

Musto, A., Stein, K., Eisenkolb, A., Röfer, T., Brauer, W., Schill, K. (2000). From Motion Observation to Qualitative Motion Representation. In: Freksa, Ch., Brauer, W., Habel, Ch., Wender, K. F. (Eds.): Spatial Cognition II. Lecture Notes in Artificial Intelligence 1849. Springer. 115-126.

From Motion Observation to Qualitative Motion Representation

Alexandra Musto¹, Klaus Stein¹, Andreas Eisenkolb², Thomas Röfer³,
Wilfried Brauer¹, and Kerstin Schill²

¹ Technische Universität München, 80290 München, Germany
{musto, steink, brauer}@in.tum.de

² Ludwig-Maximilians-Universität München, 80336 München, Germany
{amadeus, kerstin}@imp.med.uni-muenchen.de

³ Universität Bremen, Postfach 330440, 28334 Bremen, Germany
roefertzi.de

Abstract. Since humans usually prefer to communicate in qualitative and not in quantitative categories, qualitative spatial representations are of great importance for user interfaces of systems that involve spatial tasks. Abstraction is the key for the generation of qualitative representations from observed data. This paper deals with the conversion of motion data into qualitative representations, and it presents a new generalization algorithm that abstracts from irrelevant details of a course of motion. In a further step of abstraction, the shape of a course of motion is used for qualitative representation. Our approach is motivated by findings of our own experimental research on the processing and representation of spatio-temporal information in the human visual system.

1 Introduction

It is a matter of fact that humans are capable of storing transient data, e.g. for tracking an object through occlusion [14], or for the reproduction of a path formed by a moving black dot on a computer screen (see section 4.1). For this prima facie trivial capability a bunch of problems has to be solved by the human visual system. Here we are concerned with the integration of representations with different time stamps. This general logistic problem arises especially when complex spatio-temporal information has to be classified, e.g., a sophisticated gesture in Japanese Kabuki Theatre, but is immanent to formal-technical applications too. As an example, a locomotion-capable agent who has explored route knowledge about some environment and who has to communicate a possible route to a human listener in a human format has to transform the raw data (obtained by odometry) into some *qualitative* format (see [1] for an approach on a similar problem).

When representing an observed course of motion qualitatively, the most important step is abstraction. This paper presents a new possibility of how to abstract from irrelevant details of a course of motion in order to represent the coarse structure qualitatively. A further step of abstraction is the segmentation and classification of the gained qualitative representation. In this step any detail of the course of motion concerning

spatio-temporal variation is omitted and it is represented merely as a sequence of motion shapes. Both, generalization and segmentation are essential aspects of the qualitative modeling of motion representation. This qualitative stage is part of an integrated approach [13] which ranges from the modeling of qualitative representation to the modeling of spatio-temporal information on earlier stages of the human visual system. In this context, we also propose a memory model on the level of visual concept formation which tries to explain human capabilities of reproducing the spoor of a moving object.

2 Qualitative Motion Representation

Representing a course of motion qualitatively means abstracting from irrelevant details to get only the big directional changes which can then be translated in some qualitative categories for direction. So, an observed course of motion should be simplified by some generalization algorithm (e. g. the one described in 3) that suppresses small zigzags and deviations (fine structure) and returns a course of motion that contains only the big directional changes and the overall shape information (coarse structure).

The directional changes can then be represented qualitatively, as well as the distance between each directional change. We can do this by mapping the numeric values for distance and angle into a corresponding interval, such collapsing numerical values which we do not want to distinguish into a single qualitative value.

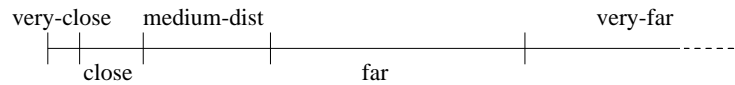


Fig. 1. A discretization of the distance domain

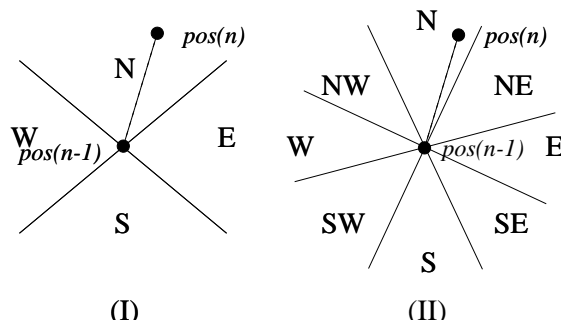


Fig. 2. I: 4 direction relations – II: 8 direction relations

Figure 1 and 2 show how this can be done. In the direction domain, we model only positional change, so we can use the well known orientation relations from qualitative spatial reasoning. A discretization with intervals of equal size can be systematically derived on any level of granularity like shown in [6]. Possibilities of how to discretize the distance domain are shown in [2]. The qualitative values for distance can e. g. be named *very-close*, *close*, *medium-dist*, *far*, *very-far*, the values for direction *north*, *south*, *east*, *west*.

The resulting qualitative representation of a course of motion is a sequence of qualitative motion vectors (QMV's). The vector components are qualitative representations of direction and distance like shown above. A qualitative representation of motion speed can be added by simply mapping the ratio of time elapsed from one directional change to the next and the distance covered in this time into qualitative intervals.

The mapping of the observed course of motion into a QMV sequence should best take place after a suitable generalization algorithm has been applied. However, it is also possible to do it the other way round, that is to generate a QMV representation containing coarse and fine structure of the course of motion and to apply the generalization algorithm to this QMV sequence. All developed algorithms (see [7] and section 3) can be used with numerical vector sequences as well as with QMV sequences. In fact, the algorithm presented here only smoothes the course of motion when applied to a numeric vector sequence, but generalizes only when applied to a QMV sequence, as we shall see.

3 Generalization

By generalization, a motion track is simplified and the fine structure is suppressed, stressing on the coarse form, containing the information on the major directional changes and the overall shape of the course of motion. Such an operation can be useful when processing motion information with a limited memory capacity, e.g. in biological systems, but also in technical systems like the Bremen Autonomous Wheelchair, where generalization is used for navigation on basis of recorded locomotion data [10].

Analytical generalization algorithms as used by, e.g., cartographers [3, 5], are mostly not suitable for our purpose, since the computational effort of computing a smooth curve is not necessary with our discrete representation, but so considerable that it would perhaps not be possible to generalize "on the fly" in a real time environment. Other algorithms like the one described in [8] need the complete motion path, and therefore they are also not suitable for processing on the fly.

So we developed two incremental generalization algorithms, the ε - and Σ -Generalization which are described in [7]. An alternative approach is based on a spatio-temporal memory model introduced in [12], which is a necessary prerequisite for the representation and processing of intricate motion information in the biological visual system.

In contrast to the sequential approaches described in [7] it enables the parallel access to all motion information gathered within the last few timecycles, i. e. to a sequence (of a certain length l) of successive motion vectors, in order to extract and generalize

information about a course of motion. The following algorithms are based on the main idea of this model. Parallel access to l successive vectors is equal to a window of size l sliding on the motion track:

Step n : Access to $v_n, v_{n+1}, \dots, v_{n+l-1}$
 Step $n + 1$: Access to $v_{n+1}, v_{n+2}, \dots, v_{n+l}$

Technically this is a FIFO-buffer of size l .

Linear Sliding Window Generalization. A simple idea of using the local information of few successive vectors for generalization is to calculate the averages of k successive vectors (here $k = l$: compute the average of all parallel accessible vectors):

$$w_t = \frac{1}{k} \sum_{i=s}^{s+k-1} v_i, \quad w_{t+1} = \frac{1}{k} \sum_{i=s+1}^{s+k} v_i, \quad \dots,$$

where w_t is the corresponding result vector for the sliding window from v_s to v_{s+k-1} . The index t marks the middle of the sliding window. With odd k holds $t = s + \frac{k-1}{2}$. Thus we get a new vector sequence where each w_t is the average of k successive vectors v_i .

Nested Sliding Window Generalization. [9] used the idea of the spatio-temporal storage and parallel access for a different approach: the motion vectors $v_n, v_{n+1}, \dots, v_{n+l-1}$ are used to compute the motion vector ($u_{n,n+l-1}$) for the whole motion of the observed object within the timeslice of the spatio-temporal memory:

$$u_{n,n+l-1} = \sum_{i=n}^{n+l-1} v_i, \quad u_{n+1,n+l} = \sum_{i=n+1}^{n+l} v_i, \quad \dots$$

Thus we get a sequence of overlapping vectors: $u_{n,n+l-1}, u_{n+1,n+l}, \dots$

In this sequence, v_{n+l-1} is covered by the combined vectors $u_{n,n+l-1}$ to $u_{n+l-1,n+2l-2}$, v_{n+l} by $u_{n+1,n+l}$ to $u_{n+l,n+2l-1}, \dots$

Now each w_t is computed as the average of all the $u_{a,b}$ covering v_t :

$$w_t = \frac{1}{k'} \sum_{a \leq t \leq b} u_{a,b} = \frac{1}{k'} \sum_{i=t}^{t+l-1} u_{i-l+1,i},$$

where k' is the number of input vectors: Each vector $u_{a,b}$ is the sum of l vectors v_i , and l vectors $u_{a,b}$ are added, therefore $k' = l \cdot l = l^2$.

Generic Sliding Window Generalization. Having a closer look on the algorithm above shows two nested sliding windows used:

$$w_t = \frac{1}{k'} \sum_{i=t}^{t+l-1} u_{i-l+1,i} = \frac{1}{l^2} \sum_{i=t}^{t+l-1} \sum_{j=i-l+1}^i v_j$$

$$= \frac{1}{l^2}(v_{t-l+1} + 2 \cdot v_{t-l+2} + \dots + l \cdot v_t + (l-1) \cdot v_{t+1} + \dots + v_{t+l-1}).$$

The inner sliding window (sum of the v_i) is of size l , the outer one used for computation of the average of size $k = 2l - 1$. Rewriting indices leads to

$$w_t = \frac{4}{(k+1)^2}(v_1 + 2 \cdot v_2 + \dots + 2 \cdot v_{k-1} + v_k),$$

a weighted sum over k successive vectors v_i .

In general any set of weights can be used, so the generic form is

$$w_t = \frac{1}{\sum \alpha_i}(\alpha_1 v_1 + \alpha_2 v_2 + \dots + \alpha_k v_k),$$

with a sliding window of size k , and with the linear and nested sliding window generalizations as special cases with $\alpha_i = 1$ (linear) and $\alpha_1 = 1, \alpha_2 = 2, \dots, \alpha_{k-1} = 2, \alpha_k = 1$ (nested).

Processing of Start and End of a Motion Track. The start and end of the track have to be processed by an adapted (shortened) sliding window for starting the window with the first k vectors would shorten the whole track: Imagine a track of length n processed with a sliding window of length n , then the whole track would be shortened to $\frac{1}{n}$ th of its length.

Properties The sliding window generalization algorithm does not (as the ε - and Σ -generalization, see [7]) transform a vector sequence with many small movement vectors into one with a few large vectors to simplify the motion track, because the output track contains the same number of vectors as the input.



Fig. 3. Sliding Window Generalization with size $k=3$

The track is smoothed due to calculating the average of neighboring vectors, so sudden small but rapid changes in direction are lessened as shown in Fig. 3 with linear sliding window of size 3.

If we use this algorithm on vector sequences with qualitative directions, the track is not only smoothed, but generalized: the mapping from numeric vectors in qualitative intervals gives a threshold below which all successive vectors are mapped to the same direction. If the track is sufficiently smoothed, little deviations are so totally suppressed. So and by fine-tuning the parameters α_i , the sensitivity of the generalization can be

chosen. As can be seen in figure 3, the gradient of the smoothed track is determined by choice of the α_i : With $\alpha_1 = \alpha_2 = \alpha_3 = 1$ (linear), the gradient is the same at all neighboring points of the original kink. With $\alpha_1 = \alpha_3 = 1, \alpha_2 = 2$ (nested), the gradient is the steepest at the point, where the kink was in the original track. With an arbitrary choice of the α_i , other operations can be done on the track. E.g., with $\alpha_1 = 1, \alpha_2 = \alpha_3 = 0$, all vectors in the window would be pushed one position forward.

4 Segmentation

4.1 Human Visual Perception.

When asked, humans can reproduce the motion of single black dot moving along a (invisible) path on an otherwise white background of a computer screen (*target path*). To our knowledge there is no experimental data as to how precise such a stimulus can be reproduced. As a first step, we devised an experiment to examine as to how well humans can reproduce simple trajectories. In an experiment subjects (Ss) had to watch a target path and subsequently reproduce its spoor by moving the finger along the surface of a touch screen. First results of the experiments are shown in Fig. 4.

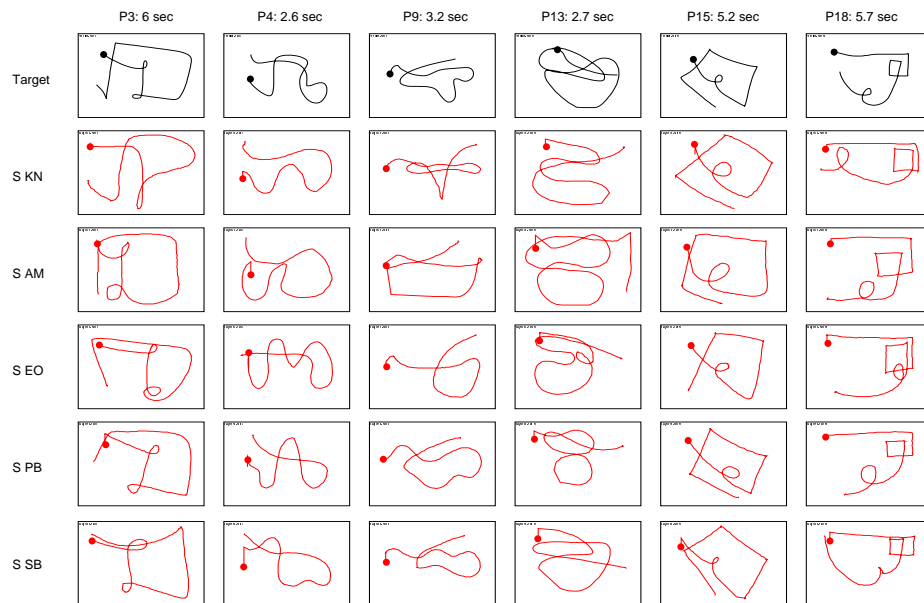


Fig. 4. Ss saw a target path (first row). Immediately after it had finished they had to reproduce its spoor by moving the finger along the screen surface (touch screen). The performance of 5 Ss for 6 target paths (columns) is depicted.

Given the state of research this subsection has necessarily a speculative character. Some of the trajectories seem to be reproduced better than others (compare target path P13 and P18). It also seems that some patterns are adequately “generalized” and classified (P18: square loop, round loop) whereas others provide more difficulties, like most reproductions of P9 which can be considered “over-generalized”. It may be the case that certain subpatterns of a motion stimulus can be matched “on the fly” with existing internal templates, like, e.g. “loop” or “kink”. It is then possible that a dynamic structuring based on forms, i.e. *segmentation* can be performed whereas in other cases no template is found and the resulting representation is poor.

However, a difference in reproduction quality may mean that for some patterns it is simply easier to generate the respective motor commands whereas others are difficult to “express”. It may also mean that the motor commands are of the same level of difficulty but the internal representation of some patterns is worse than others. If the template is in an eidetic form, i.e., there is no “higher” description based for instance on a form vocabulary, there is no reason why the curved pattern P13 in fig. 4 is reproduced worse than P18. It is in most cases a nontrivial task to determine the quality of the reproduction. Also, we know of no way to exclude the influence of the motor component in our reproduction task.

4.2 Segmentation Algorithm

Inspired by the idea that courses of motion might be segmented into pieces that are matched with existing templates and so classified, we developed an algorithm that segments a QMV sequence such that it can be represented very compactly by a sequence of motion shapes.

The basic step for this is to find out where one shape ends and the next begins. The algorithm described below will transform a QMV sequence to the basic shapes straight line, 90° curves to right and left and backward turns in different sizes, where the size of a curve is its radius. In case of eight direction relations, we also have 45° curves to right and left as well as 135° curves to right and left. More complex shapes¹ are built by combination of these basic ones. We restrict our illustrating example to the use of four direction relations. The case of eight relations is analog.

The input for the algorithm is a generalized QMV sequence, so we do not have to worry about minimal movements; all movements below a minimal size are filtered by generalization. Thus, we can take each turn in the QMV sequence to be a curve in the shape representation. Since we do not know where a curve starts and ends, we would not know, for example, whether a QMV sequence has to be segmented into a *left-turn*, *straight-line* and a *left-turn*, or into an *u-turn-left*. To get smooth shapes, we therefore made the decision to give every curve the maximum size possible, distributing the distances among all the curves starting with the smallest. Hence, we work on the representation of our QMV sequence as a track in the QMV linear space as described

¹ from a predefined shape vocabulary, containing shapes like *loop-left*, *loop-right*, *u-turn-left*, *u-turn-right*, *s-curve-left*, ...

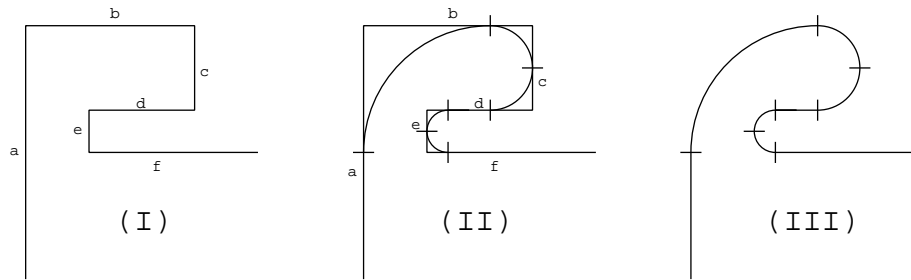


Fig. 5. Segmentation of a QMV sequence

in [7]. This track is made up of horizontal and vertical straight lines as shown in figure 5 (I).

The segmentation of each line will contain two curves, except for the first and the last line. We sort the lines by size, every line with two curves counts only half its length, for it will be split into two. In our example we get e, c, d, b, f, a . Then we start at the smallest line e . It has two curves, so it is split in the middle and the attached lines d and f are shortened by this amount (half size of e , as shown in figure 5 (II)). Then the changed lines are sorted in with their new length (d now has just one curve from c to d and so is sorted in with full length whereas e and f have no curves left and are not sorted in at all), and we start from the beginning (for our example this means c, b, d, a). If we proceed on a line with only one curve left we give the curve the maximum size possible, as shown at line b). We are done when no lines are left.

Figure 5 (III) shows the graph of our shape representation of the QMV sequence. It reads

```
<straight-line middle-size> <right-turn big> <right-
turn middle-size> <right-turn middle-size> <straight-line
small> <left-turn small> <left-turn small> <straight-line
middle-size>.
```

Now we can build more complex shapes. Two attached 90° turns of same size and direction (like at the lines c and e) can easily be combined to an u-turn, four attached 90° turns of same size can be combined to a loop, etc.

In a further step, we can suppress straight lines with very small size compared with the adjacent curves and take curves with similar size as equal. So, we can build complex shapes from simple shapes that are not matching the template completely.

The synthesis of some shapes in the example sequence results in

```
<straight-line middle-size> <right-turn big> <u-turn-
right middle-size> <straight-line small> <u-turn-left
small> <straight-line middle-size>.
```

This segmentation algorithm also works well when processing a motion sequence on the fly with a lookahead of 2 to 3 vectors. The only problematic case for processing on the fly is when the vectors in the sequence get constantly shorter, which is a rare case. It can nevertheless be processed with small error, as shown in [15].

4.3 Memory Model

Can the shape representation be used to explain how humans reproduce the spoor of a course of motion? Given the preliminary character of the research results in section 4.1, this section is surely speculative, too.

Nevertheless, we suggest a model that, given a segmentation of a course of motion in basic shapes, constructs more complex shapes on the fly and with a limited memory capacity.

We assume that at this level of processing, we have a similar memory model for parallel access like the model proposed in [12] for an earlier stage, with a few modifications:

- We assume not a simple FIFO-Buffer like used in the generalization algorithm, but more a kind of a stack: Incoming shapes on top of stack are “popped” and combined to a more complex shape as soon as possible, and the more complex shape is then again “pushed” on top of the stack.
- Our preliminary results strongly suggest that, if the course of motion is too “long” to be memorized correctly², not the beginning is forgotten, but features that are not very salient or pronounced. Therefore, in our model, if the motion sequence becomes too long for being hold in the buffer, not the oldest shapes are removed, but the shapes that are least salient. This is the reason why parallel access must be granted.

To find out what “salient” means, experiments have to be conducted. For the moment, we assume that a shape is the more salient, the more complex it is. Complex means here that the shape is constructed from simpler shapes. The more basic shapes participate in a complex shape, the higher is its complexity, but also the more information is coded in a single buffer element. Therefore it seems to be a reasonable strategy to supersede the simple shapes that carry the least information. This basic measurement of saliency can be weighted by several factors, e. g. age of the shape in the buffer and size of the shape. It seems also that shapes with pronounced corners are more salient than others. Another factor seems to be that the first and the last shape are remembered better due to primacy/recency effects, which should also be considered.

Figure 6 shows the construction of the complex shape sequence from the course of motion in figure 5 in a memory buffer of size 5.

This memory structure with parallel access to all elements also allows that a higher processing level extracts global features from the elements stored in the buffer, for example the shape of the overall arrangement of the shape sequence in space, similar as described in [13] for a lower processing stage.

² Where too long means not the temporal duration, but the number of shapes that have to be held in memory

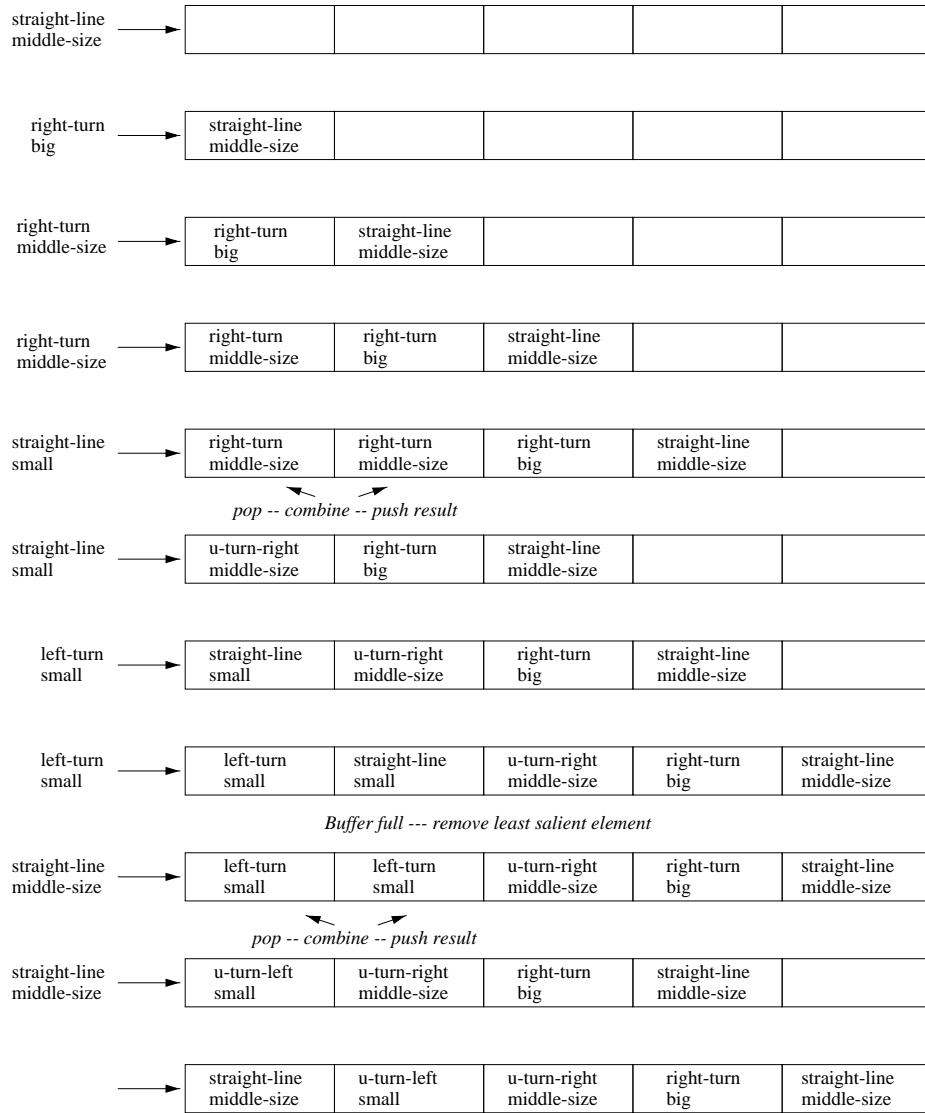


Fig. 6. Memory buffer for shape representation

A problem with our approach is that with the QMV representation we are unable to distinguish between a smooth curve and a sharp corner. That means that very important information is lost. Of course we could work with a numeric representation of the course of motion at the lower level, but then segmentation is much more difficult than with QMVs. Another possibility is to store additional information about the smoothness of a directional change with the QMVs.

5 Results and Future Work

In [9], the ε - and the nested sliding window generalization were applied concurrently to the same navigation approach in a simulation of the Bremen Autonomous Wheelchair [11]. The generalization was there used for navigation of the wheelchair (a semi-autonomous robot) along pre-taught routes. The track of the taught route was generalized and stored, and the track of the actual route was generalized on the fly and compared to the stored track. So, it could be determined whether the wheelchair has found its way correctly or made a mistake in following the route. In this work, both algorithms were applied to a semi-qualitative representation of the tracks, i. e., directions were represented qualitatively and distances numerically. With the qualitative directions, the results of both algorithms were satisfying and very similar, only the sliding window generalization always produces smooth tracks without sharp corners which is not the case with the ε -generalization.

Up to now, we have no experimental data whether our generalization algorithms correspond to anything that takes place in motion perception. This is a topic for future work.

Neither do we have experimental results on the memory buffer model so far. In this context, we have to conduct first experiments on the saliency of motion shapes. From these findings, we can then define a saliency function to determine which of the shapes in the buffer can be forgotten first. Then we can apply our algorithm to the same trajectories human subjects have to reproduce in experiments to see whether similar results are produced, which would validate our model.

References

1. Elisabeth André, Guido Bosch, Gerd Herzog, and Thomas Rist. Characterizing trajectories of moving objects using natural language path descriptions. In *Proceedings of the 7th ECAI*, volume 2, pages 1–8, Brighton, UK, 1986.
2. Eliseo Clementini, Paolino Di Felice, and Daniel Hernández. Qualitative representation of positional information. *Artificial Intelligence*, 95(2):317–356, 1997.
3. Douglas and Peucker. Algorithms for reduction of the number of points required to represent a digitized line or its caricature. *The Canadian Cartographer*, 10/2, 1973.
4. Christian Freksa and David M. Mark, editors. *Spatial Information Theory. Cognitive and Computational Foundations of Geographic Information Science. International Conference COSIT'99*, volume 1661 of *Lecture Notes in Computer Science*, Berlin, Heidelberg, New York, August 1999. Springer.
5. Emmanuel Fritsch and Jean Philippe Lagrange. Spectral representations of linear features for generalisation. In Andrew U. Frank and Werner Kuhn, editors, *Spatial Information Theory. A Theoretical Basis for GIS. European Conference, COSIT'95*, volume 988 of *Lecture Notes in Computer Science*, pages 157–171, Berlin, Heidelberg, New York, September 1995. Springer.
6. Daniel Hernández. *Qualitative Representation of Spatial Knowledge*, volume 804 of *Lecture Notes in Artificial Intelligence*. Springer, Berlin, 1994.
7. Alexandra Musto, Klaus Stein, Kerstin Schill, Andreas Eisenkolb, and Wilfried Brauer. Qualitative Motion Representation in Egocentric and Allocentric Frames of Reference. In Freksa and Mark [4], pages 461–476.

8. Jonas Persson and Erland Jungert. Generation of multi-resolution maps from run-length-encoded data. *International Journal of Geographical Information Systems*, 6(6):497–510, 1992.
9. Hans Raith. Methoden zur Analyse von Bewegungsinformationen in technischen Systemen und Implementierung eines biologischen Modells zur Repräsentation spatio-temporaler Informationen. Diplomarbeit, 1999. Institut für Informatik, Technische Universität München.
10. Thomas Röfer. Route navigation using motion analysis. In Freksa and Mark [4], pages 21–36.
11. Thomas Röfer and Axel Lankenau. Architecture and applications of the Bremen Autonomous Wheelchair. In P. P. Wang, editor, *Proceedings of the Fourth Joint Conference on Information Systems*, volume 1, pages 365–368. Association of Intelligent Machinery, 1998.
12. Kerstin Schill and Christoph Zetsche. A model of visual spatio-temporal memory: The icon revisited. *Psychological Research*, 57:88–102, 1995.
13. Kerstin Schill, Christoph Zetsche, Wilfried Brauer, Andreas Eisenkolb, and Alexandra Musto. Visual representation of spatio-temporal structure. In B. E. Rogowitz and T. N. Pappas, editors, *Human Vision and Electronic Imaging. Proceedings of SPIE*, pages 128–137, 1998.
14. Brian J. Scholl and Zenon W. Pylyshyn. Tracking multiple items through occlusion: Clues to visual objecthood. *Cognitive Psychology*, (38):259–290, 1999.
15. Klaus Stein. Generalisierung und Segmentierung von qualitativen Bewegungsdaten. Diplomarbeit, 1998. Institut für Informatik an der TU München.