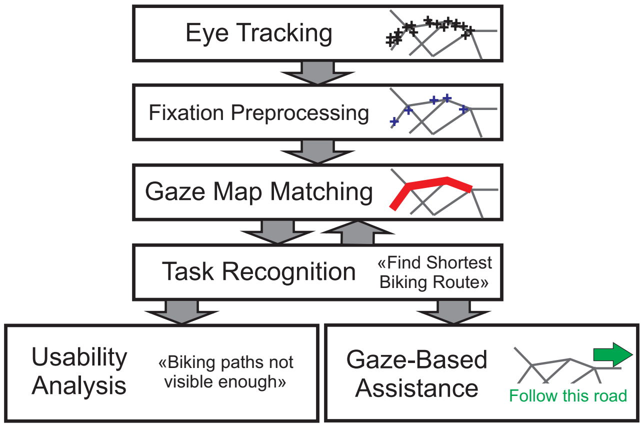

Figure 3: Gaze map matching and task recognition.

Definition 1. A *geo-referenced fixation sequence* is a time-indexed sequence $\phi = \langle (x_0, y_0, \pi_0), \ldots, (x_n, y_n, \pi_n) \rangle$, where (x_t, y_t) is the fixation position at time t in the screen coordinate system C_s, and π_t: $C_s \rightarrow C_m$ denotes the projection from screen to map coordinate system valid at time t.

As explained in section 2.1, eye tracking research generally assumes that perception takes place only when the gaze remains still for a minimum amount of time. This is why fixation sequences are used instead of gazes here. The preprocessing converting a gaze sequence to a (much shorter) fixation sequence is omitted here (refer to section 5.1.2). We can now define the gaze map matching problem as follows:

Definition 2. Let **G** denote a set of geographic vector features, all defined in C_m. Let ϕ denote a geo-referenced fixation sequence with projections to C_m. The *gaze map matching problem* consists in finding the according feature sequence $\gamma = \langle g_0, \ldots, g_n \rangle$ from $\mathbf{G} \cup \{\omega\}$, where the i-th element of ϕ is matched with the i-th element of γ, and ω is assigned to those fixations that cannot be mapped to features from **G**.

Gaze map matching algorithms need the projection information to convert from C_s to C_m, and to perform any distance computations that may become necessary (e.g., buffering). Fixations too far away from any feature may be assigned ω, i. e., 'not matchable'.

3.3 Gaze Map Matching and User Tasks

For general scene perception it has been shown that a person's gaze behavior is influenced by the task [33]. Early studies on eye tracking and cartography have indicated that the same is true for maps [26]: the task a person is solving on a map will influence her gaze. In the bicycle map example presented in section 3.1, for instance, a participant told to find a biking route from A to B will probably look at A, B, and then follow different route options between them. A free-viewing task ('tell me whether you like the visual design of this map'), will probably lead to a more dispersed and less clearly structured gaze behavior, maybe with more gazes on visually salient features.

The automatic recognition of the task a viewer is performing on a map could build upon gaze map matching (see Fig. 3). The recognized task in turn would serve as input

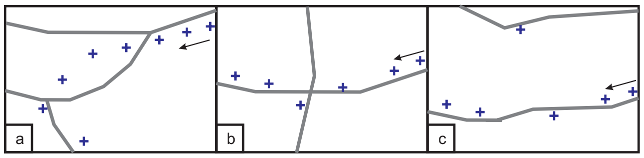

Figure 4: Inaccuracy and disambiguation.

for gaze-based assistance, or be analyzed w.r.t. usability requirements. Although task recognition is out of the scope of this paper, the task layer is relevant for us (indicated by the two converse arrows in Fig. 3): gaze map matching can be facilitated with knowledge about the task the user is performing. If we already know, for instance, that the user is searching for the shortest biking route, we can expect fixation sequences along topologically connected edges of the biking network graph, whereas topologically unconnected sequences are less likely to happen. The type of feature we can expect the user to gaze at (point, polyline, polygon) also depends on the task. These expectations can be used for ambiguity resolution when a fixation is in the vicinity of more than one feature.

Our point in this paper is that knowledge about the task can be used to make gaze map matching algorithms more intelligent. We do not aim at solving gaze map matching for any possible task.

Thus, for the rest of this paper, we focus on a specific class of tasks we call *inspection tasks*. In an inspection task, the user's gaze traverses the edges of a graph (e.g., the road segments of a street network) at low speed. In most cases, the traversal path will follow topologically connected edges. A saccade to an edge further away will only rarely happen. We perform experiments in which participants have to gaze at a predefined path on the map. By giving the participants almost no freedom we can be sure to have a ground truth against which we can evaluate the algorithms (see section 5).

3.4 Challenges

3.4.1 *Technological and Physiological Challenges*

Eye tracking data are inaccurate due to a number of technological and physiological reasons. Accuracy depends – among others – on the device used, the eye physiology, the relative position of the eye tracker to the head, the lightness conditions of the recording environment, and the calibration quality. Most eye tracker manufacturers do currently not report properly on the accuracy specifications of their devices, given certain conditions [10].

Thus, and due to the mini-saccades (see section 2.1), the gaze point recorded by the device will almost never be located exactly on the geographic feature currently inspected. The same applies to fixations. An additional challenge has been mentioned in section 2.4: fast saccades to distant parts of the map may occur at any time.

3.4.2 *Spatial Disambiguation*

The spatial disambiguation challenge is a logical consequence of inaccuracy. As the fixation points are never exactly on the inspected features, a distance based approach becomes necessary. A fixation located close enough to a feature is matched with this feature. Ambiguity occurs when a fixation is in the vicinity of more than one feature.

This problem is well-known in vehicle map matching, and typically resolved by taking a larger termporal and spatial context into account. For instance, it is clear how to match the vehicle trajectories displayed in Figs. 4a) and b) to road segments although parts of the trajectories are close to more than one road. The 'jump' from one road to the other in Fig. 4c) would be interpreted as inaccuracy (and thus be matched to the lower road) because there is no sensible way of traveling from the lower to the upper road given the speed restrictions of the vehicle.

If the points in Fig. 4 were the result of an eye tracking experiment it would not be that simple: the data shown in Fig. 4c) could then result from a participant really having looked at the upper road for a short moment (for whatever reason). In theory, it could even be possible that there is no inaccuracy in the data, i. e., that the participant has really gazed at the white area between the two roads in Fig. 4a). Another possible hypothesis could be that the gaze has been jumping back and forth between the upper and lower road in Fig. 4a).

The spatial disambiguation challenge for gaze map matching thus consists in deciding how much spatial context can be used for disambiguation or not. This is task-dependent (see section 3.3). A gaze map matching algorithm well-suited for one task may perform bad for a different task.

4. ALGORITHMICALLY MATCHING GAZE WITH A ROAD NETWORK

This section introduces two simple geometric algorithms for the gaze map matching problem (see section 3.2), and one probabilistic algorithm that can be parameterized for a specific task. A road inspection task is assumed (see section 3.3), and the set of geographic vector features **G** contains only road segments.

4.1 Closest Line Matching

The base case algorithm is a simple geometric approach that takes only the distances between the fixation points and the vector lines into consideration. For each geo-referenced fixation $\phi_{<t>}$ (the t-th element in the geo-referenced fixation sequence ϕ), the algorithm retrieves all features from **G** (accesible with a spatial index) that are located in a predefined radius, and returns the closest one of them as g_t. If no feature is within the radius, ω is returned for time t, indicating 'not matchable'.

The radius is computed in a way that it corresponds to the 30 pixels that were also used for fixation computation. We transformed this value into map units using the projection π_t from screen to map coordinate system valid when the gaze data were retrieved.

4.2 Simple Sliding Window

The simple sliding window algorithm is a first approach for spatial disambiguation. It is inspired by the situation that typically occurs at crossroads, displayed in Fig. 4b: the closest point matching algorithm would map the point in the center to the vertical line, although it is likely that this fixation is inaccurate, given the inspection of the horizontal line before and after the point.

The sliding window algorithm takes n past and m future fixations into account. In our case, a symmetrical window with $n = m = 2$ was successful. The algorithm first uses the closest line matching algorithm to retrieve the closest line for each fixation. The algorithm then sequentially moves from left to right, starting at time index 2, and compares all fixations in the 5-neighborhood around the current time index. If these match the pattern L1-L1-L2-L1-L1, where L1 and L2 have to be different, and one of L1 and L2 may be ω, the middle position (L2) would simply be replaced by L1. The runtime complexity of both, closest line matching and simple sliding window, is obviously linear with respect to the length of the fixation sequence.

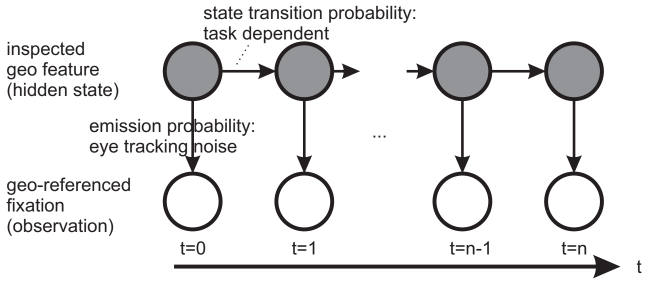

Figure 5: HMM structure

4.3 Hidden Markov Model Inference

Hidden Markov Models (HMM) are dynamic probabilistic models that are used for reasoning under uncertainty in temporally evolving situations (see [21, chapter 15.3] for an introduction). Among many other applications, they have been used in vehicle map matching to find the most likely path a vehicle has traveled [17]. In this paper, the HMM algorithm is used to demonstrate how task knowledge can improve gaze map matching.

4.3.1 Model Structure

An HMM is a graphical model with two states (more precisely: two random variables) for every time slice: one hidden and one observed state. In our case, the observed state is a geo-referenced fixation (x_t, y_t, π_t). The hidden state is the geo feature g_t the user has inspected at time t. The model evolves over time, i. e., the length of the observation sequence ϕ determines the number of time slices of the HMM (see Fig. 5). The directed edges in the graph express conditional dependencies: the geo feature a user inspects at time $(t + 1)$ depends only on the road segment she has inspected at time t (inter-slice dependencies). The geo-referenced fixation we observe at time t depends only on the geo feature a user inspects at time t (intra-slice dependencies).

The first order Markov assumption (between slices) is a simplification due to the lack of a sophisticated task model. For other tasks than road inspection it may be an oversimplification: for instance, a user performing a structured search task on the whole map is less likely to return to features she has seen at *any* time before. In contrast, a user with a comparison task may be more likely to return to features she has seen before.

4.3.2 Probability Distributions

Three probability distributions need to be specified for an HMM: the initial probability distribution, the emission probabilities, and the state transition probabilities.

Initially, i. e., for time slice $t = 0$, we assume an equal distribution over all features. This makes sense because a

Node distance	Weight
0 (same line)	5
1	4
2	2
>2 and 'unconnected'	1

Table 1: Weights used for the state transition probabilities of the HMM.

system typically has no initial expectation on which parts of the map the user will be more interested in. More sophisticated models could use feature saliency to capture the 'eye-catchingness' that may attract the first gaze.

Emission probabilities model the inaccuracy of the system, i. e., the probability that a geo-referenced fixation $\phi_{<t>} = (x_t, y_t, \pi_t)$ is observed, given the user is inspecting a given feature g_i. To our knowledge, there is no related work on how eye tracker inaccuracy can be approximated mathematically. Lacking better options, we use a zero-mean Gaussian as in [17, p.339]:

$$p(\phi_{<t>}|g_i) = \frac{1}{\sqrt{2\pi}\sigma_z} e^{-0.5(\frac{d}{\sigma_z})^2} \ ,$$

where d is the distance between $\phi_{<t>}$ and g_i, and σ_z is the standard deviation of the eye tracker. As mentioned in section 3.4.1, most eye tracking manufacturers do not report on the error measures of their devices. Besides, the standard deviation depends on the size of mini-saccades, which in turn depends on the user. We used the size of the box used for fixation computation (see section 5.1.2) as an approximation.

State transition probabilities model the dynamics of the system. In our case, this is the probability that the user's gaze stays on the same, or changes to another geo feature. This probability is highly task dependent. For inspection tasks on a network, the edge g_t, and edges that share a common node with g_t are more likely to appear in *t+1* than edges that are far away. Still, edges far away can never be assigned a zero probability (as in road matching). Our experiments showed that the values listed in Table 1 were successful in the evaluation (where weight w means a probability w-times of the probability of 'unconnected', normalized over all state transitions).

For different tasks, especially those with more freedom, these weights will differ. Free exploration tasks, as an extreme case, could have the same weight for all state transitions (which is not very interesting), or assign state transitions depending on saliency.

4.3.3 Inference

We use the Viterbi algorithm [8], a dynamic programming algorithm which finds the most likely sequence of hidden states in an HMM for a given sequence of observation – in our case: the most likely feature sequence γ for a geo-referenced fixation sequence ϕ.

On maps with many road segments the state transition probabilities were very small. For long gaze sequences, this caused Viterbi to run into underflow problems due to many multiplications. We solved this problem with a common technique: by using the natural logarithm of the probabilities and replacing multiplication by summation.

A post-processing assigned ω to positions t in the result sequence if no geo feature was in the radius of the respective $\phi_{<t>}$ – in other words, if the closest line matching algorithm would also have returned ω.

The runtime complexity of Viterbi is known to be O(n x $|\mathbf{S}|$), where n is the number of observations, and $|\mathbf{S}|$ is the size of the state space (in our case: the number of lines $|\mathbf{G}|$). For large maps and/or gaze-based assistance scenarios this may lead to inacceptable runtimes where more efficient approximate inference algorithms should be preferred over Viterbi. This was not necessary in our case.

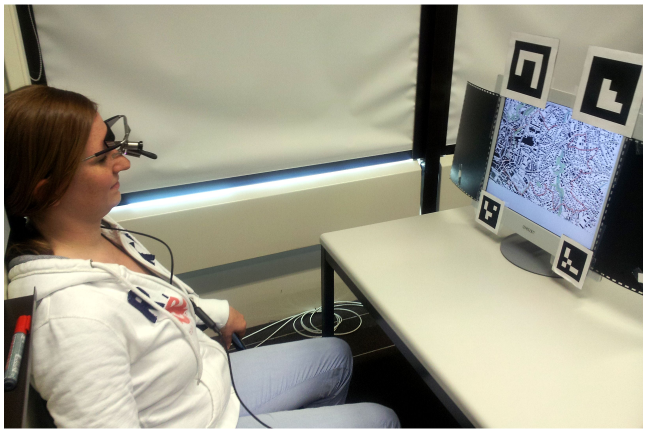

Figure 6: Data collection setup

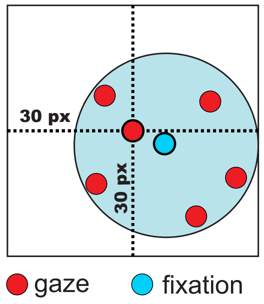

Figure 8: Fixation computation.

5. EVALUATION

5.1 Data collection

5.1.1 Hardware and Software Setup

Our hardware consisted of the Ergoneers Dikablis head-mounted mobile eye tracker with a gaze capture rate of 50 Hz. The data were transmitted via a coaxial cable to a laptop, designated only for gaze recording. From there they were transmitted via WiFi with 25 Hz to the working laptop where the framework for map visualizations, gaze capturing, and analysis was running. The maps were displayed on a 18" desktop screen (see Fig. 6).

We used a mobile head-mounted eye tracker although stationary (remote) eye trackers would return data of higher accuracy for desktop studies. This makes gaze map matching harder than necessary. However, our algorithms should also be able to handle situations with low accuracy, such as walking through the city with a map displayed on a tablet.

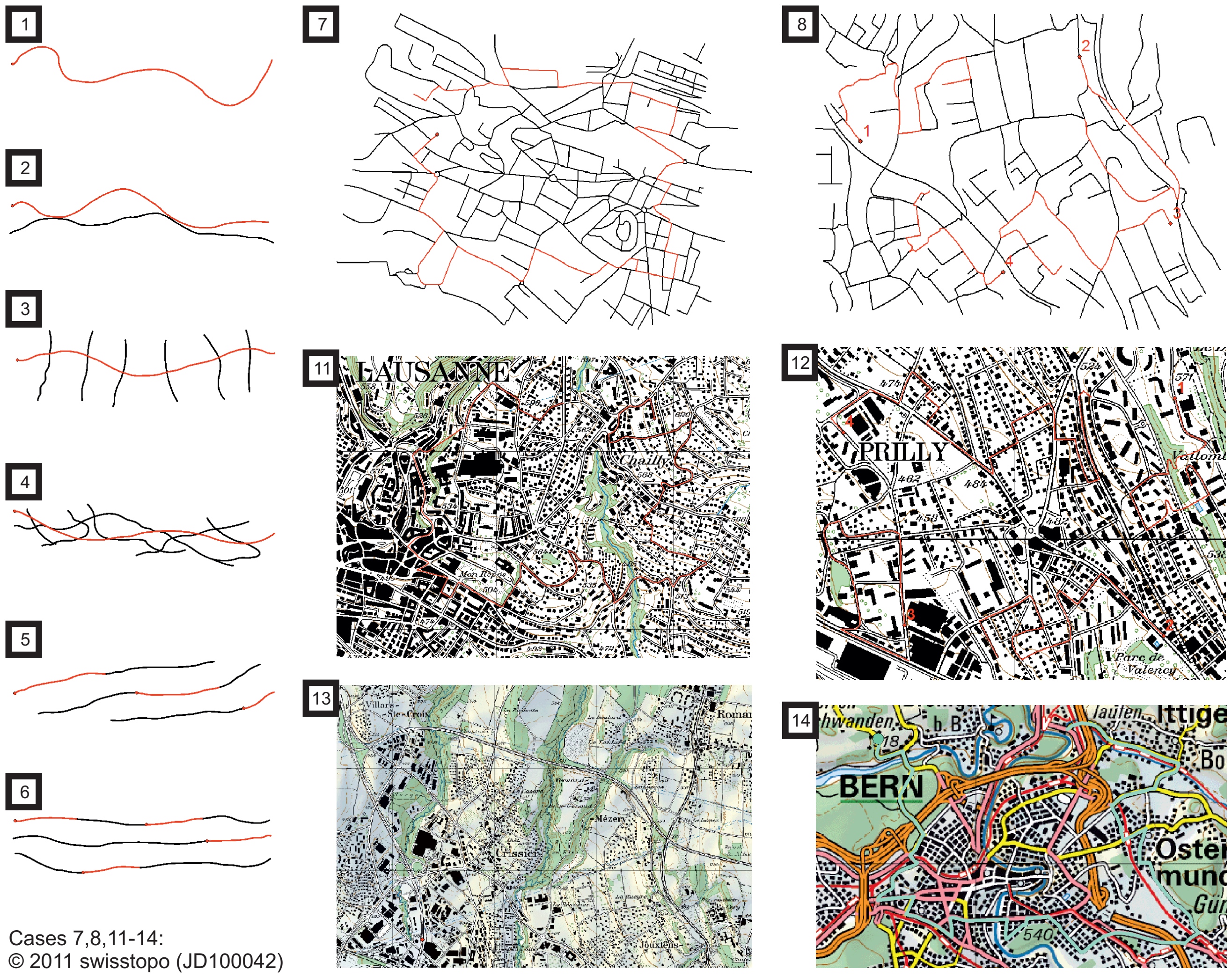

Figure 7: Cartographical material used for the evaluation (cases 9/10 are vector maps of the same regions as cases 11/12).

We developed a framework (a Java 6 application) that supports eye tracking user studies for vector and raster maps. The framework captures, pre-processes, and stores the gaze of participants while they are inspecting the map. The analysis part of the framework provides functionality to post-process the recorded gaze data after the study, i. e., to run our gaze map matching algorithms and visualize the results. The result visualization helped to get a first idea of the algorithms' functionality (Figs. 9 and 11 are screenshots of the framework, enhanced with labels).

5.1.2 Computation of Fixations

Fixations were computed from the gaze sequence as follows (see Fig. 8): a quadratic window with side length 60px was created around the first gaze point. If for a given time threshold (150 milliseconds in our case) all following gaze points occurred inside this window, a fixation (green point) was formed by taking the mean x and y of all gaze points inside the window. If a gaze point occurs outside the window, or after having created a fixation, a new window is created around the next point and the procedure starts over again.

5.1.3 Cartographical Material

The gaze data were collected for 14 maps of road networks (we call them 'cases', see Fig. 7) with increasing complexity and realism. Cases 1 to 10 were vector maps, cases 11 to 14 were raster maps. The road network vector data for the four raster cases were available and later used by the algorithms. The coordinate system C_m of all maps was the 'swiss grid' map projection[4].

Cases 1 to 6 were manually drawn base cases, cases 7 to 10 were vector road maps from a real city. The idea of the base cases was to cover typical situations in which erroneous matchings can occur due to ambiguity: 1) One line as absolute base case (distinguish between line and ω). 2) Two lines with overlapping bounding boxes. 3) A line with several (almost) 90° street crossings. 4) A line with several crossings at small angles. 5) Three lines with the instruction to jump between them at two designated positions. 6) Three lines with three jumps, some crossing the middle line.

[4]http://www.swisstopo.admin.ch/internet/swisstopo/en/home/topics/survey/sys/refsys/projections.html

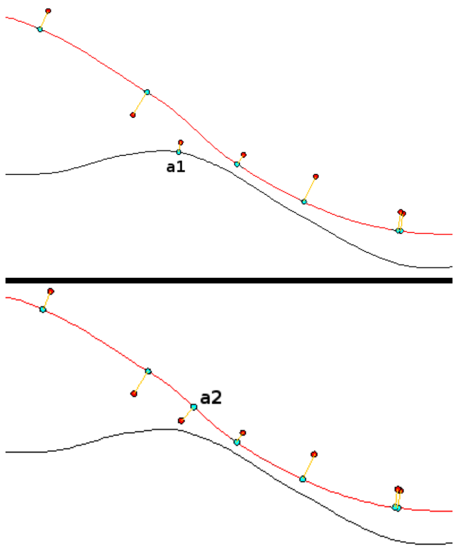

Figure 9: Spatial disambiguation through the sliding window (bottom), compared to closest line matching (top).

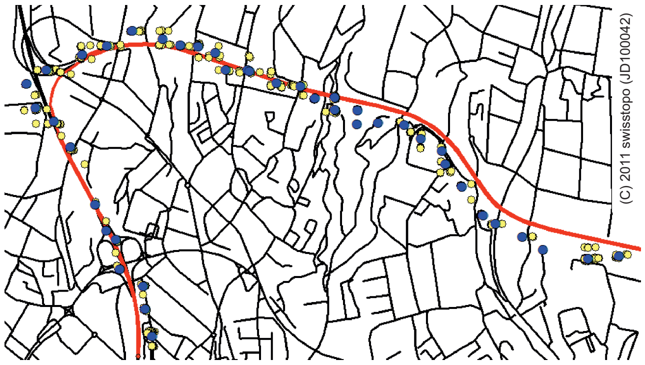

Figure 10: Example eye tracking data: gazes (yellow) and fixations (blue) on the road network underlying the raster map of case 13. Red indicates the highway the participant had to follow.

Cases 7 to 10 were vector cases based on real street network data from a topographic map (1:25.000) of the area around Lausanne, Switzerland. All topographic maps were provided by swisstopo[5] in the ESRI^{TM} Shapefile format. Cases 11 to 13 were topographic raster maps (1:25.000) of Lausanne, case 14 was a topographic raster map (1:200.000) of Bern, Switzerland. In cases, 1 to 12 a thin red line indicated the path the participants had to follow, in case 14 this line was cyan. The path in case 13 was defined by a highway.

5.1.4 Participants and Procedure

Two users participated in our experiment, each completing all 14 cases (leading to 28 datasets). The users were placed in front of the desktop screen at a distance of 80 centimeters (see Fig. 6). Four visual markers were attached

[5]Source: Bundesamt für Landestopografie swisstopo (Art. 30 GeoIV): 5704 000 000

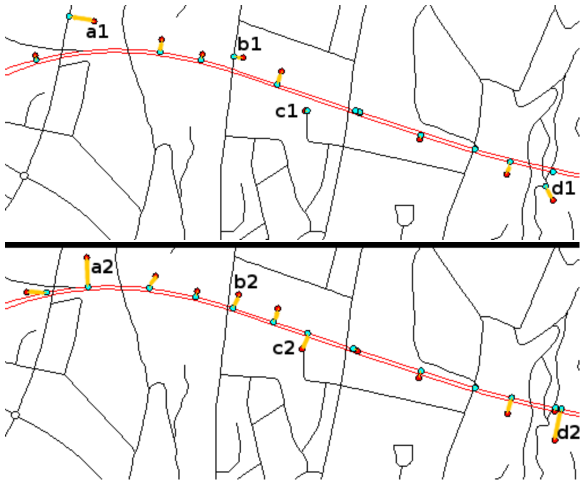

Figure 11: Spatial disambiguation through a Markov Model (bottom), compared to sliding window (top). Matched fixations from a subarea of Fig. 10.

on the screen to span up a coordinate system in which the screen coordinates for the captured captured gaze data were computed.

At the beginning of each case, participants were shown the map and given a short time to attain overview knowledge. This should avoid exploration behavior during the experiment (inspection, not exploration behavior was the focus of this study). The participants were then instructed to strictly follow the highlighted path with their gaze. In case 13, the instruction was to follow a selected highway, starting from a red dot, and change highway at the crossroads in direction East.

5.2 Results

Each of the 28 datasets was pre-processed by computing fixations from the gazes. Figure 10 shows one example dataset. Then, the data were processed with the closest line matching algorithm, the sliding window algorithm, and the HMM algorithm. The maximum runtime of the HMM algorithm was approx. 3 minutes (on a 2.33 GHz desktop computer with 8GB RAM).

All 84 resulting feature sequences were analyzed w.r.t. the number of fixations that was matched with a correct line (i. e., a line the participant was supposed to follow). The percentage of correctly matched fixations for each case, each participant, and each algorithm are listed in Table 2.

Comparing the results of closest line and sliding window, the data show that the sliding window algorithm performs better in 8 of 28 trials (29%), and worse in 3 (11%). The data, especially of participant 1, indicate that sliding window seems to help especially for simple maps with few roads, such as the manually designed base cases. Figure 9 displays an example (case 2, participant 1): the fixation marked with a1 is in the vicinity of both lines, and (probably due to inaccuracy) mapped to the wrong lower line by the closest line matching algorithm. The sliding window of size 5, eliminates this error.

Comparing the results of sliding window and HMM, the HMM clearly achieves a better result than the sliding win-

	Participant 1			Participant 2		
Cases	Closest Line	Sliding Window	Hidden Markov	Closest Line	Sliding Window	Hidden Markov
1	0.9310	1.000	0.9310	0.3142	0.3142	0.3142
2	0.7692	0.8461	0.8846	0.3888	0.3888	0.3888
3	0.6250	0.6666	0.7916	0.7142	0.7619	0.7619
4	0.4137	0.4827	0.6896	0.2941	0.2941	0.4705
5	0.5909	0.5909	0.5909	0.7857	0.7857	0.7857
6	0.6129	0.6129	0.6129	0.4400	0.4000	0.4400
7	0.3552	0.3421	0.4210	0.5243	0.5243	0.5243
8	0.3736	0.3736	0.4065	0.5365	0.5487	0.5731
9	0.4800	0.4800	0.5600	0.5396	0.5396	0.6190
10	0.4117	0.4117	0.5176	0.4555	0.4555	0.4777
11	0.3809	0.3809	0.5952	0.1904	0.1904	0.4285
12	0.5074	0.4925	0.5373	0.4069	0.4186	0.4069
13	0.2151	0.2151	0.2278	0.5978	0.6086	0.6086
14	0.4814	0.4814	0.4691	0.4867	0.4867	0.5309

Table 2: Comparing the results of the algorithms: ratio of correct matched fixations over all fixations, for two participants, 14 cases, and 3 different algorithms.

dow: it performs better in 16 trials (57%), and again 3 times (11%) worse than the sliding window. Figure 11 displays an example (case 13, participant 1): the fixations marked with a1, b1, c1, and d1 were matched wrong by the sliding window algorithm. The HMM algorithm matches them correctly to the line the user was inspecting (any of the parallel red lines was possible). This is because the transitions from the red line to the rest of the network are less likely than staying on the line. A good example is the line that c1 was matched to incorrectly, because three nodes have to be traversed to reach it from the red line.

Overall, these results indicate that more complicated maps require more sophisticated algorithms that are aware of the participant's task. A task-aware algorithm then outperforms purely geometric algorithms, such as the sliding window.

6. CONCLUSION AND OUTLOOK

This paper has introduced the gaze map matching problem which, to our knowledge, is the first attempt to an automated content-based analysis of gaze with respect to geographic features displayed on a map. The goal of gaze map matching consists in automatically determining the features a person has looked at, given inaccuracy and spatial ambiguity. Various applications in the domains of usability studies, geo visualization, and gaze-based assistance exist.

At this stage of our research, we considered gaze map matching as a geometrical problem. We demonstrated a probabilistic algorithm, working with a Hidden Markov Model, that performs better than simpler algorithms because it is able to exploit knowledge about the higher level task, an 'inspection task' in our case. An essential question for future research is how gaze map matching can be performed for other tasks. On the other hand, we will try to automatically recognize the task a user is performing on a map from her gaze. This is related to work on activity recognition from gaze [2], and intention recognition from trajectories [12]. As indicated in Fig. 3, the steps of task recognition and gaze map matching are interconnected.

Our 28 gaze datasets were taken from 2 participants. One issue for future work is to include a larger number of participants in the experiments to account for individual differences. Sections 4 and 5 discussed only line features. Future work should consider points and polygons, too.

We are planning to investigate novel concepts for gaze-based assistance that build upon gaze map matching. Interactive maps were used to motivate gaze map matching; in our study users were not allowed to pan or zoom. However, as the zoom level was treated separately in theory and implementation, the step to interactive maps is small. The HMM state transition probabilities were assigned with a systematic try-out, based on plausible assumptions. Clearly, learning these task-dependent probabilities would be a worthwhile endeavor for future work. Incremental algorithms for gaze map matching are another open research topic to enable gaze-based assistance on cartographic maps.

Acknowledgements

We wish to thank our participants for contributing their gaze data. Special thanks also to Martin Raubal for supporting our research. Our discussions with Martin helped to shape some of the ideas expressed in this paper. In addition, we would also like to thank Alain Geiger for his input on (vehicle) map matching.

7. REFERENCES

[1] K. Bektas and A. Çöltekin. An approach to modeling spatial perception for geovisualization. In *Proceedings of STGIS 2011: Spatial Thinking and Geographic Information Sciences*, 2011.

[2] A. Bulling, J. Ward, H. Gellersen, and G. Tröster. Eye movement analysis for activity recognition using electrooculography. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(4):741–753, April 2011.

[3] A. Çöltekin, S. I. Fabrikant, and M. Lacayo. Exploring the efficiency of users' visual analytics strategies based on sequence analysis of eye movement recordings. *International Journal of Geographical Information Systems*, 24(10):1559–1575, 2010.

[4] A. Çöltekin, B. Heil, S. Garlandini, and S. I. Fabrikant. Evaluating the effectiveness of interactive map interface designs: A case study integrating usability metrics with eye-movement analysis.

Cartography and Geographic Information Science, 36:5–17, 2009.

[5] C.-C. Chen, C. A. Knoblock, and C. Shahabi. Automatically conflating road vector data with orthoimagery. *Geoinformatica*, 10(4):495–530, Dec. 2006.

[6] M. A. Cobb, M. J. Chung, H. Foley III, F. E. Petry, K. B. Shaw, and H. V. Miller. A rule-based approach for the conflation of attributed vector data. *GeoInformatica*, 2:7–35, 1998. 10.1023/A:1009788905049.

[7] A. T. Duchowski. *Eye Tracking Methodology: Theory and Practice*. Springer, London, 2nd edition, 2007.

[8] G. Forney. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268 – 278, march 1973.

[9] I. Giannopoulos, P. Kiefer, and M. Raubal. GeoGazemarks: Providing gaze history for the orientation on small display maps. In *Proceedings of the 14th International Conference on Multimodal Interaction*, ICMI '12, New York, NY, USA, 2012. ACM. to appear.

[10] K. Holmqvist, M. Nyström, and F. Mulvey. Eye tracker data quality: what it is and how to measure it. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, ETRA '12, pages 45–52, New York, NY, USA, 2012. ACM.

[11] R. J. K. Jacob. The use of eye movements in human-computer interaction techniques: What you look at is what you get. *ACM Trans. Inf. Syst.*, 9(2):152–169, 1991.

[12] P. Kiefer. *Mobile Intention Recognition*. Springer, New York, 2011. PhD Thesis, Otto-Friedrich-Universität Bamberg, Germany.

[13] P. Kiefer, F. Straub, and M. Raubal. Location-aware mobile eye tracking for the explanation of wayfinding behavior. In *Proceedings of the AGILE'2012 International Conference on Geographic Information Science*, 2012.

[14] P. Kiefer, F. Straub, and M. Raubal. Towards location-aware mobile eye tracking. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, ETRA '12, pages 313–316, New York, NY, USA, 2012. ACM.

[15] Y. Lou, C. Zhang, Y. Zheng, X. Xie, W. Wang, and Y. Huang. Map-matching for low-sampling-rate gps trajectories. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '09, pages 352–361, New York, NY, USA, 2009. ACM.

[16] P. Majaranta, U. K. Ahola, and O. Spakov. Fast gaze typing with an adjustable dwell time. In *Proc. of the International Conf. on Human Factors in Computing Systems (CHI)*, pages 357–360. ACM, 2009.

[17] P. Newson and J. Krumm. Hidden markov map matching through noise and sparseness. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '09, pages 336–343, New York, NY, USA, 2009. ACM.

[18] T. Opach and A. Nossum. Evaluating the usabailty of cartographic animations with eye movement analysis. In *25th International Cartographic Conference 2011*, page 11, 2011.

[19] M. A. Quddus, W. Y. Ochieng, and R. B. Noland. Current map-matching algorithms for transport applications: State-of-the art and future research directions. *Transportation Research Part C: Emerging Technologies*, 15(5):312–328, 2007.

[20] K. Rayner. Eye movements in reading and information processing: 20 years of research. *Psychological bulletin*, 124(3):372–422, Nov. 1998.

[21] S. J. Russell and P. Norvig. *Artificial Intelligence - A Modern Approach*. Prentice Hall, third edition, 2010.

[22] A. Saalfeld. Conflation automated map compilation. *International Journal of Geographical Information Systems*, 2(3):217–228, 1988.

[23] E. Safra, Y. Kanza, Y. Sagiv, and Y. Doytsher. Efficient integration of road maps. In *Proceedings of the 14th annual ACM international symposium on Advances in geographic information systems*, GIS '06, pages 59–66, New York, NY, USA, 2006. ACM.

[24] A. Schmidt. Implicit human computer interaction through context. *Personal and Ubiquitous Computing*, 4(2/3):191–199, 2000.

[25] R. Schuchard, B. Connell, and P. Griffiths. An environmental investigation of wayfinding in a nursing home. In *Proceedings of the 2006 symposium on Eye tracking research & applications*, ETRA '06, pages 33–33, New York, NY, USA, 2006. ACM.

[26] T. R. Steinke. Eye movement studies in cartography and related fields. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 24(2):40–73, 1987.

[27] S. Stellmach and R. Dachselt. Investigating gaze-supported multimodal pan and zoom. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, ETRA '12, pages 357–360, New York, NY, USA, 2012. ACM.

[28] G. Taylor and G. Blewitt. Road reduction filtering using GPS. In *3rd AGILE Conference on Geographic Information Science*, 2000.

[29] T. Vildan and R. J. Jacob. Interacting with eye movements in virtual environments. In *Proc. of the CHI 2000 Conference on Human factors in computing systems*, pages 265–272. ACM, April 1-6, 2000.

[30] V. Walter and D. Fritsch. Matching spatial data sets: a statistical approach. *International Journal of Geographical Information Science*, 13(5):445–473, 1999.

[31] M. Weber, L. Liu, K. Jones, M. J. Covington, L. Nachman, and P. Pesti. On map matching of wireless positioning data: a selective look-ahead approach. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '10, pages 290–299, New York, NY, USA, 2010. ACM.

[32] J. M. Wiener, C. Hölscher, S. Büchner, and L. Konieczny. Gaze behaviour during space perception and spatial decision making. *Psychological Research*, pages 1–17, 2011.

[33] A. Yarbus. *Eye Movements and Vision*. Plenum, New York, 1967.