

Generic creation of landscapes and modelling of complex parts of road networks

**Armin Kaussner
Christian Mark
Hans-Peter Krueger
Hartmut Noltemeier**

Interdisziplinäres Zentrum fuer Verkehrswissenschaften an
der Universitaet Wuerzburg
Psychologisches Institut
Roentgenring 11
D-97070 Wuerzburg

Abstract

In databases based on the architecture developed at the IZVW, the road network can be changed during simulation. Changes can be made depending on drivers' behaviour, drivers' state, or triggered by the supervisor of an experiment. We present two improvements of this architecture. The first deals with automatic generation of landscapes surrounding the roads. When designing the database for his experiment, the researcher assigns certain landscape types (farmed, wooded, inhabited, ...) to each side of the road. Thereof, the landscape including terrain heights and cultural features is generated during simulation. Specified by a few parameters the landscape surrounding a road can change each time the road comes into the sight of the driver. The second improvement facilitates modelling of complex parts of the road network, like junctions, roundabouts etc. This parts are represented very detailed. The representation contains information used by the simulation of traffic: The run of the lanes, tangent angles, curvatures and the relations of the lanes among each other.

Introduction

If a driving simulator is used as a research tool - as it is the case at the IZVW (Center for Traffic Sciences) - a special architecture of the database is needed which is quite different from the well-known concept of fixed data bases. The basic requirement for the research data base is that the road network can be changed during simulation. These changes can be triggered by the behavior of the driver, his/her state or by other conditions defined by the researcher. In the introduction we give a short summary of this architecture (for more details see [KGKN01]). The remainder of this paper describes two issues that expand the flexibility of this new database concept.

The road network is represented as a graph $RN = (V, E)$. The set of nodes is partitioned into two parts: $V = COURSES \cup AREAS$.

A node $v \in COURSES$ represents a continuous road of arbitrary length without turn-offs. Within v the number of lanes, their width, types and driving directions do not change.

A node $v \in AREA$ is a rectangular area containing lanes that represent a more complex part of the road network, e.g. conjunctions between $COURSES$ with different cross sections, crossways, drive-ups to highways etc. The modeling of $AREAS$ is addressed in the next section.

Each node in V consists of so called lane cells which, in turn, contain parametric curves in 3D space describing the run of the lanes. An example of a $v \in COURSES$ is shown in fig. 1.

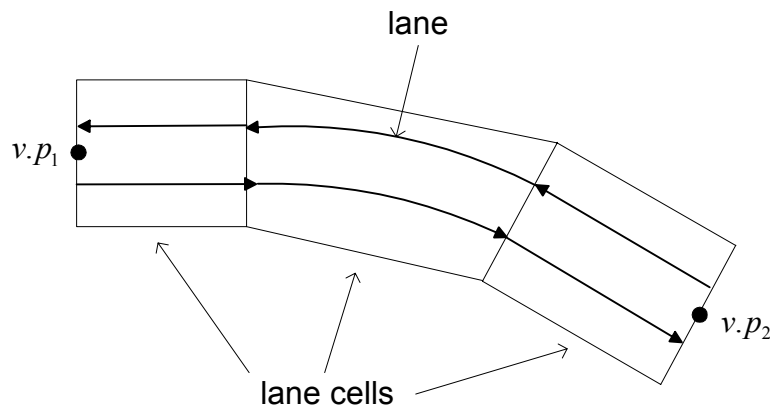


Fig. 1: Example of $v \in COURSES$. The node v consists out of three lane cells. Each lane cell holds two parametric curves that define the run of the lanes. The total number of lanes, their width and driving direction define the cross section of v .

The nodes in V are connected via so called ports. A $v \in COURSES$ has two ports, $v.p_1$ at the beginning of the roadway and $v.p_2$ at the end. A $v \in AREAS$ has an arbitrary number of ports $v.p_1, \dots, v.p_n$. Each port is placed at one of the sides (top, bottom, left, right) of the underlying rectangle.

An edge out of E always connects ports of two nodes: $E = \{(v.p_i, w.p_j) : v, w \in V\}$. From a topological view these connections can be ambiguous, as shown in fig. 2: $c_1.p_2$ connects simultaneously to $c_2.p_2$ and $a_1.p_3$, which is – geometrically – not possible. These ambiguities are resolved by an algorithm called geometric instantiation. It relies on the fact, that during the simulation the driver has a limited region of visibility. Only inside this region the road network has to be geometrically consistent (and therefore non-ambiguous). Outside this region the geometric restrictions must not be taken into account.

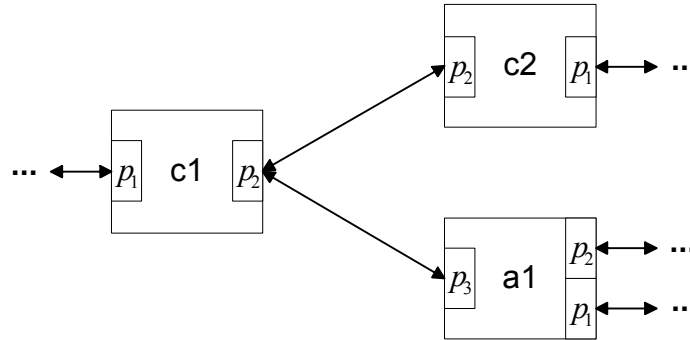


Fig. 2: Part of a road network that is ambiguous.

$$c_1, c_2 \in \text{COURSES}, a_1 \in \text{AREAS}$$

At each time point t_i of the simulation the geometric instantiation extracts a graph $RN^*(t_i) \subseteq RN$ out of RN . $RN^*(t_i)$ is geometrically consistent inside the drivers region of visibility. Starting from the drivers current position, the algorithm traverses RN until it reaches nodes which are not visible. If, during this process, a port of a node becomes visible and more than one edge emanates from this port, one of them has to be selected. Therefore, the algorithm evaluates conditions that are stored in the ports. When designing the database, the user can impose these conditions on measures taken during simulation (e.g. physiological and behavioural measures etc.). In fig. 2 the user would have to define conditions for $c_1.p_2$.

Automatic generation of landscapes

Overview

By using the IZVW database architecture, the researcher can design a database tailored for his problem. After having defined the road network, he has to create a surrounding landscape. Since he is not interested in the details of the landscape, he only wants to assign certain landscape types to nodes $v \in \text{COURSES}$ and control their basic appearance by a few parameters. At the moment he can select between the landscape types “inhabited”, “farmed” and “wooded”.

A landscape type consists of two parts: (1) A description of the terrain heights and (2) 3D models of objects that are placed on the terrain

Each time a node gets visible for the driver, the landscape is generated from these two parts. Thus, the landscape of a node can vary during simulation.

An algorithm similar to ours is given in [H01]. Although it models the distribution of objects more detailed, its major drawback is that it doesn't allow the manipulation of height data during simulation.

Landscape heights

Let $v \in COURSES$ be a node modeling a road segment. On each side of v a set of so called height functions $\{F_1(d), \dots, F_n(d)\}$, is defined. A height function $F_i(d)$ starts at a certain position $s_i \in [0, S]$ on the road, with S as the total length of the road modeled by v . It measures the elevation of a point (relative to the road surface) which has distance d to the road (fig. 3). The maximum value of d is chosen in a way that no intersections of two height functions on the inner side of a bend exist.

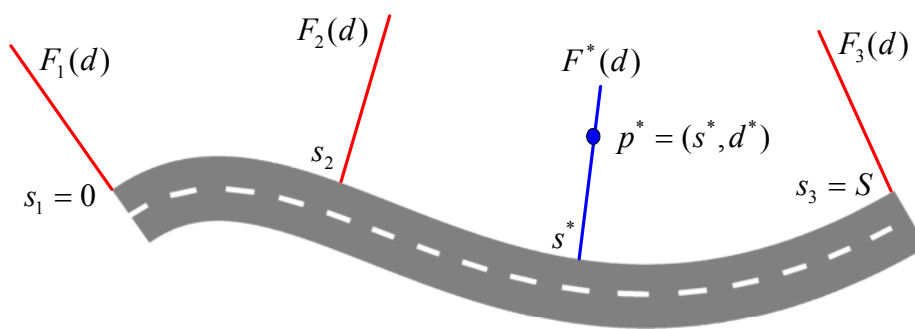


Fig. 3: A road segment with height functions on its left side.

For example, a simple definition of the height functions can be of the form

$$F_i(d) = G(d, d_g^{(i)}) \cdot \left(A_1^{(i)} \cdot \sin(f_1^{(i)} \cdot d \frac{2\pi}{1000}) + A_2^{(i)} \cdot \sin(f_2^{(i)} \cdot d \frac{2\pi}{1000}) + o_i \right) \quad (1)$$

A sum of two superposed sine waves and a constant offset o_i is multiplied by a smoothing function G , which has the following properties:

$$\begin{aligned} G(0, d_g^{(i)}) &= 0, \quad G(d_g^{(i)}, d_g^{(i)}) = 1 \text{ for } d \geq d_g^{(i)} \\ G'(0, d_g^{(i)}) &= G'(d_g^{(i)}, d_g^{(i)}) = 0 \end{aligned}$$

Independent from the slopes of the sine waves and the offset, this function ensures a soft increase of elevation up to a distance $d_g^{(i)}$ from the road.

Height curves can be defined in different ways. They have in common that they can be represented efficiently by storing a few "form parameters". The form parameters in eq. (1) are: $d_g^{(i)}, A_1^{(i)}, f_1^{(i)}, A_2^{(i)}, f_2^{(i)}$

Let $p^* = (s^*, d^*)$ be an arbitrary point that lies between two pre-defined height curves F_i and F_{i+1} , i.e. $s_i < s^* < s_{i+1}$. To determine the exact height of p^* , the parameters of the corresponding height curve F^* must be interpolated between the parameters of the two neighboring height curves (see fig.3).

Objects

The objects which are characteristic for a specific landscape type are organised in categories. Each category contains a set of 3D models. In addition, a landscape type stores constraints describing if objects of a certain category may overlap those of another category or not.

Fig. 4 shows the categories and constraints of a simple “farmed” landscape type.

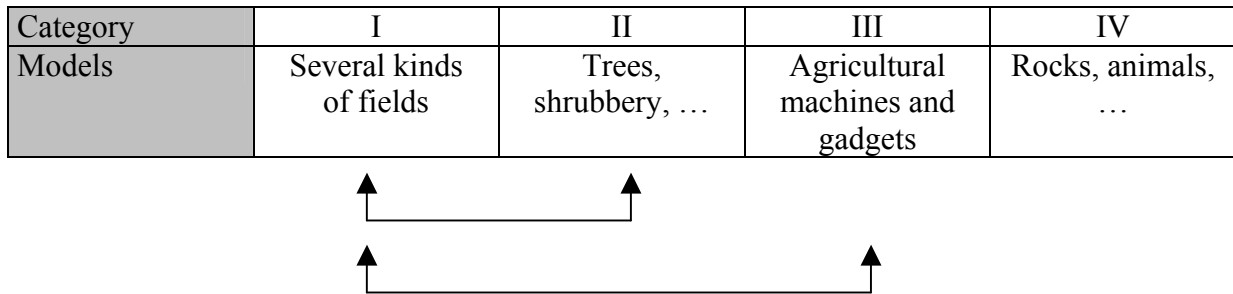


Fig. 4: A simple landscape type “farmed”. The arrows mean “must not overlap”

The landscape generation algorithm chooses objects out of each category and tries to place them in a way that

- the objects of a category have a certain density. Therefore, an interval is stored for each category. Each time a node gets visible the densities are drawn randomly out of these intervals.
- the constraints are maintained. This is done by using R-Trees (see [G84]) as a method for spatial indexing. This data structure allows queries like “is the space needed for the bounding box of object x already occupied by the bounding box of another object y”.

Modeling complex parts of the road network

Overview

Databases used in driving simulators usually store two correlated types of data: The first type describes the graphical representation of the road network and its surrounding landscape. This information is organized in terms of vertices, polygons and textures. In addition, there has to be logical information concerning the run of the lanes, their relations among each other, traffic rules etc. This logical road network (LRN) can be derived automatically from the graphical representation, as shown in [B00] or serve as a base for the graphical representation (see [CE99], [CE00]). In the IZVW database architecture it is used in the second way: First the LRN is modeled precisely and yields the graphical information. This section addresses how complex parts of the LRN, i.e. nodes $v \in AREAS$, can be defined in an intuitive way and represented efficiently.

Structure

A $v \in AREAS$ represents a rectangular area. At the sides of the rectangle there can be an arbitrary number of ports $v.p_1, \dots, v.p_n$. A node $v' \in COURSES$ which is connected to v via

this ports is required to have a certain cross section profile. Therefore, each $v.p_i$ stores information as shown in fig. 5.

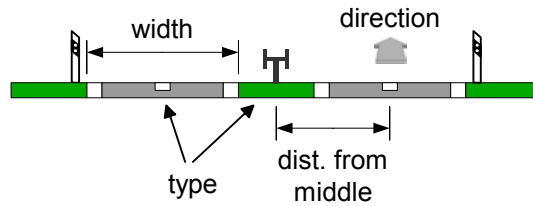


Fig. 5: Information about each lane of a node of type *COURSES*. This data is stored in the ports of *AREAS*.

A lane l has two ports called $l.begin$ and $l.end$. Similar to the road network, these lanes are the nodes of a graph $L = (V_L, E_L)$, with $E_L = \{(l.end, l'.begin) : l, l' \in V_L\}$. Fig. 6 shows a junction modeled in this manner.

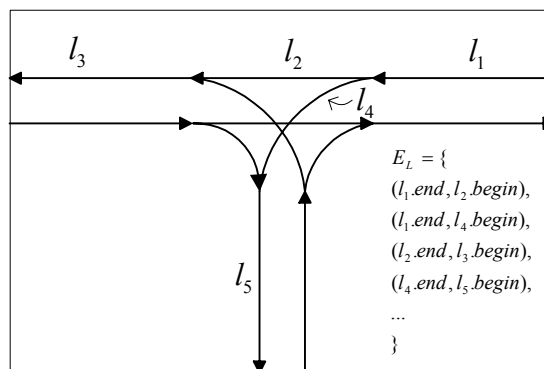


Fig. 6: Graph of lanes

If a port of a lane is not connected to another lane, it always lies on a port of the area. For example, in fig. 6 the ports $l_1.begin, l_3.end$ and $l_5.end$ are associated with area ports.

Analytic description of lanes

The **run of a lane** is represented analytically as parametric curve $l.p : [0, S_l] \rightarrow \mathbb{R}^2$ with respect to a coordinate system that has its origin in the center of the areas' rectangle. Thereby S_l denotes the length of the lane. From the parametric curve the tangent angle (denoted as $l.\alpha(s)$) and the curvature (denoted as $l.\kappa(s)$) can be derived. Three types of parametric curves exist:

- Straight lines, specified by their start and end points.
- Simple bends, specified by their start and end points $l.p(0), l.p(S_l)$ as well as the tangent angles $l.\alpha(0), l.\alpha(S_l)$ in these points (fig. 7). Simple bends always connect to straight lines or segments of super-curves (below).

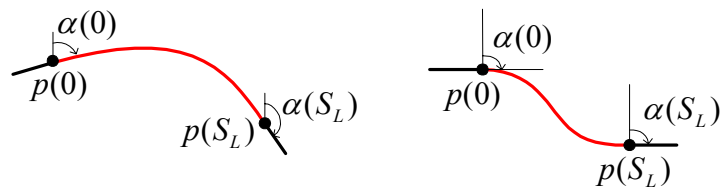


Fig. 7: Two examples of simple bends.

- Segments of super-curves. Some types of junctions can not be modelled using straight lines and simple bends. An example is a roundabout as shown in fig. 8. In this case, the circle is represented by a parametric curve $v.p(s)$ of length S , which is stored “globally” in the area $v \in V$ (there can be several super-curves stored in one area). A lane can represent a segment of this super-curve. The segment is specified by a pointer to a super-curve as well as by a parameter s_1 of the super-curve, which corresponds to the parameters 0 of the segment. Therefore, the parametric curve of the segment can be written as $l.p(s) = v.p(s + s_1)$. The functions for tangent angles $l.\alpha(s)$ and curvatures $l.\kappa(s)$ can be obtained in the same way.

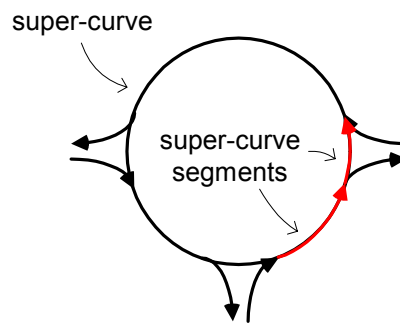


Fig. 8: A super-curve representing the circle of a roundabout. Each lane that lies on that circle is modelled as a segment of the super curve.

When defining the lanes of a $v \in AREAS$, the parametric curves of the type “simple bends” can be derived automatically. For example, the left side of fig. 9 shows the junction of fig. 6 as seen by the user when defining the LRN. The definition of the roundabout of fig. 8 can be seen in the right part of the figure. An algorithm replaces the red lines with simple bends. Since curves of this type always are connected with straight lines or segments of super-curves (both of which the tangent angles and the start/end points are known), they simply can be determined by interpolation.

Additional attributes of lanes

Several modules of the simulation which retrieve information from the database need supplementary information. We give two examples of such additional attributes.

The vehicles of the ambient traffic need to know the direction in which specific sequence of lanes will lead. This information is called “**heading**” of a lane $v.l_i$. It is stored as $v.l_i.headings \subseteq \{straight, left, right\}$. In fig. 9, let $v.l_i$ be the lane that is marked with \odot . Then

$v.l_i.headings = \{left, right\}$. The lane $v.l_j$ which is tagged with $\textcircled{2}$ has the headings $v.l_j.headings = \{straight, left\}$.

To facilitate defining **relations of lanes among each other** the lanes of a $v \in AREAS$ can be grouped in so called lane cells. Let $v.lc_1, \dots, v.lc_m$ denote the lane cells of v . Each lane of v is an element of exactly one lane cell. There are no empty lane cells. The assignment of lanes to lane cells has to be done in a way that the lane cells subdivide areas into logical regions.

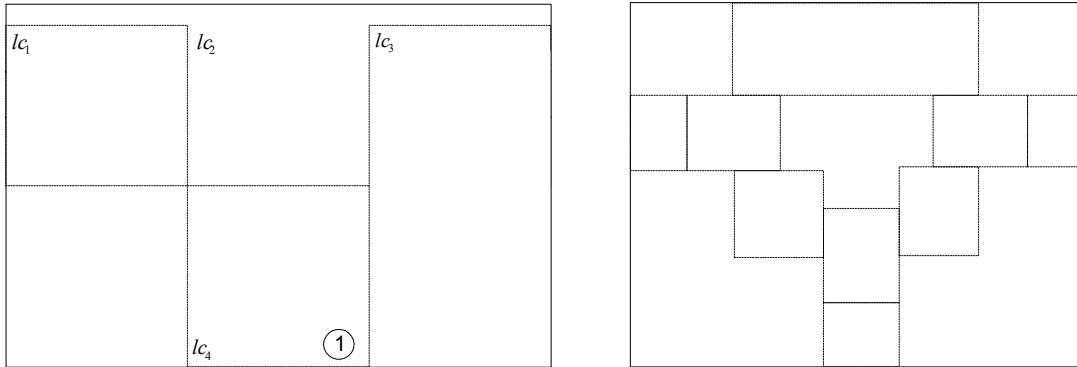


Fig. 9: Parametric curves of the type “simple bend” can be derived automatically (red). Lanes are grouped in lane cells (dotted rectangles).

The dotted rectangles in fig.9 give examples for such subdivisions: In the left junction the lane cells $v.lc_1, v.lc_3$ and $v.lc_4$ represent approaching/leading away lanes. Lane cell $v.lc_2$ contains the lanes of the actual intersection. The lanes of the roundabout in the right part of fig. 9 are grouped in the same way.

A lane has four sets, $v.l_i.Oncoming, v.l_i.CrossingLeft, v.l_i.CrossingRight$ and $v.l_i.LeadingAlong$. Each set contains lanes of the same lane cell as $v.l_i$.

Fig. 10 shows lc_2 of the junction in the left part of fig. 9.

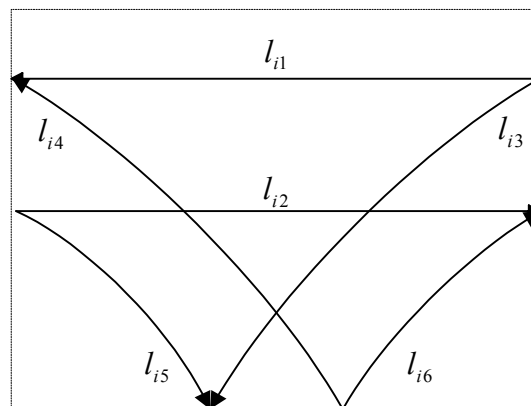


Fig. 10: lc_2 of the left part of fig. 9

The lane l_{i1} has the following relations:

$$v.l_{i1}.Oncoming = \{v.l_{i2}\}, v.l_{i1}.CrossingLeft = \{v.l_{i4}\}$$

The relations of $v.l_{i6}$ are:

$$v.l_{i3}.Oncoming = \{v.l_{i6}\}, v.l_{i3}.CrossingLeft = \{v.l_{i4}\}, v.l_{i3}.CrossingRight = \{v.l_{i2}, v.l_{i5}\}$$

Conclusion

We presented two concepts extending the usability of the IZVW database architecture.

The automatic generation of landscapes enables an user to easily define the landscape surrounding a road network: He assigns certain landscape types to parts of the road network and controls their appearance by a few parameters. The landscape is generated dynamically during simulation. This has the additional advantage, that the landscape surrounding the same part of the road network can change each time it gets visible for the driver.

Complex parts of the road network (e.g. junctions, roundabouts, drives to highways) can be modeled very detailed. The run of the lanes is represented via parametric curves. Positions, tangent angles and curvatures can be derived with arbitrary precision. In addition, several attributes can be stored for each lane. We gave two examples of attributes that are used for controlling the vehicles of the ambient traffic.

References

- [B00] A. C. Bailey: Advancement in logical road network design for the Leeds driving simulator, in Proc. of the Driving Simulation Conference, DSC2000, Paris, 2000
 - [CE99] O. Carles, S. Espié : Database generation system for road applications, in Proc. of the Driving Simulation Conference, DSC1999, Paris, 1999
 - [CE00] O. Carles, S. Espié : Multi-level environments modelling for road simulations, in Proc. of the Driving Simulation Conference, DSC2000, Paris, 2000
 - [G84] A. Guttman: R-TREES: A dynamic index structure for spatial searching, in Proc. ACM SIGMOD, 1984
 - [H01] J. Hammes: Modeling of ecosystems as a data source for real-time terrain rendering, in LNCS 2181, Springer, 2001
 - [KGKN01] A. Kaussner, M. Grein, H.-P. Krüger, H. Noltemeier: An architecture for driving simulator databases with generic and dynamically changing road networks, in Proc. of the Driving Simulation Conference, DSC2001, Sophia-Antipolis, 2001
-