Proving Safety of Traffic Manoeuvres on Country Roads*

Martin Hilscher, Sven Linker, Ernst-Rüdiger Olderog

Department of Computing Science, University of Oldenburg, Germany
{hilscher, linker, olderog}@informatik.uni-oldenburg.de

Abstract. We adapt the Multi-lane Spatial Logic MLSL, introduced in [1] for proving the safety (collision freedom) of traffic manoeuvres on multi-lane motorways, where all cars drive in one direction, to the setting of country roads with two-way traffic. To this end, we need suitably refined sensor functions and length measurement in MLSL. Our main contribution is to show that also here we can separate the purely spatial reasoning from the underlying car dynamics in the safety proof.

1 Introduction

The safety of road traffic can be increased by new assistance systems that are based on suitable sensors and communications between cars. Reasoning about car safety (collision freedom) is reasoning about hybrid systems involving car dynamics and spatial considerations. To simplify this reasoning, we proposed in [1] to separate the purely spatial reasoning from the car dynamics. To formalise this idea, we introduced a dedicated *Multi-lane Spatial Logic* (MLSL) for expressing spatial properties on multi-lane motorways, where the traffic is flowing on several lanes, but in one direction. We focused on the lane-change manoeuvre.

MLSL is inspired by Moszkowski's interval temporal logic [2], Zhou, Hoare and Ravn's Duration Calculus [3], and Schäfer's Shape Calculus [4]. MLSL is a two-dimensional extension of interval temporal logic, where one dimension has a continuous space (the position in each lane) and the other has a discrete space (the number of the lane). In MLSL we can, for example, express that a car has reserved a certain space on its lane. Safety then amounts to proving that under certain assumptions the reservation of different cars are always disjoint.

In this paper we turn to the more intricate setting of (multi-lane) country roads, with two-way traffic. We investigate the safety of an overtaking manoeuvre in the presence of opposing traffic. The original definition of MLSL only allowed for qualtitative reasoning, which is not aedequate for the definition of the overtaking protocol. Whenever a car changes into opposing traffic, it has to check beforehand whether the free space is sufficiently large for completing the manoeuvre. Hence we extend MLSL with the possibility to measure lengths.

* This research was partially supported by the German Research Council (DFG) in the Transregional Collaborative Research Center SFB/TR 14 AVACS.

Our findings are that modulo these small extensions, the setting of MLSL is well suited to cover this setting and manoeuvre.

Related Work. Safety on multi-lane motorways was investigated extensively in the context of the California PATH project, where manoeuvres of car platoons including lane change have been studied [5]. Here the car dynamics was an integral part of the safety reasoning. Safety in urban traffic scenarios has been studied in [6]. The manoeuvres include lane change, double lane change with opposing traffic, right and left turns. The analysis is based on an abstract graph representing two possibly conflicting car trajectories and on some simplifying assumptions like a constant speed of the cars involved.

The idea of *separating* the dynamics from the control layer is pursued in the controller design for hybrid systems. In [7] the authors introduce abstraction and refinement in a hierarchical design of hybrid control systems. In [8] the synthesis of control laws for piecewise-affine hybrid systems is introduced. In [9] controller patterns were proposed to simplify safety proofs for two cooperating traffic agents. Proving system correctness across a number of different abstraction layers continues a line pioneered in the ProCoS (Provably Correct Systems) project [10], in which He Jifeng made significant contributions.

The novelty of our approach is that the control layer is given by spatial properties formalised in MLSL. In this paper we extend this approach to country roads with two-way traffic. To this end, we refine the abstract traffic model of [1] by taking traffic directions into account, by defining new sensor functions, and by adding length measurement in the logic MLSL.

2 Abstract Model

Throughout this paper we work with an abstract model of (multi-lane) country roads with two-way traffic that emphasises spatial properties, but hides the car dynamic as much as possible. In this model a country road has an infinite extension with positions represented by the real numbers, and lanes are represented by natural numbers $0, 1, \ldots, n$. At each moment of time each car, with a unique identity denoted by letters A, B, $\ldots$, has its position *pos*, speed *spd*, and acceleration *acc*. On country roads cars can move in two directions, one with increasing position values, in pictures shown from left to right, and one with decreasing position values, in pictures shown from right to left.

The abstract model is introduced by allowing for each car only local views of this traffic. A view of a car E comprises a contiguous subset of lanes, and has a bounded extension. A view containing all lanes with an extension up to a given constant, the *horizon*, is called *standard view*.

What a car "knows" of its view is expressed by formulas in the multi-lane spatial logic MLSL introduced in [1]. It extends interval temporal logic [2] to two dimensions, one with a continuous space (the position in each lane) and the other with a discrete space (the number of the lane). Such a formula consists of a finite list of lanes, where each lane is characterized by a finite sequence of segments. A segment is either occupied by a car, say E, or it is empty (*free*). For

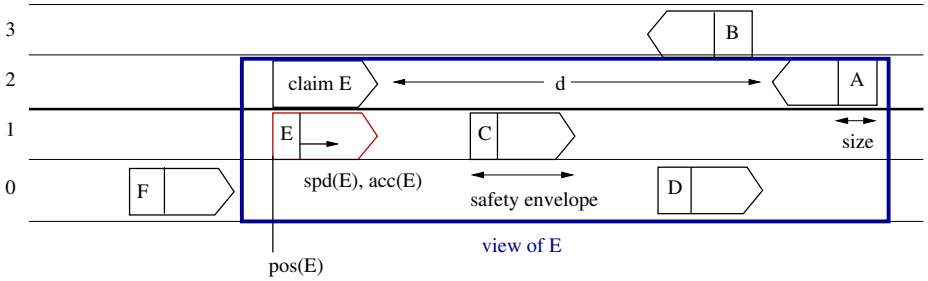

Fig. 1. View of car E covering a bounded extension of lanes 0, 1, and 2. Car E sees its own reservation and claim for preparing a change from lane 1 into the opposing traffic of car A on lane 2 to overtake car C ahead in lane 1. In this view, E does not see the cars B and F.

instance, in the view of car E shown in Fig. 1, the following formula ϕ holds:

$$\phi \equiv \left\langle \begin{array}{c} \mathit{free} \frown \mathit{cl}(E) \frown (\mathit{free})^d \frown A \\ \mathit{free} \frown \mathit{re}(E) \frown \mathit{free} \frown C \frown \mathit{free} \end{array} \right\rangle ,$$

where $\frown$ is the *chop operator* of interval temporal logic; it serves to separate adjacent segments in a lane. In the logic we can distinguish whether a car E has *reserved* a space in a lane ($\mathit{re}(E)$) or only *claimed* a space ($\mathit{cl}(E)$) for a planned lane change manoeuvre. We stipulate that reserved and claimed spaces have the extension of the *safety envelopes* of the cars, which include at each moment the speed dependent braking distances. The formula ϕ expresses also that there is a distance d between the claim of car E and the opposing car A.

We shall use such formulas as guards and location invariants of abstract controllers for car manoeuvres. In a technical realisation of such controllers, the properties that may appear in the formulae stipulate suitable sensors of the cars, for instance distance sensors.

2.1 Traffic snapshot

We recall definitions from [1], extending them where needed. Let $\mathbb{L} = \{0, \ldots, N\}$, for some fixed $N \geq 1$, denote the set of lanes, with typical elements l, m, n. We assume a globally unique identifier for each car and take $\mathbb{I}$ as the set of all such *car identifiers*, with typical elements $A, B, \ldots$. To formalise two-way traffic, we refine the setting of [1] in two ways. First, we assume a *border* $b \in \mathbb{L}$ such that traffic on the lanes $l \leq b$ normally drives in the direction of increasing values of $\mathbb{R}$ (from left to right), and traffic on the lanes $l > b$ normally drives in the direction of decreasing values of $\mathbb{R}$ (from right to left). Only on the lanes b and $b+1$ cars may temporarily drive in opposite direction to perform an overtaking manoeuvre. Second, we partition $\mathbb{I}$ into the sets $\mathbb{I}_{\rightarrow}$ and $\mathbb{I}_{\leftarrow}$, i.e., $\mathbb{I}_{\rightarrow} \cup \mathbb{I}_{\leftarrow} = \mathbb{I}$ and $\mathbb{I}_{\rightarrow} \cap \mathbb{I}_{\leftarrow} = \emptyset$. The subscripts indicate the *driving direction* of the cars. Cars in

$\mathbb{I}_{\rightarrow}$ drive from left to right, and cars in $\mathbb{I}_{\leftarrow}$ drive from right to left. For simplicity we assume $\mathbb{I}_{\rightarrow}$ and $\mathbb{I}_{\leftarrow}$ to be countably infinite.

Definition 1 (Traffic snapshot). *A* traffic snapshot $\mathcal{TS}$ *is a structure* $\mathcal{TS} = (res, clm, pos, spd, acc)$, *where* res, clm, pos, spd, acc *are functions*

- $res : \mathbb{I} \to \mathcal{P}(\mathbb{L})$ *such that* $res(C)$ *is the set of lanes that car* C *reserves,*
- $clm : \mathbb{I} \to \mathcal{P}(\mathbb{L})$ *such that* $clm(C)$ *is the set of lanes that car* C *claims,*
- $pos : \mathbb{I} \to \mathbb{R}$ *such that* $pos(C)$ *is the position of the rear of car* C *on its lane,*
- $spd : \mathbb{I} \to \mathbb{R}$ *such that* $spd(C)$ *is the current speed of car* C*,*
- $acc : \mathbb{I} \to \mathbb{R}$ *such that* $acc(C)$ *is the current acceleration of car* C*.*

Let $\mathbb{TS}$ *denote the set of all traffic snapshots.*

Definition 2 (Transitions). *The following* transitions *describe the changes that may occur at a traffic snapshot* $\mathcal{TS} = (res, clm, pos, spd, acc)$*. We use the overriding notation* $\oplus$ *of Z for function updates [11].*

$$\begin{aligned}
\mathcal{TS} \xrightarrow{t} \mathcal{TS}' \quad \Leftrightarrow \quad & \mathcal{TS}' = (res, clm, pos', spd', acc) \\
& \wedge \forall C \in \mathbb{I}\colon pos'(C) = pos(C) + spd(C) \cdot t + \tfrac{1}{2} acc(C) \cdot t^2 \\
& \wedge \forall C \in \mathbb{I}\colon spd'(C) = spd(C) + acc(C) \cdot t \qquad (1) \\
\mathcal{TS} \xrightarrow{\mathsf{acc}(C,a)} \mathcal{TS}' \quad \Leftrightarrow \quad & \mathcal{TS}' = (res, clm, pos, spd, acc') \\
& \wedge acc' = acc \oplus \{C \mapsto a\} \qquad (2) \\
\mathcal{TS} \xrightarrow{\mathsf{c}(C,n)} \mathcal{TS}' \quad \Leftrightarrow \quad & \mathcal{TS}' = (res, clm', pos, spd, acc) \\
& \wedge |clm(C)| = 0 \wedge\ |res(C)| = 1 \\
& \wedge \{n+1, n-1\} \cap res(C) \neq \emptyset \\
& \wedge clm' = clm \oplus \{C \mapsto \{n\}\} \qquad (3) \\
\mathcal{TS} \xrightarrow{\mathsf{wd\ c}(C)} \mathcal{TS}' \quad \Leftrightarrow \quad & \mathcal{TS}' = (res, clm', pos, spd, acc) \\
& \wedge clm' = clm \oplus \{C \mapsto \emptyset\} \qquad (4) \\
\mathcal{TS} \xrightarrow{\mathsf{r}(C)} \mathcal{TS}' \quad \Leftrightarrow \quad & \mathcal{TS}' = (res', clm', pos, spd, acc) \\
& \wedge clm' = clm \oplus \{C \mapsto \emptyset\} \\
& \wedge res' = res \oplus \{C \mapsto res(C) \cup clm(C)\} \qquad (5) \\
\mathcal{TS} \xrightarrow{\mathsf{wd\ r}(C,n)} \mathcal{TS}' \quad \Leftrightarrow \quad & \mathcal{TS}' = (res', clm, pos, spd, acc) \\
& \wedge res' = res \oplus \{C \mapsto \{n\}\} \\
& \wedge n \in res(C) \wedge\ |res(C)| = 2 \qquad (6)
\end{aligned}$$

The transitions allow for the passage of time (1), with cars moving along the road, and a change of acceleration (2). These two transitions model abstractly the dynamic aspects of cars. To prepare for a lane change, a car may *claim* a neighbouring lane, which can be thought of as setting the turn signal, while it is not already in progress of changing lanes or has set the turn signal already (3). A car may *reserve* a previously claimed lane (5). Also, it may withdraw claims (4) and reservations (6), as long as at least one lane remains reserved by a car.

2.2 View

In our safety proof we will restrict ourselves to finite parts of a traffic snapshot $\mathcal{TS}$ called views, the intuition being that the safety of manoeuvres can be shown using local information only.

Definition 3 (View). *A* view *V is defined as a structure $V = (L, X, E)$, where*

- *$L = [l, n] \subseteq \mathbb{L}$ is an interval of lanes that are visible in the view,*
- *$X = [r, t] \subseteq \mathbb{R}$ is the extension that is visible in the view,*
- *$E \in \mathbb{I}$ is the identifier of the car under consideration.*

A subview *of V is obtained by restricting the lanes and extension we observe. For this we use sub- and superscript notation: $V^{L'} = (L', X, E)$ and $V_{X'} = (L, X', E)$, where L' and X' are subintervals of L and X, respectively.*

For a car E and a traffic snapshot $\mathcal{TS} = (res, clm, pos, spd, acc)$ we define the standard view *of E as*

$$V_s(E, \mathcal{TS}) = (\mathbb{L}, [pos(E) - h, pos(E) + h], E),$$

where the horizon h is chosen such that a car driving at maximum speed can, with lowest deceleration, come to a standstill within the horizon h plus twice the distance it maximally takes to perform the overtake, i.e., the worst distance needed to pass a car and for changing lanes into the opposing traffic and back again. This way a car can be perceived early enough when planning an overtaking manoeuvre.

We can give a rough estimate for the horizon h. Let us assume that a car E driving at 100 km/h wants to overtake a slower car travelling at 80 km/h. Passing the slower car will then take about 17 seconds. In this time the car travels approximatly 500 m. Furthermore, we assume an overapproximation for the braking distance of 50 m for driving at 100 km/h and 40 m for 80 km/h. Then we can safely overapproximate h with 1.5 kilometers, i.e., twice the travelling distance plus $500m$. Thus h exceeds both braking distances plus the distance needed for two lane changes.

Sensor Function. In [1] we introduced a sensor function $\Omega_E : \mathbb{I} \times \mathbb{TS} \to \mathbb{R}_+$ for each car E which, given a car identifier C and a traffic snapshot, provides the length of the car C, as perceived by E. For country roads we need to change this definition taking the opposing driving directions of cars into account.

Definition 4 (New sensor function). *We define the new sensor function $\Omega_E^{new} : \mathbb{I} \times \mathbb{TS} \to \mathbb{R}$ as:*

$$\Omega_E^{new}(C, \mathcal{TS}) = \begin{cases} \Omega_E(C, \mathcal{TS}) \text{ for } C \in \mathbb{I}_{\rightarrow} \\ -\Omega_E(C, \mathcal{TS}) \text{ for } C \in \mathbb{I}_{\leftarrow} \end{cases}$$

For a view $V = (L, X, E)$ and a traffic snapshot $\mathcal{TS} = (res, clm, pos, spd, acc)$ we introduce the following abbreviations:

$$res_V : \mathbb{I} \to \mathcal{P}(L) \text{ with } res_V(C) = res(C) \cap L \tag{7}$$

$$clm_V : \mathbb{I} \to \mathcal{P}(L) \text{ with } clm_V(C) = clm(C) \cap L \tag{8}$$

$$len_V : \mathbb{I} \to \mathcal{P}(X) \text{ with } len_V(C) = \begin{array}{l}[\min(pos(C), pos(C) + \Omega_E^{new}(C, \mathcal{TS})), \\ \max(pos(C), pos(C) + \Omega_E^{new}(C, \mathcal{TS}))] \cap X\end{array} \tag{9}$$

The functions (7) and (8) are restrictions of their counterparts in $\mathcal{TS}$ to the sets of lanes considered in this view. The function (9) gives us the part of the road that car E perceives as occupied by a car C, cut at the edges of the view's extension. We changed the corresponding definitions of [1]. The new definition of len_V is due to the opposing traffic. The changes of res_V and clm_V correct a technical mistake in the original paper. We note that the results from [1] remain valid even under the modified model given above.

2.3 Multi-Lane Spatial Logic with Length Measurement

We define the *multi-lane spatial logic* MLSL extended by length measurement for road segments. We start from two kinds of variables. The set of *car variables* ranging over car identificators is denoted by CVar, with typical elements c, d. To refer to the car owning the current view, we use a special variable ego $\in$ CVar. The set of *real variables* ranging over real numbers is RVar, with CVar$\cap$RVar $= \emptyset$ and typical elements x, y. The set of all *variables* is Var = CVar $\cup$ RVar, with typical element u, v. For the length measurement we need real-valued terms.

Definition 5 (Terms). *Real-valued* terms θ *of MLSL are given by the syntax*

$$\theta ::= r \mid x \mid f(c_1, \ldots, c_n) \mid g(\theta_1, \ldots, \theta_n),$$

where $r \in \mathbb{R}$, $x \in$ RVar, $c_1, \ldots, c_n \in$ CVar, *and* f, g *are* n*-ary function symbols with* $\mathbb{R}$ *as result type. We denote the set of all terms with* Θ.

Formulae of MLSL are built up from six atoms, boolean connectors and first-order quantification. Furthermore, we use two *chop* operations. The first chop is denoted by $\frown$ like for interval logics, while the second chop operation is given only by the vertical arrangement of formulae. Intuitively, a formula $\phi_1 \frown \phi_2$ is satisfied by a view V with the extension $[r, t]$, if V can be divided at a point s into two subviews V_1 and V_2, where V_1 has the extension $[r, s]$ and satisfies ϕ_1 and V_2 has the extension $[s, t]$ and satisfies ϕ_2, respectively. A formula $\begin{smallmatrix}\phi_2 \\ \phi_1\end{smallmatrix}$ is satisfied by V with the lanes l to n, if V can be split along a lane m into two subviews, V_1 with the lanes l to m and V_2 with the lanes $m + 1$ to n, where V_i satisfies ϕ_i for $i = 1, 2$.

Definition 6 (Syntax). *Formulae ϕ of MLSL with length measurement are given by the syntax*

$$\phi ::= \mathit{true} \mid u = v \mid \ell = \theta \mid \mathit{free} \mid \mathit{re}(\gamma) \mid \mathit{cl}(\gamma)$$
$$\mid \phi_1 \wedge \phi_2 \mid \neg\phi_1 \mid \exists v\colon \phi_1 \mid \phi_1 \frown \phi_2 \mid \begin{matrix}\phi_2\\ \phi_1\end{matrix}$$

where γ is a variable or a car identifier, u and v are variables, and θ is a term. We denote the set of all MLSL formulae by Φ.

In a length measurement $\ell = \theta$ the letter ℓ stands for *length* as in [3], We use ϕ^θ as an abbreviation for the formula $\phi \wedge \ell = \theta$.

Definition 7 (Valuation and Modification). *A* valuation *is a function* $\nu\colon \mathrm{Var} \to \mathbb{I} \cup \mathbb{R}$ *that* respect types *(maps car variables to car identifiers and real variables to real values). Inductively we lift ν to a function $val_{\mathcal{TS},\nu}\colon \Theta \to \mathbb{I} \cup \mathbb{R}$:*

$$val_{\mathcal{TS},\nu}(\theta) = \begin{cases} r & \text{if } \theta = r \in \mathbb{R} \\ \nu(x) & \text{if } \theta = x \in \mathbb{R} \\ \hat{f}_{\mathcal{TS}}(\nu(c_1), \ldots, \nu(c_n)) & \text{if } \theta = f(c_1, \ldots, c_n) \\ \hat{g}(val_{\mathcal{TS},\nu}(\theta_1), \ldots, val_{\mathcal{TS},\nu}(\theta_n)) & \text{if } \theta = g(\theta_1, \ldots, \theta_n), \end{cases}$$

where $\hat{f}_{\mathcal{TS}}, \hat{g}$ are the interpretations of f, g, with subscript $\mathcal{TS}$ indicating a possible dependency on a traffic snapshot $\mathcal{TS}$. For a valuation ν we use the overriding notation $\nu \oplus \{v \mapsto \alpha\}$ to denote the modified valuation*, where the value of v is modified to α. We assume that this modification respects types.*

We define partitioning of discrete intervals. We need this notion to have a clearly defined chopping operation, even on the empty set of lanes.

Definition 8 (Chopping discrete intervals). *Let I be a discrete interval, i.e., $I = [l, n]$ for some $l, n \in \mathbb{L}$ or $I = \emptyset$. Then $I = I_1 \ominus I_2$ if and only if $I_1 \cup I_2 = I$, $I_1 \cap I_2 = \emptyset$, and both I_1 and I_2 are discrete intervals such that $\max(I_1) + 1 = \min(I_2)$, or $I_1 = \emptyset$ or $I_2 = \emptyset$ holds.*

Since the semantics of formulae depends on both views and valuations, we will only consider valuations ν which are *consistent* with the current view $V = (L, X, E)$, which means that we require $\nu(\mathrm{ego}) = E$. In the following definition, we require that the spatial atoms hold only on a view with exactly one lane and an extension greater than zero. In the semantics of *free*, we abstract from cars visible only at the endpoints of the view.

Definition 9 (Semantics). *The* satisfaction $\models$ *of formulae with respect to a traffic snapshot $\mathcal{TS}$, a view $V = (L, X, E)$ with $L = [l, n]$ and $X = [r, t]$, and a valuation ν consistent with V is defined inductively as follows:*

$$\mathcal{TS}, V, \nu \models \mathit{true} \qquad \text{for all } \mathcal{TS}, V, \nu$$

$$
\begin{array}{lcl}
\mathcal{TS}, V, \nu \models u = v & \Leftrightarrow & \nu(u) = \nu(v) \\
\mathcal{TS}, V, \nu \models \ell = \theta & \Leftrightarrow & |X| = val_{\mathcal{TS},\nu}(\theta) \\
\mathcal{TS}, V, \nu \models \mathit{free} & \Leftrightarrow & |L| = 1 \text{ and } |X| > 0 \text{ and} \\
 & & \forall\, C \in \mathbb{I}\colon len_V(C) \cap (r,t) = \emptyset \\
\mathcal{TS}, V, \nu \models re(\gamma) & \Leftrightarrow & |L| = 1 \text{ and } |X| > 0 \text{ and } \nu(\gamma) \in \mathbb{I} \text{ and} \\
 & & res_V(\nu(\gamma)) = L \text{ and } X = len_V(\nu(\gamma)) \\
\mathcal{TS}, V, \nu \models cl(\gamma) & \Leftrightarrow & |L| = 1 \text{ and } |X| > 0 \text{ and } \nu(\gamma) \in \mathbb{I} \text{ and} \\
 & & clm_V(\nu(\gamma)) = L \text{ and } X = len_V(\nu(\gamma)) \\
\mathcal{TS}, V, \nu \models \phi_1 \wedge \phi_2 & \Leftrightarrow & \mathcal{TS}, V, \nu \models \phi_1 \text{ and } \mathcal{TS}, V, \nu \models \phi_2 \\
\mathcal{TS}, V, \nu \models \neg\phi & \Leftrightarrow & \text{not } \mathcal{TS}, V, \nu \models \phi \\
\mathcal{TS}, V, \nu \models \exists v\colon \phi & \Leftrightarrow & \exists\, \alpha \in \mathbb{I} \cup \mathbb{R}\colon \mathcal{TS}, V, \nu \oplus \{v \mapsto \alpha\} \models \phi \\
\mathcal{TS}, V, \nu \models \phi_1 \frown \phi_2 & \Leftrightarrow & \exists\, s\colon r \le s \le t \text{ and} \\
 & & \mathcal{TS}, V_{[r,s]}, \nu \models \phi_1 \text{ and } \mathcal{TS}, V_{[s,t]}, \nu \models \phi_2 \\
\mathcal{TS}, V, \nu \models \begin{smallmatrix}\phi_2\\ \phi_1\end{smallmatrix} & \Leftrightarrow & \exists\, L_1, L_2\colon L = L_1 \ominus L_2 \text{ and} \\
 & & \mathcal{TS}, V^{L_1}, \nu \models \phi_1 \text{ and } \mathcal{TS}, V^{L_2}, \nu \models \phi_2
\end{array}
$$

We write $\mathcal{TS} \models \phi$ if $\mathcal{TS}, V, \nu \models \phi$ for all views V and consistent valuations ν.

We remark that both chop modalities are associative. For the definition of the controller we employ some abbreviations. In addition to the usual definitions of $\vee, \rightarrow, \leftrightarrow$ and $\forall$, we use a single variable or car identifier γ as an abbreviation for $re(\gamma) \vee cl(\gamma)$. Furthermore, we use the notation $\langle\phi\rangle$ for the two-dimensional modality *somewhere* ϕ, defined in terms of both chop operations:

$$\langle\phi\rangle \equiv \mathit{true} \frown \begin{pmatrix} \mathit{true} \\ \phi \\ \mathit{true} \end{pmatrix} \frown \mathit{true}.$$

In the following, the main application of the somewhere modality is to abstract the exact positions on the road from formulae, e.g., to identify overlaps of claims and safety envelopes. If a view V satisfies the formula $\exists c\colon \langle cl(\text{ego}) \wedge re(c)\rangle$, then there is a part on some lane in V occupied by both the claim of the car under consideration and the safety envelope of some car c.

3 Controllers for Overtaking with Perfect Knowledge

We now present a protocol realised by several controllers for the overtaking manoeuvre. The complexity of the controllers depends on the knowledge available for each car about the surrounding cars. For simplicity, we assume here *perfect knowledge*, i.e., all cars know the extension of all safety envelopes within their view. This assumption can be formalised by instantiating the sensor function Ω_E (see Sec. 2.2) as $\Omega_E(I, \mathcal{TS}) = se(I, \mathcal{TS})$, where se is a function returning the

safety envelope, an overapproximation of the braking distance. This implies that the car E can perceive a car C entering its view as soon as part of the safety envelope enters E's view. Clearly, this is an idealization that in reality would require very powerful sensors for each car. For the setting of motorways we have considered also more realistic sensor functions in [1], but for country roads we leave this for future work.

We extend the lane-change automaton LCP of [1] to take the new situation at the border lanes into account. On the rest of the country road the automaton works as before, with slight modifications described in Sec. 3.3. We assume that ego $\in \mathbb{I}_{\rightarrow}$, i.e., ego is driving in direction of increasing values of $\mathbb{R}$. For the border lanes, we need to employ communication with the surrounding cars to guarantee safety, even though we assume perfect knowledge. Otherwise cars from the lanes next to the border b, i.e., from $b-1$ and $b+2$, could move into the free space either in front of the car C, which E wants to overtake, or into the lane that E needs for overtaking C. For example, in Fig. 1 car D could move into lane 1 and then block the space E needs to move back. Similarly, the car B would be allowed to move into lane 2 and block the space E needs to pass C. To prohibit this behaviour, we use a *helper automaton* as described in Sec. 3.2.

To simplify the safety proof in Sec. 4 we structure the overtaking manoeuvre into the following three phases, also shown in Fig. 2.

1. Change lanes into opposing traffic.
2. Pass the car driving in front.
3. Change lanes back into the original driving lane.

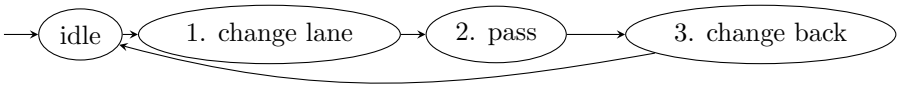

Fig. 2. Protocol for overtaking

For phase 1 and 3 we will present controllers as timed automata [12] with data variables ranging over $\mathbb{I}$ and $\mathbb{L}$. We allow for MLSL formulae as transition guards and invariants and evaluate them over the standard view V_s (see Sec. 2.2) of the car the controller is implemented in. Furthermore we use the labels of the transition system of $\mathcal{TS}$ as actions. Finally, communication is modelled via *broadcast messages*, similar to UPPAAL [13]. The notation we use for communication is inspired by CSP [14]. For phase 2 we will omit the presentation of an automaton due to its simplicity but give a detailed description.

Our controllers rely on a spatial decomposition of the overtaking manoeuvre. To formalise this, we assume function symbols interpreted as functions yielding certain distances for cars travelling at speeds determined by the current traffic snapshot $\mathcal{TS}$ (cf. Def. 7):

- $d(t)$ yields the maximal *distance* a car can travel within a given time t;
- $d_{lc}(E)$ yields the distance needed for a *lane change* of a car E driving at a speed determined by $\mathcal{TS}$;
- $d_{pass}(E, C)$ yields the distance needed for a car E to *pass* a car C in front, given their speeds determined by $\mathcal{TS}$;
- $d_{lcb}(E, C)$ yields the distance needed for a *lane change back* of a car E after passing car C with speeds determined by $\mathcal{TS}$;
- $d_{max} = d(t_{max})$, where the time t_{max} is large enough for a car to safely change lanes twice and pass another car.

3.1 Overtaking protocol

Overall we want to maintain the property that all reservations are disjoint. We formalise the unwanted situations in MLSL by the formula *collision check*:

$$cc \equiv \exists c : c \neq \text{ego} \wedge \langle re(\text{ego}) \wedge re(c) \rangle \quad .$$

Now we will present controllers for the three phases of the manoeuvre that will maintain this property in a setting with opposing traffic. To construct the complete controller for the overtaking protocol we fuse the locations without any outgoing arcs, named q_4 in Fig. 4 and Fig. 5, with the initial locations of the next phase of the protocol.

1. Changing Lanes into Opposing Traffic Intuitively, we can describe this phase of the manoeuvre as follows. The car ego drives on lane n (its *original lane*) and wants to change into the *target lane* $n + 1$ next to n. Then ego first sets a claim on the target lane and checks the following three properties:

1. Does the claim intersect with the reservation or claim of any other car?
2. Is there enough space on the original lane to change back during the completion of the manoeuvre?
3. Is there enough space for the overtaking manoeuvre on the target lane?

Should one of these conditions be violated, withdraws ego its claim and therefore aborts the manoeuvre. Otherwise, ego proceeds by sending a message that it starts an overtaking manoeuvre, turning the claim into a reservation and starting moving over to the target lane. Furthermore, this message obliges the overtaken car to keep its velocity constant.

We formalise the first property, the *potential collision*, i.e., car ego's claim intersects with another car's claim or reservation, in the following MLSL formula:

$$pc \equiv \exists c : c \neq \text{ego} \wedge \langle cl(\text{ego}) \wedge c \rangle \quad .$$

For checking that there is enough space on the target lane and on the original driving lane, we need two functions k_1 and k_2 (see Fig. 3):

$$k_1 : \mathbb{I} \times \mathbb{I} \to \mathbb{R}_+ \text{ with } (E, C) \mapsto d_{lc}(E) + d_{pass}(E, C)$$

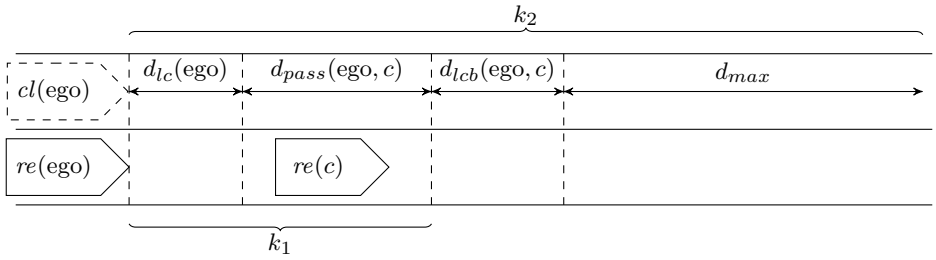

Fig. 3. Lengths referenced to during overtaking manoeuvre

is the distance needed by car E for a lane change with its current speed and for passing the car C, during which E may accelerate to a higher speed.

$$k_2 : \mathbb{I} \times \mathbb{I} \to \mathbb{R}_+ \text{ with } (E, C) \mapsto k_1(E, C) + d_{lcb}(E, C) + d_{max}$$

adds the distance it takes E to change back into the original lane (with the speed assumed while passing car C in front plus the maximal distance d_{max} a car on the opposing lane can travel in the time it takes to overtake C. With these two functions we can now formulate the remaining two properties.

$$esol(c) \equiv \left\langle re(\text{ego}) \frown (free \frown re(c) \frown free)^{k_1(\text{ego},c)} \frown free^{d_{lcb}(\text{ego},c)} \right\rangle$$

states that there is *enough space on the original lane*. Since ego is currently holding a claim, it can reserve only one lane, and hence *esol* refers to ego's original lane. In front of the reservation there has to be enough space such that ego can safely pass the car c and then fit in front of c (cf. Fig. 3).

$$estl(c) \equiv \left\langle cl(\text{ego}) \frown free^{k_2(\text{ego},c)} \right\rangle$$

states that there is *enough space on the target lane*. We fix the target lane by checking for ego's claim. In front of the claim there has to be enough space to complete the overtaking manoeuvre plus the maximal distance travelled by an opposing car during the time of the manoeuvre (cf. Fig. 3).

Figure 4 shows the controller part for this phase of the manoeuvre. We start in q_0 which we assume to be safe, i.e., satisfying $\neg cc$. We hold ego's original driving lane in the variable n and make use of the additional variable l saving the target lane. We use the constant t_{lc} for the time that is needed to change lanes. Note that due to the guard $n = b$ this controller only takes effect if ego is driving on the border and ego's target lane is the lane with opposing traffic.

2. Passing the car ahead The controller for the car ego to pass by car c of the manoeuvre is rather simple. The car ego accelerates to a higher speed, remains at that speed until it has passed car c, and initiates the next phase of

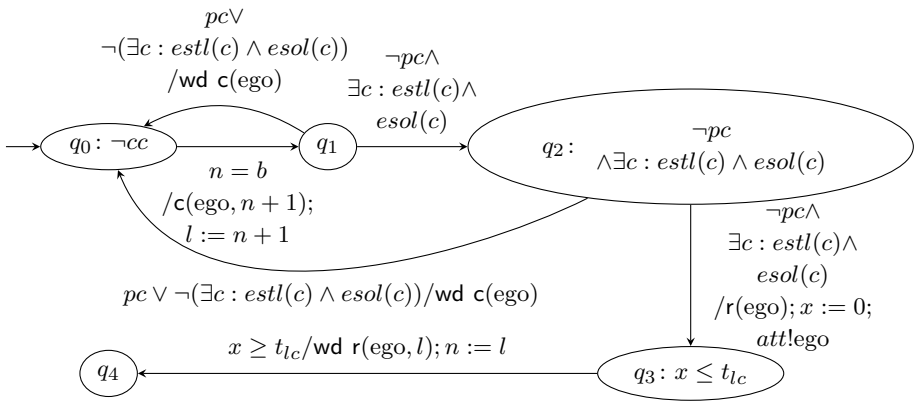

Fig. 4. Controller for the lane-change manoeuvre into opposing traffic

the manoeuvre. Furthermore, whenever ego receives a request for a lane change from a car c, i.e., the message $req?c$, ego checks whether c wants to occupy space needed for the overtaking manoeuvre, i.e., whether $\langle re(\text{ego}) \frown true \frown cl(c) \rangle$ holds. If this is the case, ego denies c's request by sending the message $no!c$. We omit the presentation of the described controller.

3. Changing Lanes Back into Original Driving Lane The controller for changing the lane back into the original driving lane (see Fig. 5) is a drastically simplified version of the controller in Sec. 3.1. We already established with *esol* that there is enough space to change lanes back into the original lane. We only need to set the variable l to the original lane $n-1$, then we can claim it ($\mathsf{c}(\text{ego}, n-1)$), reserve the lane ($\mathsf{r}(\text{ego})$), and change over within t_{lc} time.

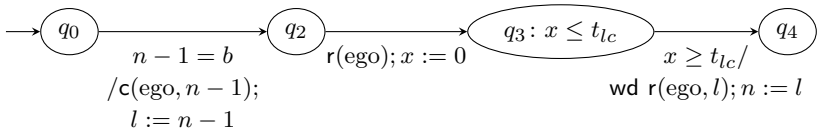

Fig. 5. Controller LCP for the lane-change manoeuvre with perfect knowledge back into orignal lane

3.2 Helper controller

The automaton in Fig. 6 is glued to the helper controller from [1], to prohibit cars from moving to space used during the overtaking manoeuvre. As soon as a car

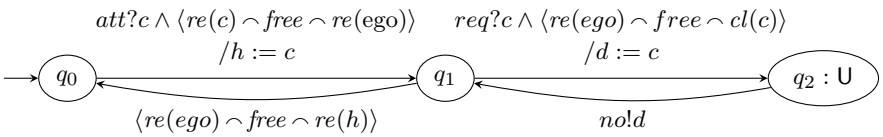

Fig. 6. Additional automaton for helper controller

starts to overtake another car C, it calls for *attention* by sending its identity on channel *att* (see Fig. 4). After receiving the message $att?c$, the car C can realise that it is being overtaken, by checking whether $\langle re(c) \frown free \frown re(\text{ego})\rangle$ holds. Then C reacts to each request $req?d$ of other cars to move into the free space in front of it with a negative reply $no!d$. The end of the overtaking manoeuvre can be perceived by C by means of the formula $\langle re(ego) \frown free \frown re(h)\rangle$. Combined with the lane change controller from [1] this maintains that no other car moves into the free space in front of the car which is being overtaken.

3.3 Changing Lanes for Non-Border Lane Manoeuvres

In Fig. 7 we present the slightly modifed controller from [1], which in cooperation with the previous controllers ensures safety of the complete manoeuvre. This controller is responsible for the lane change if the car is not moving into opposing traffic during the lane change.

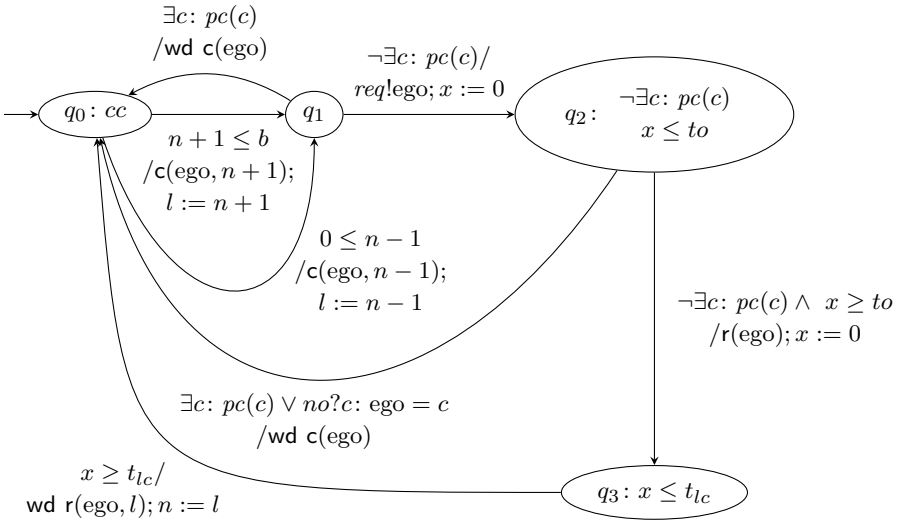

Fig. 7. Controller LCP for the lane-change manoeuvre with perfect knowledge on non-border lanes

Similar to the controller in Sec. 3.1 it maintains the current driving lane in the variable n and the target lane in the variable l. Upon attempting a lane change ego sets its claim on the target lane and brodcasts a request $req!\text{ego}$ awaiting an answer for the next to time units. If no abort message $no?c : \text{ego} = c$ arrives within this time-out, ego turns its claim into a reservation and starts to move over. After t_{lc} the manoeuvre is completed and ego withdraws the original reservation. We assume the time-out to to be large enough such that a car which previously moved into opposing traffic can answer within this time bound.

Note that ego receives a *no* message in the following two situations:

1. The car C behind ego is currently being overtaken.
2. ego wants to change into a lane a car D is currently using to overtake.

In the first case C sends a *no* to ensure that the space ego perceives *free* is maintained *free* for the overtaking car behind C. In the second case D sends a *no* message to ego to maintain that the space it currently uses to complete the overtake manoeuvre remains *free*.

4 Safety of Overtaking Manoeuvre

In this section, we will give an informal safety proof of the controllers shown in Sec. 3. Since the reservations of the model are overapproximations of the braking distance of cars together with their physical size, we understand safety as the non-overlapping of any reservations. Claims however may overlap, since they represent only intentions of future car positions.

We make the following assumptions on the country road traffic. The assumptions are slight extensions to the assumptions for our proof of motorway situations [1], to take the opposing traffic into account.

Assumption A1. There is an *initial safe* traffic snapshot $\mathcal{TS}_0$.

Assumption A2. Every car C is equipped with a *distance controller* that keeps the safety property invariant with respect to all cars driving in the same direction as C under time and acceleration transitions, i.e., for every transition $\mathcal{TS} \xrightarrow{t} \mathcal{TS}'$ and $\mathcal{TS} \xrightarrow{\mathsf{acc}(C,a)} \mathcal{TS}'$ if $\mathcal{TS}$ is safe also $\mathcal{TS}'$ is safe.

Informally, this means that the distance controller admits a positive acceleration of C only if the space ahead permits this. Also, if the car ahead is slowing down, the distance controller has to initiate braking (with negative acceleration) of C to reduce the extension of its reservation (the safety envelope).

Assumption A3. Every car is equipped with a controller implementing the protocol in Sec. 3.1.

Finally, we assume the horizon h to be of appropriate size, as stated in Sec. 2.

Assumption A4. The horizon h is $h = se_{max} + 2 \cdot d_{max}$. This length stems from the distance we need for the overtaking manoeuvre to take place, as shown in the controllers, as well as the maximal length of the safety envelope se_{max}.

Theorem 1 (Safety). *Under the assumptions A1 to A4, the protocol specifying the overtaking procedure of Section 3 is safe.*

Proof. We sketch the safety proof by analysing the three phases of the protocol. Let $V = (\mathbb{L}, X, E)$ be the view of the car $E \in \mathbb{I}_{\rightarrow}$ performing the overtaking of a car C and $\mathcal{TS}_0$ the initial safe snapshot (A1).

The first phase is essentially the lane change procedure of our previous work [1], under different assumptions. However, the new assumptions are stronger in the following sense. First, we only allow for the lane change in one direction, i.e., if E is driving on lane b, it may only change to lane $b+1$. Furthermore, in the proof for motorway situations, the horizon was at least of the length of the safety envelope. Since by A4 the horizon is still large enough, the lane change manoeuvre can take place in a safe manner. In this phase, the controller has to ensure that in the third phase, there is still enough space in front of C to change back to lane b. Therefore, at the instant, when E extends its reservation, it broadcasts its ID on the channel *att*. Now C is the only car, where the guard in the helper automaton (Fig. 6) is satisfied. Any car attempting to change to b in front of C, i.e., where C would serve as a helper car, will receive the denial to change lanes, until E has changed back to b, as stated in the guard on the transition back to the initial location of the helper automaton. Hence, there will be still free space for E on lane b in the third phase of the protocol. Similarly, a car $D \in \mathbb{I}_{\leftarrow}$ trying to perform a lane change to lane $b+1$ will receive a denial directly from ego.

Now assume that phase one was successful, i.e., E started its lane change at snapshot $\mathcal{TS}_0$ and successfully changed to lane $b+1$ on a subsequent snapshot $\mathcal{TS}_1$. Then the free space in front of E on $b+1$ is still at least $d_{pass}(E,C) + d_{lcb}(E,C) + (d_{max} - d(t_{lc}))$. The car E needs $d_{pass}(E,C)$ space during this phase. Due to the definition of d_{max} an opposing car may cover at most a distance of $d_{max} - 2 \cdot d(t_{lc})$ in the passing phase. Hence in the worst case after this phase, there is still at least $d_{lcb}(E,C) + (d_{max} - d(t_{lc}) - (d_{max} - 2 \cdot d(t_{lc})) = d_{lcb}(E,C) + d(t_{lc})$ free space left. Similarly to the description of the first phase, E denies all cars wanting to change to its current lane the permission to do so.

For the last phase of the protocol, observe that $d_{lcb}(E,C)$ is the space needed by E to change back to lane b, while $d(t_{lc})$ is the largest distance an opposing car may cover when driving at maximal velocity. Hence after the final lane change, i.e., before E removes its reservation from lane $b+1$, in the worst case the start of the envelope of the opposing traffic is directly in front of E's reservation, but they are not yet overlapping. Since the removal of reservations is instantaneous, there is no point in time where the these reservations can overlap.

Now assume that a car $D \in \mathbb{I}_{\leftarrow}$ on lane $b+1$ with a view $V' = (\mathbb{L}, X', D)$ outside the horizon of E is also planning an overtaking manoeuvre of a car A. Observe that this also implies that E is outside the horizon of D, i.e., $\max(len_V(E)) < pos(D) - h$. An unsafe situation may only occur, when the

spaces used to change back on the original lanes overlap, i.e., when

$$\begin{array}{rcl} & & \max(len_V(E)) + d_{lc}(E) + d_{pass}(E,C) + d_{lcb}(E,C) \\ & > & \min(len_{V'}(D)) - (d_{lc}(D) + d_{pass}(D,A) + d_{lcb}(D,A)) \\ \iff & & \max(len_V(E)) + d_{lc}(E) + d_{pass}(E,C) + d_{lcb}(E,C) - d_{max} \\ & > & \min(len_{V'}(D)) - (d_{lc}(D) + d_{pass}(D,A) + d_{lcb}(D,A)) - d_{max} \end{array} \quad (10)$$

Now by definition $d_{max} \geq d_{lc}(E) + d_{pass}(E,C) + d_{lcb}(E,C)$. Hence we get

$$\max(len_V(E)) \geq \max(len_V(E)) + \ d_{lc}(E) + d_{pass}(E,C) + d_{lcb}(E,C) - d_{max} \quad (11)$$

By definition, we have $|len_{V'}(D)| \leq se_{max}$, and hence $\min(len_{V'}(D)) = pos(D) - |len_{V'}(D)| \geq pos(D) - se_{max}$. Since also $d_{lc}(D) + d_{pass}(D,A) + d_{lcb}(D,A) \leq d_{max}$, we have that

$$\begin{array}{l} \min(len_{V'}(D)) - (d_{lc}(D) + d_{pass}(D,A) + d_{lcb}(D,A)) - d_{max} \\ = pos(D) - |len_{V'}(D)| - (d_{lc}(D) + d_{pass}(D,A) + d_{lcb}(D,A)) - d_{max} \\ \geq pos(D) - se_{max} - (d_{lc}(D) + d_{pass}(D,A) + d_{lcb}(D,A)) - d_{max}. \\ \geq pos(D) - (se_{max} + 2 \cdot d_{max}) \end{array} \quad (12)$$

By putting (10), (11) and (12) together and using $h = se_{max} + 2 \cdot d_{max}$, we get $\max(len_V(E)) > pos(D) - h$, which means that E is visible within the standard view of D, and hence contradicts our assumption. □

5 Conclusion

The novelty in our approach is the identification of a level of abstraction that enables a purely spatial reasoning on safety. In [1] this was demonstrated for motorways. In this paper we showed that with small extensions to the previously developed setting, in particular explicit length measurement, we can also deal with the two-way traffic of country roads. We proved safety for arbitrarily many cars on country roads locally, by considering a limited amount of space.

In our future work, we would like to study variations of the assumptions made in our safety proofs. First of all, we will lift the assumption of perfect knowledge. In [1] we have already done this for the case of one-way motorway traffic. There the more realistic sensor function assumes that each car knows only the physical size of all other cars in its view; the safety envelope it knows only of itself. Further, we intend to study the scenarios of urban traffic as in [6]. Also, we plan to link our work to hybrid systems: a refinement of the spatial reasoning in this paper to the car dynamics is of interest. There we could benefit from the work of [7,8,9].

So far, our proof of the Safety Theorem 1 is by hand. We show that the transitions between traffic snapshots obeying our controllers preserve a suitable safety invariant. It is our aim to ultimately provide automatic support for such

proofs. In [15] steps in this direction are presented. There a proof system for an extended version of the logic called EMLSL is introduced. EMLSL embraces temporal operators, so reasoning about the transitions can be conducted within this logic. However, mechanic support for the proofs remains a topic of future research.

References

1. Hilscher, M., Linker, S., Olderog, E.R., Ravn, A.: An abstract model for proving safety of multi-lane traffic manoeuvres. In Qin, S., Qiu, Z., eds.: Intern. Conf. on Formal Engineering Methods. Volume 6991 of LNCS., Springer (2011) 404–419
2. Moszkowski, B.: A temporal logic for multilevel reasoning about hardware. Computer **18** (1985) 10–19
3. Zhou, C., Hoare, C., Ravn, A.: A calculus of durations. Information Processing Letters **40** (1991) 269–276
4. Schäfer, A.: Axiomatisation and decidability of multi-dimensional duration calculus. Information and Computation **205** (2007) 25–64
5. Lygeros, J., Godbole, D.N., Sastry, S.S.: Verified hybrid controllers for automated vehicles. IEEE Transactions on Automatic Control **43** (1998) 522–539
6. Werling, M., Gindele, T., Jagszent, D., Gröll, L.: A robust algorithm for handling traffic in urban scenarios. In: Proc. IEEE Intelligent Vehicles Symposium, Eindhoven, The Netherlands (2008) 168–173
7. Moor, T., Raisch, J., O'Young, S.: Discrete supervisory control of hybrid systems based on l-complete approximations. Discrete Event Dynamic Systems **12** (2002) 83–107
8. Habets, L.C.G.J.M., Collins, P., van Schuppen, J.: Reachability and control synthesis for piecewise-affine hybrid systems on simplices. IEEE Transactions on Automatic Control **51** (2006) 938–948
9. Damm, W., Hungar, H., Olderog, E.R.: Verification of cooperating traffic agents. International Journal of Control **79** (2006) 395–421
10. He, J., Hoare, C.A.R., Fränzle, M., Müller-Olm, M., Olderog, E.R., Schenke, M., Hansen, M.R., Ravn, A.P., Rischel, H.: Provably correct systems. In Langmaack, H., de Roever, W.P., Vytopil, J., eds.: FTRTFT. Volume 863 of LNCS., Springer (1994) 288–335
11. Woodcock, J., Davies, J.: Using Z – Specification, Refinement, and Proof. Prentice Hall (1996)
12. Alur, R., Dill, D.L.: A theory of timed automata. TCS **126** (1994) 183 – 235
13. Behrmann, G., David, A., Larsen, K.G.: A tutorial on UPPAAL. In Bernardo, M., Corradini, F., eds.: Formal Methods for the Design of Real-Time Systems, Springer–Verlag (2004) 200–236
14. Hoare, C.A.R.: Communicating sequential processes. CACM **21** (1978) 666–677
15. Linker, S., Hilscher, M.: Proof theory of a multi-lane spatial logic. In Liu, Z., Woodcock, J., Zhu, H., eds.: 10th Intern. Col. on Theoretical Aspects of Computing (ICTAC). (2013) to appear.