Intelligent Prioritization and Filtering of Labels in Navigation Maps

Mikael Vaaraniemi
BMW Forschung und Technik GmbH
München, Germany
mikael.vaaraniemi@bmw.de

Markus Goerlich
BMW Forschung und Technik GmbH
München, Germany
markusgoerlich@mac.com

Aick in der Au
TEXTURE-EDITOR GbR
München, Germany
aick@texture-editor.com

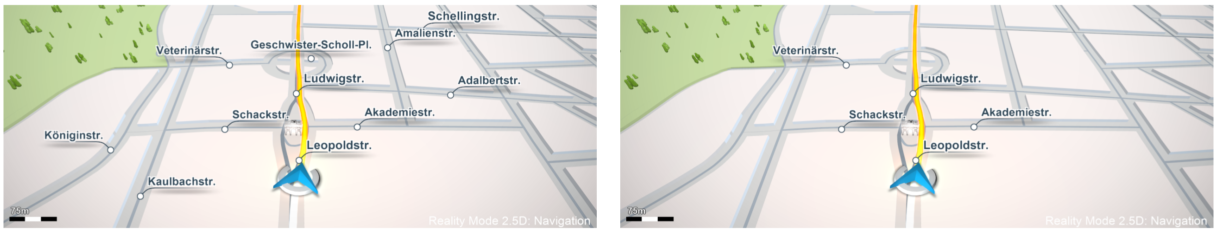


Figure 1: Filtering of labels in a 3D navigation map: (left) without filter (right) with filter.

ABSTRACT

The description of objects in navigation maps by textual annotations provides a powerful means for orientation and visual data exploration. However, displaying labels for all features leads to a cluttered map with unreadable labels and occluded information. Therefore, the overall goal is to display the most important and filter out the less important labels. In this paper, we present a general approach for filtering labels. We use the navigation in automotive maps as an application to test our approach. This involves the creation of a priority metric for ranking labels in maps. Our flexible system allows runtime configuration of the priority. Moreover, we keep the temporal coherency of label filtering; hence, jittering of labels does not occur. The system is predictable, modular, and can easily be adapted to new applications. On medium-class hardware, our real-time system is capable of filtering on average 1000 labels within 12 ms. A concluding expert study validates our approach for navigation purposes. All candidates approve the resulting clear labeling layout.

Keywords
Annotation, Labeling, Ranking, Filtering, Navigation, 3D maps, GIS.

1. INTRODUCTION

The description of objects by textual annotations provides a powerful means for visual data exploration. Compared to images, they have to be actively read, but can precisely describe objects with selected information. In Geographic information systems (GIS) labels are used to describe geospatial data, such as road networks and demographic data. In these systems, labels are important for the user's orientation and understanding of map elements. Unfortunately, the finite screen-space of desktop PCs limits the maximum amount of labels that can be displayed at the same time. This becomes even more problematic when we have a high amount of information on a small spatial area, e.g. when displaying road names of dense cities like Tokyo. The 3D view of a map accentuates this problem; Tilting the view condenses the projected spatial area onto the screen. Hence, the main problem when annotating all map features is a cluttered map with unreadable labels. Fig. 2 displays the problem, that occurs when showing all available labels of Munich in Germany at the same time.

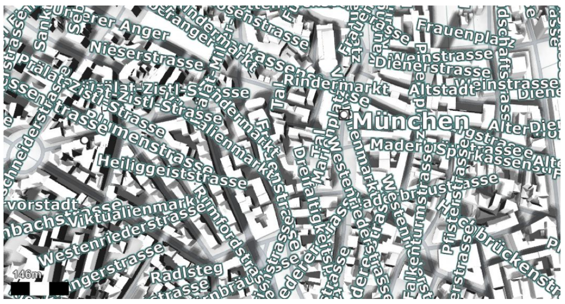


Figure 2: Display of all labels in Munich, Germany.

Therefore, to create a usable annotated map we have to prioritize all existing labels to select and display only the most important ones. Of course, removing labels is only feasible in non-critical applications.

The filtering system should follow some prerequisites. We limit the number to approximately 4 to 8 labels to stay consistent with the findings of Alvarez [AF07] and the Miller's law [Mil56]. Alvarez states in his first experiment, that a user can track between 4 fast objects and 8 slow objects on average. Additionally, more and more GIS applications have to cope with dynamic content, requiring runtime prioritization of labels without prior knowledge, e.g. when loading KML files into Google Earth. Finally, we follow the labeling rules set by Vaaraniemi et al. [VTW12]. They state that labels should behave in a temporally coherent way: they should not flicker or jump around the map.

Advantages of a label filtering approach are at hand. First, because we render less labels, we lower the overall performance impact, e.g. for placement, collision detection and rendering [VTW12]. Second, less collision between labels occur, which results in an enhanced readability. Third, the recognition and tracking of individual labels becomes easier. Fourth, the graphical association of a label with its map feature is enhanced. This is consistent with Imhof's statement [Imh75], who names legibility and the graphical association of a label with its feature as characteristics of good lettering.

1.1. Goals and Contributions

Based on the aforementioned requirements and our intended application area, we defined the goals of such a system as follows: We want to (a) display most important and filter out less important labels, (b) select approximately 4 to 8 labels, (c) create a flexible filtering system, that is configurable at runtime, (d) create a predictable, reproducible and understandable system, and (e) maintain temporal coherency where jittering of labels does not occur.

Prioritization and selection of labels is a very application specific topic. In this paper, we use navigation in 3D maps as an application to test our approach. Our proposed approach depicted in Fig. 1 has the following contributions:

- General approach for filtering a set of labels
- Priority metric for labels in navigation maps
- Temporal coherency of label filtering

In Section 2 we will present related work and survey filtering in existing commercial navigation systems. In Section 3, we will propose a system composed of 3 stages: configuration, ranking of labels and a rule-based selection. The implementation details are stated in Section 6. In Section 7, we evaluate the performance of our approach and present the findings of a conducted expert study. Finally, we conclude this paper with future research areas.

2. RELATED WORK

To assess the current state-of-the art, we first evaluated existing filtering and prioritization schemes for labels.

Simple Label Prioritization. Tatemura [Tat00] introduces fisheye maps, an approach for information exploration based on features with priorities. In his application, he uses a priority metric based on image similarities. Been [BDY06] presents an approach to label 2D maps in real-time. However, filtering is only slightly addressed: labels are selected on the basis of a geographic region and / or scale. For instance, he drops local road names when zoomed out. Bertini et al. [BRL09] introduce an excentric labeling where lenses focusses information. However, manual input for filtering labels is required: the user selects with checkboxes which labels should be displayed.

Assumed Prioritization. In most cases, the approaches describing placement and collision detection of labels are based on a ranked list [BF00, AHS05, MD06], [VTW12, VFW12]. They usually assume that the prioritization was already done in a preprocessing step. For instance, Luboschik et al. [LSC08] mention a ranking for GIS based on the feature type. Mote [Mot07] mentions a ranking based on the population of cities or the Google page ranking when displaying web searches. Moreover, Stein and Décoret [SD08] present a GPU-based real-time approach for labeling a 3D scenery. However, to compensate the drawbacks of their greedy approach, an Appolonius diagram defines the label placement order.

Conclusion. Most research on interactive labeling systems is based on the optimal placement and collision detection of labels (see Fig. 3, (violet)). However, there is almost no literature about the prioritization schemes used in existing systems. In most approaches, the ranking is defined by the importance of the map feature or set with simple characteristics, e.g. the distance to the camera. Hence, to better grasp related filtering systems, we decided to survey commercial navigation systems.

2.1. Survey: Filtering of Labels in Commercial Navigation Systems

Filtering of labels is an essential component of current navigation systems. However, in most cases it is unknown how the filtering of labels works and which algorithms are internally used. Hence, we conducted a survey with several navigation devices, namely, the TomTom Go 940 Live, Garmin nüvi 765T, Falk F10, Becker Z108 and the Navigon Select App for iOS.

2.1.1. Quality: Readability

In our first test, we evaluated the quality of the labeling. For every device, we counted in two different cities at eight locations all labels displayed in the map. We counted the labels at different zoom levels: the default zoom while driving and the maximum zoom level. Then, for each label, we checked it's readability. This survey can be see in Table 1.

Readability of Labels. As can be seen in Table 1, only the TomTom device achieved a readable labeling at every zoom level. All the other devices rendered labels, which collided or became clipped by the outer edges of the screen.

Navigation System	Default zoom level		Maximum zoom level	
	Total #	Readable #	Total #	Readable #
TomTom Go 940	9	9	4	4
Garmin nüvi 765T	9	5	2	2
Falk F10	12	7	5	4
Becker Z108	11	7	6	4
Navigon select	14	10	6	3

Table 1: Results of our survey of the readability of labels at different zoom levels in commercial navigation systems. This table shows the total number of labels and the number thereof of readable labels. The numbers are averaged over eight different locations.

2.1.2. Filtering Criterias

In the following tests, we tried to comprehend which criteria are used in the filtering of labels. This was done by following the same navigation route in the city of Munich in Germany with each device.

Repetitions of Road Labels. Not a single device repeats the road names, even if there would be enough screen-space.

Filtering: Horizontal Roads. All devices seem to prioritize roads which are rendered horizontally in screen-space. Garmin tries to show only intersecting roads.

Current Road Label. TomTom and Falk always show the current road on the map. The other devices display the current road as a separate layer at the bottom of the screen.

Filtering: Route vicinity. Only TomTom and Garmin use sometimes the route as a filtering criteria. On the other hand, Falk, Becker and Navigon try to squeeze as many labels as possible on the screen.

Filtering: Distance to user. Only Garmin and Becker prioritize labels near to the current user's position.

2.1.3. Conclusion

Every system uses it's own combination of filtering criteria. However, not a single one uses all criteria to create a good labeling layout with all important labels.

3. 3-STAGE FILTERING SYSTEM

The key idea of our approach is to create a ranking of all labels by computing a score for each label. This score depends on the label's type and the spatial relationship to the map environment. Using this priority list, we can intelligently select and filter the displayed labels. The result can be seen in Fig. 1 and Fig. 4.

Our filtering system depicted in Fig. 3 is composed of 3 stages:

1. *Configuration* – The first stage configures the filtering system depending on the current situation. For example, we can configure the system to output the four most important road labels if we drive along a route. Moreover, when we are searching the map for restaurants, we could restrict the labels to restaurant POIs and the surrounding roads.
2. *Label Ranking* – The second stage creates a ranking of all existing labels using a scoring system. It is computed using scoring modules called *evaluators*. The following Section 4 describes all types of modules with their score computation.
3. *Rule-Based Selection* – The last stage is a rule-based system which uses the ranking from last stage and predefined rules to select the labels to display. This stage enforces rules to ensure a specific behavior, e.g. limiting the maximum amount of labels.

Our approach distinguishes the following categories:

- City Labels, point features.
- Point Of Interest (POI), point features.
- Road Labels, line features.

City and POI labels describe a point feature and have a fixed 2D position on the map. Road labels describe a line feature stored as a 2D polyline [VTW11]. The road's label can be placed along the entire polyline composed of linear street segments.

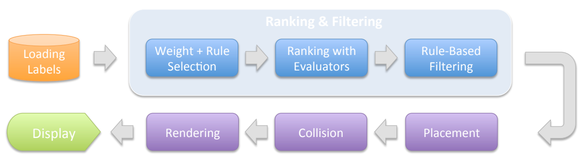


Figure 3: Overview of the entire labeling pipeline: After loading the labels from the map database (orange), the 3-stage filtering system (blue) presented in this paper filters the labels. Then, the selected labels are sent for placement onto the screen (violet). At runtime, we compute if any collision between labels occurs to finally render them to the display (green).

4. LABEL RANKING: EVALUATORS

We evaluate the score of a single label with a ranking system composed of evaluator modules. Each label is sent to each evaluator to compute an arbitrary score. Finally, the scores of a label i are summed up to create a final score $score(i)$.

4.1. Evaluators

We define evaluators based on the following criteria:

- Category Evaluator: score is based on the label's road or city or POI class
- Placement Evaluator: score is based on the past label's visibility to the user
- Distance Evaluator: score is based on the distance between the label and the user's position
- Route Evaluator: score is based on the vicinity of the label to the route
- Driving Direction Evaluator: score relates to the driving direction
- Road Angle Evaluator: score is based on the relationship between the driving direction and road direction
- Road Length Evaluator: score is based on the ratio between the road segment and the label's length

Furthermore, each evaluator specifies separately if higher or lower scores are better.

The presented evaluators were created mainly to rank labels of a 3D navigation map. Transferring our system to other application areas should be easily possible but would require a re-evaluation of each evaluator and possibly requires adding new ones.

4.1.1. Category Evaluator

This basic evaluator scores labels based on the category of their corresponding map element, i.e. the road class or the city's importance. These categories are precompiled into the map database and fetched at runtime. Hence, the resulting score is:

$$score(i)_{cat} = score(i)_{db} \tag{1}$$

Usually, highways have the highest score, followed by federal roads and main roads. Streets of lower importance, such as ordinary urban roads or walking paths, receive a lower score. The larger a city is, the more likely it is probable that it is known to the driver and thus contributes to a better orientation. Therefore, capitals and major cities return high scores. The less important a city is, e.g. in terms of population number, the lower the score gets.

4.1.2. Placement Evaluator

The placement evaluator is the key element to achieve a temporally coherent label filtering. The idea of this evaluator is, that already displayed labels are boosted through a higher score. Such a scoring prevents jittering, e.g. when labels become visible for a brief moment and disappear when another label achieves a slightly higher score.

$$score(i)_{place} = \begin{cases} 1 & \text{if label } i \text{ visible} \\ 0 & \text{else} \end{cases} \tag{2}$$

4.1.3. Distance Evaluator

The distance evaluator prioritizes labels around the user's location, e.g. the current car position (CCP). Therefore, the score is given by the distance between the label and the user's current position.

For the user's position p_{ccp} and the label i with position p_i the score is computed as follows:

$$score(i)_{dist} = \|p_i - p_{ccp}\| \tag{3}$$

4.1.4. Route Evaluator

This evaluator is used when an active route guides the user to his destination. The goal is to achieve a greater focus on labels positioned closer to the route. Therefore, for each candidate position p_i of a label i, the shortest distance d to the route is calculated. This distance does not map linearly to a score, since a label that is twice as far away of the route is not necessarily half as important. Instead we use a logarithmic function to create a focus tube around the route.

For label i with position p_i and the shortest distance d of label i to the route we get the following score:

$$d = \|p_i - \text{route}\| \tag{4}$$
$$score(i)_{route} = log_d * \sqrt{d} \tag{5}$$

4.1.5. Driving Direction Evaluator

The driving direction evaluator focuses on the labels ahead of the user. This is achieved by computing the deviation of a label's position from the direction of travel.

The driving direction is given by a 2D angle α_{ccp}. Moreover, we compute the vector between the user's position p_{ccp} and the label's position p_i. This is transformed to an angle α_{label_i}. We compute the differential angle spanned between the driving direction and the label's position.

With the normalized deviation $\hat{p}$ and the corresponding angle α_{label_i} we get:

$$\hat{\mathbf{p}} = (p_{ccp} - p_i)/\|p_{ccp} - p_i\| \tag{6}$$

$$\alpha_{label_i} = atan2(p_y, p_x) \tag{7}$$

$$score(i)_{direction} = \|\alpha_{ccp} - \alpha_{label_i}\| \tag{8}$$

We define the angle to be 0° when the label is in front of the user, 90° states that it's on either side and 180° if it's directly opposite to the direction of travel.

4.1.6. Road Angle Evaluator

The goal of this evaluator is to increase the score of labels from roads stretching vertically to the driving direction. Such roads mainly include crossroads and intersections that are especially relevant for navigation maneuvers. Even if this is a coarse simplification, we assume that most of these roads can be reached directly from the current route. In contrast, parallel roads can usually not be reached without entering another road first.

To compute the score, the direction of travel α_{ccp} is determined. Then, for each label candidate position p_i, the angle α_{road} of the corresponding road segment is computed. The smallest differential angle α_{angle} is used to create the score $score(i)_{angle}$ as follows:

$$\alpha_{angle} = \|\alpha_{road} - \alpha_{ccp}\| \tag{9}$$

$$score(i)_{angle} = \begin{cases} cos(\alpha_{angle}) & \text{if } \alpha_{angle} \leq 90° \\ 0 & \text{if } \alpha_{angle} > 90° \end{cases} \tag{10}$$

The advantage of this approach is that we do not need a complex graph-based structure of the road network to find intersecting roads. However, this evaluator still needs the road's geometry, i.e. a polyline. This means that neither POIs or city labels can be scored.

4.1.7. Road Length Evaluator

The road length evaluator prevents the placement of labels which names are longer than their corresponding road in screen-space. If the label would still be shown, it would hide the road, suggest a longer road and even encompass on neighboring roads.

The score $score(i)_{length}$ is based on the screen-space ratio between the length of the name compared to the projected road's length. First, we project the road's polyline into screen-space and compute the length l_{road}. Second, we compute the screen-space length of the rendered label's name l_{label}. If the label is shorter than the projected road segment, it receives a perfect score of 1. If the name is longer, the score is given by the ratio of the road's length and the label length.

$$score(i)_{length} = \begin{cases} 1 & \text{if } l_{label} \leq l_{road} \\ l_{road}/l_{label} & \text{else} \end{cases} \tag{11}$$

Again, this evaluator works only for roads and no other label categories can be considered.

4.2. Final Score

We have computed for every label i and each evaluator k the $score(i)_k$. This score is normalized to $score(i)'_k$ as described in Section 6.2. Moreover, we introduce weights w_k to configure the influence of each evaluator k. Each normalized score $score(i)'_k$ is factored with his weight w_k and summed up to a final score for label i:

$$score(i)'_k = \frac{score(i)_k}{\max\limits_{1\leq j\leq n}(score(j)_k)} \quad \text{(n: \#labels)} \tag{12}$$

$$score(i)_{global} = \sum_k score(i)'_k * w_k \quad (k \in \text{evaluator}) \tag{13}$$

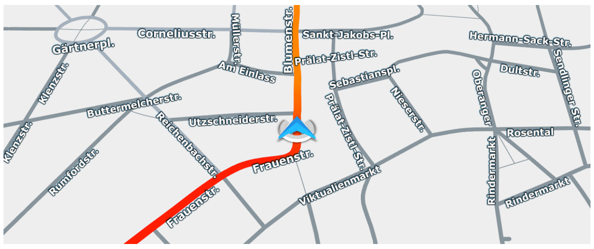
Computing this for every label results in a global ranking needed for the intelligent selection described in the following Section.

5. CONFIGURATION & RULE-BASED SELECTION

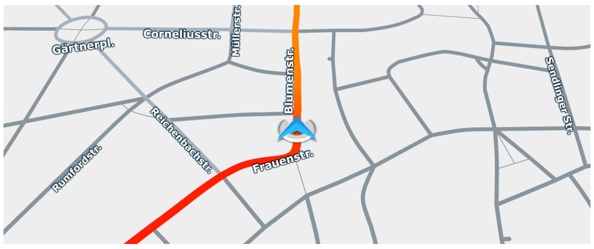
Selecting labels solely based on the ranking can not ensure a desired result. We solve this problem by introducing two methods: Based on the current context and the desired result we (a) configure the evaluators' weights and (b) instate predefined rules to select labels.

5.1. Configuration: Weights

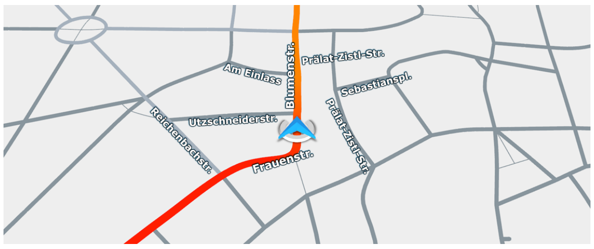
It is necessary to change the effect of the individual filters in specific situations. Using weights, the influence of a filter on the overall score can be changed. In different modes, different weights are needed. For instance, it makes no sense to use the route evaluator when there is no route. Analogously, when the user is exploring the map, the distance and direction evaluators become irrelevant. In the following Table 2, a configuration example of the weights w_k for the evaluators k can be found.



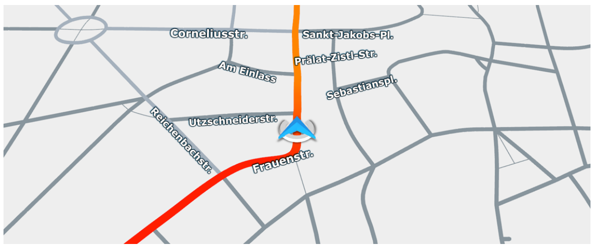
(a) No filtering.



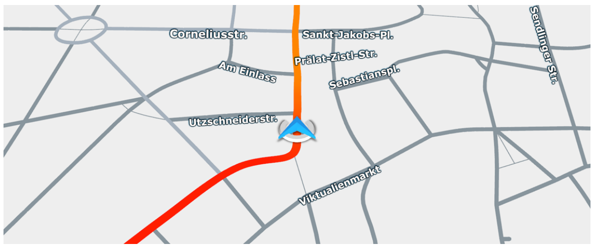
(b) Category Evaluator: Filtering based on the label's type.



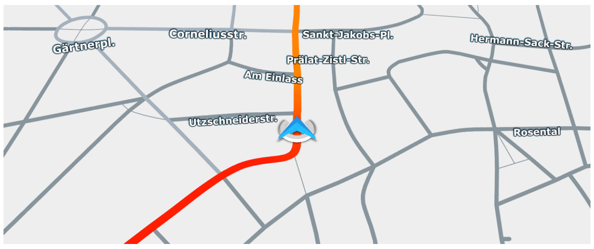
(c) Distance Evaluator: Filtering based on the distance to the user.



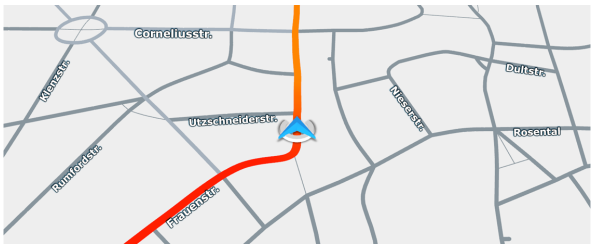
(d) Route Evaluator: Filtering based on the route's vicinity.



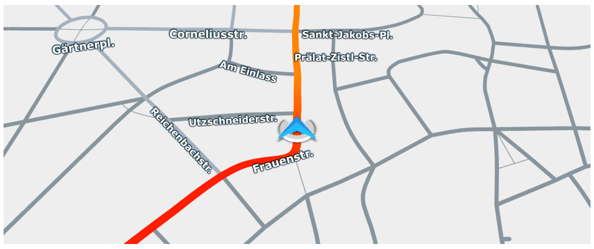
(e) Driving Direction Evaluator: Filtering based on driving direction.



(f) Road Angle Evaluator: Filtering based on driving vs. road direction.



(g) Road Length Evaluator: Filtering based on name vs. road length.



(h) All: Filtering based on all evaluators.

Figure 4: Our approach is based on evaluators ranking the labels with the help of map and user properties. This ranking is then used to filter the set of labels. In this figure, the current user's position is marked with the blue arrow and the user's navigation route is colored in orange. To display the impact of every evaluator, we limit the label number to eight and let a single evaluator process all labels.

5.2. Rule-Based Selection

The last stage of our system is a rule-based selection of labels based on the overall configuration and the ranking generated by the evaluators. These rules can overrule the ranking and ensure a specific behavior. We instate the following rules:

Repetition of Labels. At predefined zoom thresholds, we allow the repetition of labels for longer line features, e.g. motorways, rivers and country borders.

Maximum Number of Labels. We limit the maximum number of labels to be displayed.

Zoom-dependent Rule. Depending on the current map zoom level we hide or display certain label categories. Generally speaking, we remove predefined road or city classes when zooming out. For instance, when zooming out, we stop showing road names. In contrast, when zooming closely into the map, we stop showing country names.

Special Labels. We also instate a special rule to ensure the display of special labels. In a navigation system, some labels should always be shown, e.g. the start, the destination, and the current road.

Evaluator	Information Mode Weight	Route Mode Weight
Category	0.8	0.5
Placement	0.6	0.7
Distance	0.0	1.0
Route	0.0	1.0
Driving Dir.	0.0	0.5
Road Angle	0.7	0.5
Road Length	0.9	0.8

Table 2: Two configuration examples of the evaluator's weights. In the route mode, we have an active route and a current user's position. Hence, all labels around the user, the route and intersecting roads are important. In the information mode, we do not have a route and the exploration of the map with all it's labels becomes the most important task. Setting weights to 0.0 disables irrelevant evaluators.

6. THE LABEL SCORING SYSTEM

Until now, we assumed that labels have a single position. However, line features like roads are composed of several line segments. Their annotation can be placed at any position along these segments.

6.1. Road Labels: Candidate Scores

As can be seen from the given examples, most evaluators require a fixed position for a label, e.g. to determine if a label is near the route. For point features like city and POI labels, each evaluator returns a single score because they are stored with a single position in the map database. However, road labels could be placed at any position along the road's geometry. We simplify this to achieve real-time computation: for each road label we store all its linear road segments. Such road segments have a limited resolution and do not always match the curvature of the road as illustrated in Fig. 5. We allow the placement of road labels at every mid-point of a road segment.

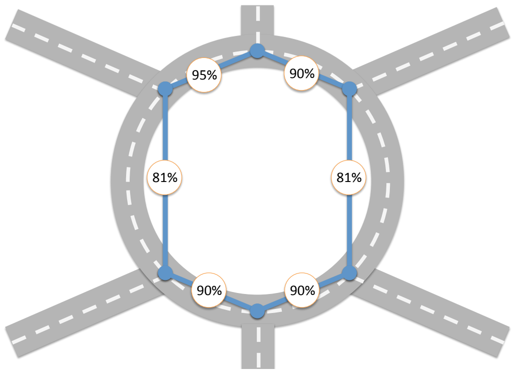


Figure 5: Road segments as candidates for the placement of a roundabout's label. Possible placement positions for road labels exists only at segment mid-points.

Candidate Scores. For each road label, we store one total score that determines the priority of the entire road and a second set of candidate scores, one for each segment, that determines the best placement position. Both score types are independent and cannot be compared with each other. The total score for road labels is calculated by taking the maximum score of all it's candidates.

As a result, the label placement system needs to check for a given label if it can be placed at a candidate position. It checks all candidate positions in descending sequence of its candidate scores. With both score types the label filtering system can be kept independent from other parts of the label placement system. Information about label collisions are not needed and all labels can be processed independently from each other.

Hence, for road labels and if the evaluator requires a position, we need to compute multiple candidate scores – one for each segment's mid-point. Evaluators which are unrelated to the label's position, i.e. the placement evaluator, just distribute their score to all segments.

Example. Consider the case depicted in Fig. 6 where labels for Street 1 and Street 2 need to be placed: Label *Street 1* (belonging to the left road) has the highest priority with a score of 95%. Therefore, it's the first label to be placed at C1 (green) into the screen. Label *Street 2* with a score of 93% has three possible candidate positions (blue): The left segment with the highest candidate score (C1) of 99%, C2 with 50% and C3 with 0%. We would like to place the label *Street 2* at C1 but it fails due to collision with the label *Street 1*. Therefore, because C2 (blue) represents the second best score, the label *Street 2* places itself onto C2.

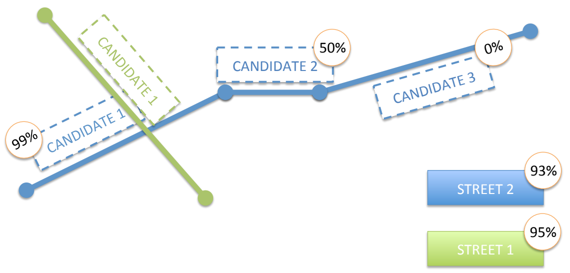


Figure 6: Road labels store their overall score (bottom, right) and additional candidate scores to determine the best possible placement along a road.

6.2. Score Normalization and Weighting

In a second step, we normalize all scores. The total score for each label is calculated by taking the weighted sum of all resulting scores as shown in Table 3(a). Normalization is done by dividing by the maximum score over all labels of each evaluator. For road labels consisting of multiple segments, we take the score from the best segment, i.e. depending on the evaluator either the minimum or maximum (see Sec. 4.1). Candidate scores for road labels are evaluated in a very similar fashion: Normalization is done by dividing by the maximum candidate score over all segments for each evaluator (see Table 3(b)).

7. RESULTS

The implementation of this filtering system was programmed with C++ and OpenGL 3.2 and runs on Windows 7 (x86). In this section we analyze the runtime behavior of the system followed by the results from the conducted expert study.

7.1. Performance

In our application, the number of labels n the system is processing varies between ≈ 500 in sparely and ≈ 9000 in densely populated areas. This high variation of n is caused by several properties: (a) level-of-detail of the map, (b) zoom level, and (c) the map database vendor. On average, we observed a range of ≈ 1000 to ≈ 4000 labels to be filtered.

The Route Evaluator is the most demanding evaluator of our approach because it requires nearest neighbor queries to the route. These queries can be computed in $O(\log m)$ time [AMN+98] with m being the number of positions of the route's polyline. In our implementation reasonable m with $m < 10.000$ do not impact the performance. Overall, the filtering system's complexity is $O(n \log m)$.

The performance results may vary depending on the activated evaluators, configuration settings and applied rules. The timing in ms for two different configurations can be seen in Fig. 7. Assuming a medium-class hardware Intel Core 2 Quad CPU clocked at 2.66 GHz, our system is capable of filtering 1000 labels within 12 ms on average. In our implementation, this is computed in

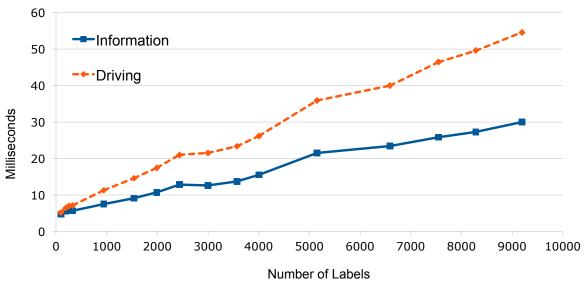


Figure 7: Performance measurements: We measured the computation time of the labeling pipeline in two different configuration: (stippled) driving mode with all evaluators enabled and (solid) information mode where the route and distance evaluator are disabled.

the rendering thread. In extreme scenarios (> 2000 labels), this could impact the overall rendering framerate. Therefore, for an enhanced performance, this system could be executed in a separate thread. This is possible because our approach is completely decoupled from the loading, placement, collision and rendering parts (see Fig. 3, violet steps).

7.2. Expert Study

We conducted an expert study at a research facility to evaluate the resulting filtering layout.

Candidates. We selected five candidates aged between 30 and 40 years with a strong background in automotive navigation: three male engineers, one person with a design and user experience background, and a psychologist specialized in user experience. Everyone had experience with commercial 3D navigation systems.

Study Design. As an introduction, we showed movies with a camera following a preset navigation route and our active label filtering. We requested the candidates to think about the shown selection of labels.

The first part of the study was designed to evaluate and compare different configurations of the filtering system. We showed four printed maps with the results of our system. Each time, the configuration of the filtering system was changed. Then, the candidates had to score the labeling quality depicted in each map on a scale 1..10 (10=best). In the second part of the study, they had to write labels manually on a printed map depicting a road network with a navigation route and a current car position icon (CCP). In the last part, we asked them which map and user criteria should influence the label filtering of a navigation map.

7.2.1. Results

Part 1 – Scoring the Filtering. On average, our filtering approach depicted in the printed maps was scored 6.62 (all scores: 6.67, 6.83, 7.33, 7.0, 5.25). The first map with 8 labels in route mode had an average score of 7.2. The second map with 14 labels in information mode scored 5.8. When comparing the route and the information mode filtering, the route mode always received higher averaged scores (first map 6.8 to 6.4, second map 7.75 to 6.5).

We deduce that our filtering approach was well received and that maps with less labels are preferable.

Part 2 – Writing Labels. Fig. 8 depicts the results of the manual placement of labels by our experts. We encoded the number of votes to different colors: At least three of the candidates voted for green labels, one of five voted for the white labels, and red labels did not receive any votes.

We deduce that the experts prefer labels around the user's position and on crossing roads.

Table 3: Normalization of the label's scores computed by the evaluators.

(a) Scores for k evaluators and n labels.

Evaluator	Label 1	$\cdots$	Label n
1 : *Category*	$score(1)_1$	$\cdots$	$score(n)_1$
$\vdots$			
k : *Length*	$score(1)_k$		$score(n)_k$
$\mathbf{score(i)_{global}}$	$\sum_{j=1}^{k}\left(\frac{score(1)_k}{\max\limits_{1\le j\le n}\left(score(j)_k\right)}w_k\right)$	$\cdots$	$\sum_{j=1}^{k}\left(\frac{score(n)_k}{\max\limits_{1\le j\le n}\left(score(j)_k\right)}w_k\right)$

(b) Candidate scores for a road label for k evaluators over m segments.

Evaluator	Segment 1	$\cdots$	Segment m
1 : *Category*	$score(1)_1$	$\cdots$	$score(m)_1$
$\vdots$			
k : *Length*	$score(1)_k$		$score(m)_k$
$\mathbf{score(i)_{seg}}$	$\sum_{j=1}^{k}\left(\frac{score(1)_k}{\max\limits_{1\le j\le m}\left(score(j)_k\right)}w_k\right)$	$\cdots$	$\sum_{j=1}^{k}\left(\frac{score(n)_k}{\max\limits_{1\le j\le m}\left(score(j)_k\right)}w_k\right)$

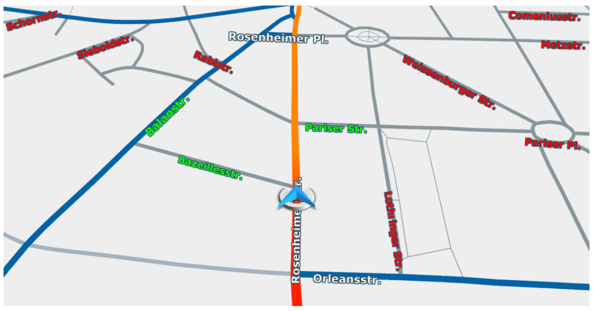


Figure 8: Printed map area we showed to our candidates. The current navigation route is displayed in orange. Each expert had to place labels on an empty map (without labels). The most voted labels are green.

Part 3 – Questions. First, we asked how many labels should be placed in a map. All experts agreed on a very low number of labels, (1 of 5) to display the current road only, and (1 of 5) to display at most 6 road labels.

Then we asked, which criteria are important for a good label placement. All experts defined the vicinity of the route as important and (4 of 5) the next crossroads. (2 of 5) mentioned the driving direction, (1 of 5) the road class, and (1 of 5) the road's length. Then, (2 of 5) mentioned landmarks and (1 of 5) well-known roads to help orientation. Districts (1 of 5), the current road (1 of 5), distance to destination area (1 of 5) were also mentioned.

Moreover, we asked how important are road labels behind the current user's position. The majority of experts (3 of 5) stated they are not important.

Finally, we asked if repetitions of road labels would be useful at higher zoom levels. Only (2 of 5) experts stated they are useful if the road branches or the course becomes unclear.

Conclusion. The results of this study shows the importance of the distance to the driving route and the current user's position on the label selection strategy. Moreover, roads crossing the current road are an important factor. All statements indicate a preference to display a minimum amount of labels. On top, the filtering system should be modular to satisfy even more criteria.

Finally, even if our filtering system was well received, we increased the weights of the route and distance evaluators to better fit the new findings. The resulting weights can be seen in Table 2.

8. CONCLUSION

In this paper, we presented an approach to filter labels in navigation maps. We introduced a 3-stage label filtering pipeline. First, it computes a score using multiple evaluators based on user's and map properties. This creates a ranking for all existing labels. Then, based on these results, predefined rules selects which labels should be displayed. With this system, we can achieve all our goals set in Section 1: The ranking creates a metric for the importance of labels in navigation maps. Then, we can limit the number of labels to a fixed amount depending on the current navigation context using the rules from Section 5. Temporal coherency is achieved with a placement evaluator (see Section 3). Moreover, our system is highly flexible as we can change the prioritization metric at runtime. Finally, the filtering system can easily be adapted to new applications.

On an Intel Core 2 Quad CPU at 2.66 GHz, our system is capable of filtering on average 1000 labels within 12 ms. Finally, the candidates of our expert study approved the clear labeling layout created by our approach.

In further research, we would like to find heuristics to detect unclear road courses where label repetions would be useful. Then, we plan to evaluate our approach while driving in a real world scenario. This should give more insights on the quality of the priority metric. Furthermore, we would like to include the user's preferences as an additional evaluator module.

9. REFERENCES

[AF07] Alvarez, G. A. and Franconeri, S. L. How many objects can you track?: Evidence for a resource-limited attentive tracking mechanism. *Journal of Vision*, 7(13):14, 2007.

[AHS05] Ali, K., Hartmann, K., and Strothotte, T. Label layout for interactive 3d illustrations. *Journal of the WSCG*, 13(1):1–8, 2005.

[AMN$^+$98] Arya, S., Mount, D. M., Netanyahu, N. S., Silverman, R., and Wu, A. Y. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *J. ACM*, 45(6):891–923, November 1998.

[BDY06] Been, K., Daiches, E., and Yap, C. Dynamic map labeling. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):773–780, 2006.

[BF00] Bell, B. A. and Feiner, S. K. Dynamic space management for user interfaces. In *Proceedings of the 13th annual ACM symposium on User interface software and technology*, UIST '00, pages 239–248. ACM, 2000.

[BRL09] Bertini, E., Rigamonti, M., and Lalanne, D. Extended excentric labeling. In *Proceedings of the 11th Eurographics / IEEE - VGTC Conference on Visualization*, EuroVis'09, pages 927–934. Eurographics Association, 2009.

[Imh75] Imhof, E. Positioning names on maps. *The American Cartographer*, 2(2):128–144, 1975.

[LSC08] Luboschik, M., Schumann, H., and Cords, H. Particle-based labeling: Fast point-feature labeling without obscuring other visual features. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1237–1244, November-December 2008.

[MD06] Maass, S. and Döllner, J. Efficient view management for dynamic annotation placement in virtual landscapes. 4073:1–12, 2006.

[Mil56] Miller, G. A. The magical number seven, plus or minus two: some limits on our capacity for processing information. *Psychological review*, 63(2):81, 1956.

[Mot07] Mote, K. Fast point-feature label placement for dynamic visualizations. *Information Visualization*, 6(4):249–260, 2007.

[SD08] Stein, T. and Décoret, X. Dynamic label placement for improved interactive exploration. In *Proceedings of the 6th international symposium on Non-photorealistic animation and rendering*, NPAR '08, pages 15–21. ACM, 2008.

[Tat00] Tatemura, J. Dynamic label sampling on fisheye maps for information exploration. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '00, pages 238–241. ACM, 2000.

[VFW12] Vaaraniemi, M., Freidank, M., and Westermann, R. Enhancing the Visibility of Labels in 3D Navigation Maps. In *Progress and New Trends in 3D Geoinformation Sciences*, Lecture Notes in Geoinformation and Cartography, pages 23–40. Springer, 2012.

[VTW11] Vaaraniemi, M., Treib, M., and Westermann, R. High-Quality Cartographic Roads on High-Resolution DEMs. *Journal of WSCG*, 19(2):41–48, 2011.

[VTW12] Vaaraniemi, M., Treib, M., and Westermann, R. Temporally Coherent Real-Time Labeling of Dynamic Scenes. In *Proceedings of the 3rd International Conference on Computing for Geospatial Research and Applications*, COM.Geo '12, pages 17:1–17:10. ACM, 2012.