Linking Spatial and Dynamic Models for Traffic Maneuvers

Ernst-Rüdiger Olderog[1], Anders P. Ravn[2], and Rafael Wisniewski[3]

***Abstract*— For traffic maneuvers of multiple vehicles on highways we build an abstract spatial and a concrete dynamic model. In the spatial model we show the safety (collision freedom) of lane-change maneuvers. By linking the spatial and dynamic model via suitable refinements of the spatial atoms to distance measures, the safety carries over to the concrete model.**

I. Introduction

This paper develops controls for vehicles maneuvers in traffic, which are by design safe. For conciseness of presentation, we concentrate on distance control and lane change in traffic on highways. To this end, we build an abstract spatial and a concrete dynamic model for these traffic maneuvers such that the concrete models refines the abstract one through symbolic linking relations. Similar links can also be built for other traffic maneuvers with an abstract spatial view linked to a concrete dynamic model and controller for velocity and distance control as well as for lateral motion (to cope with lane change).

An inspiration for the symbolic linking is the data refinement relations explored in program verification [4]. However, in the reactive setting, linking predicates as in the approach of UTP (Unifying Theories of Programming) [8] are more suitable. In summary, the approach is as follows. For the abstract, discrete model the steps are:

1) Qualitative model of the context with symbolic representation of object states.
2) Rules for interaction formulated as finite state machines operating on the symbolic states. If the state machines use communication protocols, timeout transitions may compensate for lost messages.
3) Formulation of safety properties of the symbolic state.
4) Verification of the properties.

These steps are illustrated on the case of vehicle maneuvers in Sections II and III.

When the verification is successful, it is time to consider concrete models. It has the following steps:

5) Identification of the concrete models for objects including available or at least plausible sensors and actuators.
6) Linkage through refinement relations of the symbolic state variables of the abstract model to concrete observers that are implemented using available sensors and concrete models of the individual objects. Also linking of symbolic actions to values of set points for the controls.
7) Design and validation of the controllers and observers.

These steps are illustrated on the case in Sections IV to VI.

Note that the two models may be developed concurrently. When this takes place, it is important to keep the linkage stable when doing separate iterations.

A pragmatic consideration when designing the linking in the concrete case has been to design the system so a smart car can navigate among ordinary cars. It is impractical to require all cars to be smart (automated) and able to communicate with other cars. This has implications on the sensors and actuators, see Fig. 2, as well as on the design of symbolic guards and actions.

Section VII presents a short conclusion including comments on related work.

II. Symbolic Model

In this section, we summarize and adapt the model of [7]. In the model, a multi-lane highway has an infinite extension with positions represented by real numbers in $\mathbb{R}$ and with lanes represented by natural numbers. We assume that all traffic proceeds in one direction, with increasing position values, in pictures shown from left to right. The highway is populated by uniquely identified cars, denoted by capital letters $A, B, \ldots$. Let $\mathbb{I}$ denote the set of car identifiers and $\mathbb{L} = \{0, \ldots, N\}$ denote the set of lanes.

At each moment of time, the traffic on the highway is given by a *traffic snapshot*. It records for each car the current position *pos* (at the rear end of the car), speed *spd*, and on which lanes the car *reserves* or *claims* space. The idea is that a reserved space is owned by a unique car. Thus for safety, we have to show that reserved spaces of different cars are mutually exclusive. In contrast, a claimed space is used when preparing a lane change and may overlap with claimed or reserved spaces of other cars. However, then the lane change must not take place. The extension of reserved and claimed spaces is given by the *safety distance*, which is the length of the car plus a conservative approximation of the braking distance of the car.

Definition 1: A *traffic snapshot* $\mathscr{T}$ comprises functions pos, spd, res, clm of the following type:

- $pos : \mathbb{I} \to \mathbb{R}$ where $pos(C)$ is the position of car C along the lanes,
- $spd : \mathbb{I} \to \mathbb{R}$ where $spd(C)$ is the current speed of car C,
- $res : \mathbb{I} \to \mathscr{P}(\mathbb{L})$ where $res(C)$ is the set of lanes C reserves,
- $clm : \mathbb{I} \to \mathscr{P}(\mathbb{L})$ where $clm(C)$ is the set of lanes C claims.

We denote the set of all traffic snapshots by $\mathbb{TS}$.

[1]Department of Computing Science, University of Oldenburg, Germany olderog@informatik.uni-oldenburg.de

[2]Department of Computer Science, Aalborg University, Denmark apr@cs.aau.dk

[3]Department of Electronic Systems, Automation and Control, Aalborg University,Aalborg, Denmark raf@es.aau.dk

The space occupied on the reserved and claimed lanes is given by the symbolic function *se*, the *safety envelope*. For a given traffic snapshot $\mathscr{T}$, the safety envelope of car C, $se_{\mathscr{T}}(C)$, is the interval

$$se_{\mathscr{T}}(C) = [pos(C), pos(C)+d]$$

starting at the current position $pos(C)$ of the car and of some symbolic length $d > 0$, which is intended to be the sum of the size of the car, and its current braking distance, which is dependent on its current speed $spd(C)$. The exact value of d is not known in the symbolic model, but will be determined in the concrete dynamic model.

A. View

The safety proof considers a finite segment of a traffic snapshot $\mathscr{T}$ called a *view*, the intuition being that the safety of maneuvers can be shown using local information only.

Definition 2: A *view* $V = (L, X, E)$ consists of the following: $L = [l,n] \subseteq \mathbb{L}$ is an interval of lanes visible in the view, $X = [r,t] \subseteq \mathbb{R}$ is the extension visible in the view, and $E \in \mathbb{I}$ is the identifier of the car under consideration.

A *subview* of V is obtained by restricting the lanes and extension observed. 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.

B. Spatial Logic

Properties of traffic snapshots within a given view are specified in an intuitive and yet precise way, with a two-dimensional spatial interval logic, called MLSL (Multi-Lane Spatial Logic) [7]. Formulae of this logic express the spatial status of neighboring lanes. For a lane, the spatial status describes whether parts of it are reserved or claimed by a car or completely free. To this end, MLSL has atoms $re(\gamma)$, $cl(\gamma)$, and *free*, and two chop operators: the horizontal chop $\phi_1 \frown \phi_2$ expresses that ϕ_1 holds left and ϕ_2 right, and the vertical chop $\begin{matrix}\phi_2\\ \phi_1\end{matrix}$ that ϕ_1 holds below and ϕ_2 above.

Variables ranging over car identifiers are denoted by small letters c, d, u and v. The car owning the current view, use the special variable *ego*. The set of all variables is Var.

Definition 3 (Syntax): The syntax of the *multi-lane spatial logic MLSL* is given by the following formulae:

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

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

The logic is given a simple semantics that defines the shape of traffic snapshots.

Definition 4 (Semantics): Let u and w be variables and γ be a variable or a car identifier. The *satisfaction* $\models$ of formulae is defined inductively with respect to a triple $\mathscr{M} = (\mathscr{T}, V, \nu)$ comprising a traffic snapshot $\mathscr{T}$, a view $V = (L, X, E)$ with $L = [l,n]$ and $X = [r,t]$, and a valuation $\nu : \mathrm{Var} \to \mathbb{I}$ *consistent* with V, i.e., with $\nu(ego) = E$:

$$\begin{aligned}
&\mathscr{M} \models true && \text{for all } \mathscr{M}\\
&\mathscr{M} \models u = w && \Leftrightarrow \quad \nu(u) = \nu(w)\\
&\mathscr{M} \models free && \Leftrightarrow \quad |L| = 1 \text{ and } |X| > 0 \text{ and}\\
&&& \quad \forall C \in \mathbb{I}\colon (L \subseteq res(C) \cup clm(C) \Rightarrow se_{\mathscr{T}}(C) \cap (r,t) = \emptyset)\\
&\mathscr{M} \models re(\gamma) && \Leftrightarrow \quad |L| = 1 \text{ and } |X| > 0 \text{ and}\\
&&& \quad L \subseteq res(\nu(\gamma)) \text{ and } X \subseteq se_{\mathscr{T}}(\nu(\gamma))\\
&\mathscr{M} \models cl(\gamma) && \Leftrightarrow \quad |L| = 1 \text{ and } |X| > 0 \text{ and}\\
&&& \quad L \subseteq clm(\nu(\gamma)) \text{ and } X \subseteq se_{\mathscr{T}}(\nu(\gamma))
\end{aligned}$$

The formulae $\phi_1 \wedge \phi_2$, $\neg\phi$, and $\exists w\colon \phi$ have their standard semantics. Novel are the chop operators.

$$\begin{aligned}
&\mathscr{M} \models \phi_1 \frown \phi_2 && \Leftrightarrow \quad \exists s\colon r \le s \le t \text{ and}\\
&&& \quad (\mathscr{T}, V_{[r,s]}, \nu) \models \phi_1 \text{ and } (\mathscr{T}, V_{[s,t]}, \nu) \models \phi_2\\
&\mathscr{M} \models \begin{matrix}\phi_2\\ \phi_1\end{matrix} && \Leftrightarrow \quad (l \le n \text{ and } \exists m\colon l-1 \le m \le n \text{ and}\\
&&& \quad (\mathscr{T}, V^{[l,m]}, \nu) \models \phi_1 \text{ and } (\mathscr{T}, V^{[m+1,n]}, \nu) \models \phi_2) \text{ or}\\
&&& \quad (l > n \text{ and } (\mathscr{T}, V, \nu) \models \phi_1 \text{ and } (\mathscr{T}, V, \nu) \models \phi_2),
\end{aligned}$$

where intervals $[l,n]$ with $l > n$ are empty. We write $\mathscr{T} \models \phi$ if $(\mathscr{T}, V, \nu) \models \phi$ for all views V and consistent valuations ν.

The notation $\langle\phi\rangle$ is used for the two-dimensional modality *somewhere* ϕ, defined in terms of both chop operators:

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

For example, $Safe \equiv \forall c,d : c \neq d \to \neg\langle re(c) \wedge re(d)\rangle$ expresses the safety property that any two different cars have disjoint reserved spaces.

C. Transitions

A traffic snapshot gives a static picture of the highway. The following *transitions* describe the changes that may occur at a traffic snapshot $\mathscr{T} = (pos, spd, res, clm)$.

Time can pass, which results in the cars moving along the highway to the right.

$$\mathscr{T} \xrightarrow{t} \mathscr{T}' \quad \Leftrightarrow \mathscr{T}' = (pos', spd', res, clm) \wedge pos' > pos \quad (1)$$

The new position and speed of each car is given by the dynamics of the car, which is determined at the concrete level. Note that reservations, *res*, and claims, *clm*, do not change in time passing transitions.

A car may *claim* a neighbouring lane n if and only if it does not already claim another lane or is in the progress of changing the lane and therefore reserves two lanes.

$$\begin{aligned}
\mathscr{T} \xrightarrow{c(C,n)} \mathscr{T}' \quad &\Leftrightarrow \mathscr{T}' = (pos, spd, res, clm') \quad (2)\\
&\wedge |clm(C)| = 0 \wedge |res(C)| = 1\\
&\wedge \{n+1, n-1\} \cap res(C) \neq \emptyset\\
&\wedge clm' = clm \oplus \{C \mapsto \{n\}\}
\end{aligned}$$

The overriding notation, $\oplus$, means that *clm* is updated for car C to the value n, while it is unchanged for other cars.

Furthermore, a car may *withdraw* a claim (3) or *reserve* a previously claimed lane (4) or withdraw the reservation of all but one of the lanes it is moving on (5).

$$\mathscr{T} \xrightarrow{\mathsf{wd}\ \mathsf{c}(C)} \mathscr{T}' \quad \Leftrightarrow\ \mathscr{T}' = (pos, spd, res, clm') \quad (3)$$
$$\wedge\, clm' = clm \oplus \{C \mapsto \emptyset\}$$
$$\mathscr{T} \xrightarrow{\mathsf{r}(C)} \mathscr{T}' \quad \Leftrightarrow\ \mathscr{T}' = (pos, spd, res', clm') \quad (4)$$
$$\wedge\, clm' = clm \oplus \{C \mapsto \emptyset\}$$
$$\wedge\, res' = res \ \oplus\ \{C \mapsto res(C) \cup clm(C)\}$$
$$\mathscr{T} \xrightarrow{\mathsf{wd}\ \mathsf{r}(C,n)} \mathscr{T}' \quad \Leftrightarrow\ \mathscr{T}' = (pos, spd, res', clm) \quad (5)$$
$$\wedge\, res' = res \oplus \{C \mapsto \{n\}\}$$
$$\wedge\, n \in res(C) \wedge |res(C)| = 2.$$

III. Abstract Controller

In the unconstrained transition system, the reservation transition (4) may occur at any moment, which may lead to dangerous overlaps of reserved spaces. It is the task of a lane-change controller to prevent it.

A controller is specified by a *timed automata* [2] with clocks ranging over $\mathbb{R}$ and data variables ranging over $\mathbb{L}$ and $\mathbb{I}$. Its semantics is the previously given transition system, where the valuation ν is extended with clocks, data variables, and the current state q of the controller, cf. Fig. 1, i.e., $\mathscr{C} = (\mathscr{T}, V, \nu, q)$. To restrict the transitions which may occur while time is passing and in lane-change maneuvers, MLSL formulae appear in transition guards and state invariants.

A. Changing Lanes

Consider the controller in Fig. 1, where the formulas ϕ_0 and ϕ_2 are kept symbolic. The abstract lane-change controller LCP of [7] is an instantiation of this controller. LCP assumes that every car, E, knows the full extension of claims and reservations of all cars within its view. It has *perfect knowledge* of its neighbouring cars; E perceives another car C as soon as C's safety envelope enters the view of E. In the following and in Section V, we identify the car variables *ego* and c with their values, the cars E and C, respectively.

At the initial state q_0 of LCP, the car has reserved exactly one lane, which is saved in the variable n. An auxiliary variable l stores the lane the *ego* car wants to move to. Here, the invariant ϕ_0 in q_0 expresses disjointness of *ego*'s reservation with the reservations of all other cars, i.e., the following *collision check*

$$cc \equiv \exists c\colon c \neq ego \wedge \langle re(ego) \wedge re(c) \rangle$$

is negative. Thus in LCP, we instantiate $\phi_0 \equiv \neg cc$.

Suppose *ego* intends to change to a neighboring lane, then it adheres to the following protocol. First, it claims a space on the target lane adjacent to and of the same extension as the reservation on its current lane. Subsequently, it checks for a *potential collision*, i.e., whether its claim intersects with the reservation or claim of any other car. This is expressed by the MLSL formula pc:

$$pc \equiv \exists c : c \neq ego \wedge \langle cl(ego) \wedge (re(c) \vee cl(c)) \rangle .$$

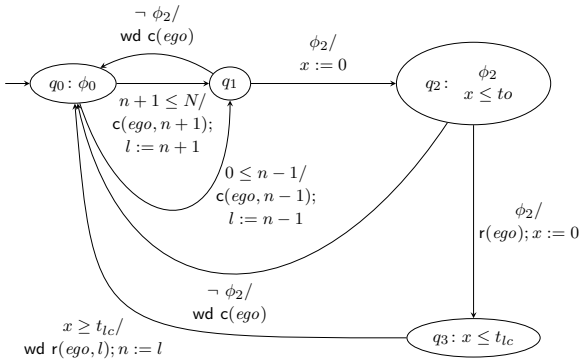


Fig. 1. Controller with symbolic formulas ϕ_0 and ϕ_2, of which the lane-change controller LCP is an instantiation with $\phi_0 \equiv \neg cc$ and $\phi_2 \equiv \neg pc$.

If pc occurs, *ego* withdraws its claim and gives up the wish to change lanes for the moment. Otherwise, without any delay, *ego* turns its claim into reservations of both neighboring lanes. During this double reservation *ego* changes lane. Once this is completed within time t_{lc}, *ego* withdraws its reservation on the original lane and continues to drive on the target lane. During this protocol, only turning the claim into a reservation may violate the safety property. Thus in LCP, we instantiate $\phi_2 \equiv \neg pc$.

B. Safety of LCP

We stipulate now that every car is equipped with the controller LCP (or that its driver manually follows its protocol). The desired safety property *Safe* is that at any moment the spaces reserved by different cars are disjoint. In MLSL,

$$Safe \equiv \forall c, d : c \neq d \Rightarrow \neg \langle re(c) \wedge re(d) \rangle ,$$

states that in any lane any two different cars have disjoint reserved spaces. The quantification over lanes arises implicitly by the negation of the somewhere modality in *Safe*. A traffic snapshot $\mathscr{T}$ is *safe* if $\mathscr{T} \models Safe$ holds. The safety property depends on the following assumptions.

- **A1** There is an *initial safe* traffic snapshot $\mathscr{T}_0$.
- **A2** Every car C keeps the safety property invariant under time transitions: for every transition $\mathscr{T} \xrightarrow{t} \mathscr{T}'$ whenever $\mathscr{T}$ is safe, also $\mathscr{T}'$ is safe.

Under these assumptions, in [7], the following result is proven by induction on the number of transitions needed to reach $\mathscr{T}$ from $\mathscr{T}_0$.

Theorem 1 (Safety of LCP): Every traffic snapshot $\mathscr{T}$ that is reachable from $\mathscr{T}_0$ by time transitions and transitions allowed by the controller LCP is safe.

IV. Concrete Model

As explained in the previous section, MLSL is used to formulate specifications for motion of vehicles on a highway. However, this two dimensional spatial logic does not address temporal properties. To this end, we abstract the time evolution. Specifically, a controller for each vehicle C will be developed that keeps the distance to the car in front at d defined in $se_{\mathscr{T}}$, or if there is no car in the range of a

distance sensor the controller maintains a reference velocity v_{ref}. Recall that d is the sum of the size of the car, its velocity-dependent braking distance and a safety margin.

Therefore, when there is no change of lane the motion of vehicles is made safe by the controller. Formally, for any safe snapshot $\mathscr{T}$, the snapshot $\mathscr{T}'$ after any time transition $\mathscr{T} \xrightarrow{t} \mathscr{T}'$ is safe:

$$\forall (C,D) \in \mathbb{I} \times \mathbb{I},\ se_{\mathscr{T}\mathscr{S}}(C) \cap se_{\mathscr{T}\mathscr{S}}(D) = \emptyset.$$

The aim of this section is to present a physical model of a vehicle, which describes the position $pos(C)$ and the speed $spd(C)$ of a vehicle C. It will lay the basis for the controller design in Section VI.

A vehicle C is characterised by its current velocity, $v_C : I \to \mathbb{R}_+$, defined on a time interval $I = [0,\tau]$ given in $[m/s]$. The length τ of the time interval is fixed but arbitrary – the maximal considered time interval between two consecutive snapshots.

The acceleration and braking of the vehicle C is realised by a torque $T \equiv T_C : I \to \mathbb{R}$ given in $[Nm]$. The torque is applied to the wheels from the transmission and braking system, and it belongs at any given time to an interval $[\underline{T},\overline{T}] \equiv [\underline{T_C},\overline{T_C}]$, where $\underline{T_C} < 0$ is the maximal torque of the brakes, and $\overline{T_C} > 0$ is the torque at full throttle.

To model aerodynamic drag force, we introduce a drag coefficient C_{W}. The drag force is proportional to the square of the velocity

$$C_{\text{W}}(t) v_C^2(t).$$

As indicated in the above equation, C_{W} varies in time, it depends on the distance δ between the cars C and D. The drag coefficient is an empirical quantity approximated by

$$C_{\text{W}}(\delta, v_D) = C_C \left(1 - \exp\left(-\frac{a\delta}{C_D v_D}\right)\right)^2,$$

where C_C, C_D are the aerodynamic coefficients of the cars C and D, and a is a constant [22]. In short, the aerodynamic coefficient of a vehicle is determined by its geometry: shape and size. The drag coefficient is positive, $\text{Image}(C_{\text{W}}) \subseteq [0, C_C]$. It converges to C_C for small distances δ and large velocities v_D.

The dynamics of the vehicle C is given by

$$(Mr^2 + J)\dot{v}_C(t) = -C_{\text{W}}(\delta(t), v_D(t)) r^2 v_C(t)^2 + rT(t),$$

where M is the mass of the vehicle C $[kg]$, J is the combined moments of inertia of the wheels $[kgm^2]$, and r is the radius of the wheels $[m]$.

Let X be the state space of the vehicle C (with the vehicle D driving in in front). It is the vector space of the velocity v_C of the vehicle C, and the distance δ from C to D, i.e., $X = \mathbb{R}^2$. We assume that both the velocity and the distance are available as indicated in Fig. 2, where sensor $\hat{v}$ measures v_C and $\hat{d}_1$ measures δ. If the vehicle D is out of range the distance sensor delivers the value ∞.

A feedback controller is a function $T : X \to [\underline{T},\overline{T}]$, which takes the current state to the torque. Negative values are realised by the braking system; whereas, the positive values are realised by the transmission (the throttle). As a consequence, $T(t) = T\left(v_C(t), \delta(t)\right)$.

To simplify the notation, we introduce

$$\begin{aligned}
x(t) &\equiv (\delta(t), v_C(t)) \in \mathbb{R}^2 \\
z(t) &\equiv v_D(t) \in \mathbb{R} \\
b &\equiv \frac{r}{Mr^2 + J} \in \mathbb{R} \\
a(x_1, z) &\equiv rbC_{\text{W}}(x_1, z) \in C^\infty(\mathbb{R}^2, \mathbb{R}_+) \\
u(t) &\equiv bT \in \mathbb{R} \\
(-\underline{u}, \overline{u}) &\equiv (-b\underline{T}, b\overline{T}) \in \mathbb{R}^2_+ \\
x_0 &\equiv (d^0, v_C^0) \in \mathbb{R}^2. \qquad (1)
\end{aligned}$$

As a result, the equations of motion are given by the following Cauchy problem with $x(0) = x_0$:

$$\begin{aligned}
\dot{x}_1(t) &= z(t) - x_2(t) \\
\dot{x}_2(t) &= -a(x_1(t), z(t)) x_2(t)^2 + u(t), \qquad (2)
\end{aligned}$$

where $u(t) \in [\underline{u}, \overline{u}]$. The subscripts of x refer to the components of the vector x.

Remark 1: The equation (2) can be used to compute the safety or braking distance $d_s(v_c^0)$ as a function of the initial velocity v_c^0 of the vehicle C. To this end, let $z(t) = 0$, i.e., the car in front instantaneously stops

$$\dot{x}_1(t) = -x_2(t) \quad \text{and} \quad \dot{x}_2(t) \leq \underline{u}$$

for $x_0 = (0, v_C^0)$. By the Gronwall theorem, the time to stop is $t \leq \hat{t} \equiv -\frac{v_C^0}{\underline{u}}$. Hence, the braking distance is at most $d_s(v_C^0) = -\frac{(v_C^0)^2}{2\underline{u}}$ (notice that $\underline{u} < 0$).

V. Linking

Using the abstract model and the concrete model, we must map the symbolic observables and events to observer functions in the controllers. In this work, we assume that each car is equipped with the observers, realised by suitable sensors, and actuators listed in Fig. 2.

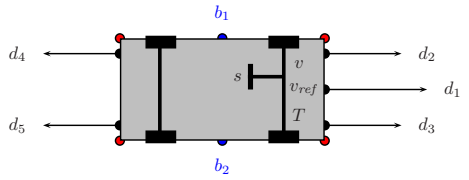
The abstract controller LPC takes a view of the traffic snapshot, represented by MLSL formulae built with the atoms $free, re(c), cl(c)$. By Theorem 1, this suffices for the safety check at the abstract level. However, the check *assumes* that the reserved or claimed spaces are large enough. Whether this assumption is true, depends on the concrete controller based on the car dynamics.

Here, we explain how the abstract controller LCP is linked to observers at the concrete level. The idea is that the MLSL formulae appearing as invariants and guards of LCP are replaced by comparisons of observer values read at the concrete level.

To have a perfect match between the two levels, we first introduce an intermediate level by modifying one invariant in the abstract controller LCP, namely the invariant ϕ_0 in state q_0, which in LCP formalises "no collision":

$$\neg cc \equiv \neg \exists c\colon\ c \neq ego \wedge \langle re(ego) \wedge re(c) \rangle\,.$$

Since the overlap $re(ego) \wedge re(c)$ is symmetric, the controller in *ego* must check forward or backward for any other car c.



We assume that each car is equipped with the following observers:

- $\hat{v}$ gives its own velocity,
- $\hat{d}_1$ gives the distance to the car ahead in the same lane,
- $\hat{d}_2$ ($\hat{d}_3$) give the distance to the car ahead in the left (right) neighboring lane,
- $\hat{d}_4$ ($\hat{d}_5$) give the distance to the car behind in the left (right) neighboring lane, and
- $\hat{b}_1$ ($\hat{b}_2$) tell whether a car on the lane next to the left (right) one is "blinking", indicating a desired lane change to the left (right) neighboring lane.

Also, each car has its blinkers and a torque T as actuators. Steering s and desired reference velocity v_{ref} are inputs from the driver.

Fig. 2. Car with observers and actuators

However, considering all cars together, it suffices that each car *ego* checks only that there is "no collision *forward*". Let *c ahead ego* abbreviate an MLSL formula expressing that car c is immediately ahead of *ego*. Then we replace the invariant ϕ_0 in state q_0 of the controller in Fig. 1 by the following formula:

$$\neg ccf \equiv \neg\exists c\colon c \neq ego \wedge \langle re(ego) \wedge re(c)\rangle \wedge \langle c \text{ ahead } ego\rangle\,.$$

We recall the resulting "forward looking" controller LCP_f. Note that logically $\neg ccf$ in LCP_f is *weaker* than $\neg cc$ in LCP, admitting more traffic snapshots. However, when all cars check $\neg ccf$ instead of $\neg cc$, safety remains guaranteed. This is formalized as follows. Consider the abstract setting A, where all cars are equipped with LCP, and the abstract forward setting A_f, where all cars are equipped with LCP_f.

Proposition 1 (Safety of LCP_f): Every transition among traffic snapshots permitted in A_f is also permitted in A.

Now we turn to the concrete setting. In the concrete controller, we will use the observables d_s yielding the safety distance needed for car *ego* at its current speed and $\hat{d}_1$ measuring the distance to the next car c ahead. The formula $\neg ccf$ is satisfied if the inequality $d_s < \hat{d}_1$ holds. Thus at the concrete level, we instantiate

$$\phi_0 \equiv d_s < \hat{d}_1\,,$$

which implies $\neg ccf$ at the abstract level, i.e., admits no more traffic snapshots than $\neg ccf$.

Next, consider the formula "no potential collision"

$$\neg pc \equiv \neg\exists c : c \neq ego \wedge \langle cl(ego) \wedge (re(c) \vee cl(c))\rangle\,,$$

which appears as invariant ϕ_2 in state q_2 and as guard of the transition from state q_2 to state q_3 in LCP, extending the reservation of *ego*. This guard is decisive for the safety of lane change.

To link $\neg pc$ with the concrete controller, we distinguish the cases of reservation and claim of c.

Case 1 : $\quad \phi_{re} \equiv \neg\exists c : c \neq ego \wedge \langle cl(ego) \wedge re(c)\rangle\,.$

This formula states that no (other) car c on *ego*'s target lane has a reservation that overlaps with *ego*'s claim. The car c may be (i) ahead of *ego* (or aligned with *ego*) or (ii) behind *ego*. In subcase (i), the concrete controller looks forward using the observables d_s giving the safety distance needed for car *ego* at its current speed and $\hat{d}_t$ (with t either 2 or 3) measuring the distance to the next car c in front of *ego* on the *target* lane of its lane change maneuver. The concrete controller checks the inequality $d_s < \hat{d}_t$. In subcase (ii), the concrete controller looks *backward* using the observables $\hat{d}_b$ (with b either 4 or 5) measuring the distance to the next car behind *ego* on the target lane and $d_{s,max}$, the maximal braking distance of any car, i.e., an *overapproximation* of the actual braking distance of that car. The concrete controller checks the inequality $d_{s,max} < \hat{d}_b$. Thus, ϕ_{re} is satisfied if $d_s < \hat{d}_t \wedge d_{s,max} < \hat{d}_b$ holds. Note that due to the overapproximation via $d_{s,max}$ this check may be *stronger* than necessary, permitting fewer lane changes than $\neg pc$, but this preserves safety.

Case 2 : $\quad \phi_{cl} \equiv \neg\exists c : c \neq ego \wedge \langle cl(ego) \wedge cl(c)\rangle\,.$

The formula states that no other car c on *ego*'s target lane has a claim that overlaps with *ego*'s claim. Such a car c may only be in a lane *next* to *ego*'s target lane. In this case, the concrete controller checks with its sensor b_t (with t either 1 or 2) on the side of the target lane for a turn signal of some car c on the lane next to the target lane. The formula ϕ_{cl} is satisfied if $\neg b_t$ holds. Thus at the concrete level, we instantiate

$$\phi_2 \equiv (d_s < \hat{d}_t \wedge d_{s,max} < \hat{d}_b) \wedge \neg b_t\,,$$

which implies $\neg pc$ at the abstract level.

Altogether, instantiating in the controller in Fig. 1 the formulae ϕ_0 and ϕ_2 by the distance inequalities and blinker sensor values as stated above, we obtain a concrete lane-change controller that we call LCP_c. Consider the concrete setting C, where all cars are equipped with LCP_c.

Proposition 2 (Safety of LCP_c): Every transition among traffic snapshots permitted in C is also permitted in A_f.

VI. Concrete Controllers

In this section, we deal with the assumption **A2** on the distance controller made in Theorem 1. We propose a sliding mode controller for a vehicle C that maintains the velocity of the vehicle at the reference v_{ref} until the distance d between C and the vehicle D in front is reached. Subsequently, the distance d is kept. If D is out of range of the distance sensor, the controller keeps the velocity at v_{ref}. In the following, we assume that at full throttle, the control $\overline{u}$ is strong enough to overcome the drag. To this end, we notice that $a(x_1,z) \in [0, rbC_C]$ for any $(x_1,z) \in R^2_+$, where the constant b is defined in (1). Let the speed limit be denoted by $\bar{v}$. Consequently, we assume that the maximal control $\overline{u} > rbC_C\bar{v}^2$. By a safe control, we understand a control that keeps the motion of a vehicle safe.

Definition 5 (Safe Control): A *safe controller* for the control system (2) and a function $z\colon \mathbb{R}_+ \to [0,\bar{v}]$ is a function $u\colon \mathbb{R}^3 \mapsto \mathbb{R}$ such that the solutions of the dynamical system (2)

with $u(t) = u(x(t), z(t))$ satisfy the following condition: If $x_1(0) \geq d$, then $x_1(t) > 0$ for all $t \in \mathbb{R}_+$.

In plain words, Definition 5 pronounces that an on-board controller is safe if: whenever the distance from the controlled car to a car in front is initially greater than d then a collision between these two cars will never happen.

Proposition 3 (Existence of a safe controller): Consider the control system (2) and a function $z : \mathbb{R}_+ \to [0, \bar{v}]$. Let $0 \leq v_{\text{ref}} < \bar{v}$, $d \equiv d(\bar{v})$, and $\alpha \equiv rbC_C\bar{v}^2$. Suppose that $\underline{u} < 0$. Let $k > 0$, define

$$L_1(x) \equiv x_2 - v_{\text{ref}},\ L_2(x,z) \equiv z - x_2 + k(x_1 - d),$$

and the following polyhedral set $P(z)$

$$P(z) \equiv \{x \in \mathbb{R}^2 |\ L_1(x) \leq 0 \text{ and } -L_2(x,z) \leq 0\}.$$

Then the following control

$$u(x,z) = \begin{cases} \underline{u} & \text{for} \quad x \in \mathbb{R}^2 \setminus P(z) \\ \overline{u} & \text{for} \quad x \in P(z) \end{cases} \tag{3}$$

is safe. Furthermore, the following two properties hold:

1) If $x_2(0) > v_{\text{ref}}$ then $x_2(t) < x_2(0)$ for all $t \in \mathbb{R}_+$ and there is $\tau \in \mathbb{R}^+$ such that $x_2(t) \leq v_{\text{ref}}$ for $t > \tau$.
2) Let $\beta \equiv \inf\{\dot{z}(t) |\ t \in \mathbb{R}_+\}$ and $\gamma \equiv \sup\{\dot{z}(t) |\ t \in \mathbb{R}_+\}$. Suppose that $\underline{u} < \beta$ and $\overline{u} > \alpha + \gamma$, and assume $0 < k < \min\{\beta - \underline{u}, \overline{u} - \alpha - \gamma\}/\bar{v}$. Then
   a) Let $0 \leq x_1(0) < d$, and suppose that $u(t) = \overline{u}$ holds on an interval $[0, \tau]$. Then $x_1(t) > x_1(0)$ for all $t \in [0, \tau]$.
   b) $\lim_{t\to\infty} x_1(t) = d$.

Proof: If $x_1(0) \in \mathbb{R}^2 \setminus P(z)$, then the following holds. There is a family of open intervals $\{(\underline{\tau}_\alpha, \overline{\tau}_\alpha) |\ \alpha \in \Lambda\}$ such that $x(\underline{\tau}_\alpha) \in P(z)$ and if $t \in (\underline{\tau}_\alpha, \overline{\tau}_\alpha)$ then $x(t) \in \mathbb{R}^2 \setminus P(z)$, hence $u(t) = \underline{u}$, and $x_1(t) > 0$. If $t \in \mathbb{R} \setminus \bigcup_{\alpha\in\Lambda}(\underline{\tau}_\alpha, \overline{\tau}_\alpha)$ then $x \in P(z)$, and thus $x_1(t) \geq d$.

We prove Property 1 and Property 2 of the proposition. To this end, we observe that for $x \in \mathbb{R}^2 \setminus P(z)$,

$$\dot{L}_1(x,z) = -a(x_1,z)x_2^2 + \underline{u} \leq \underline{u} < 0 \tag{4}$$

$$\begin{aligned} \dot{L}_2(x,z,\dot{z}) &= \dot{z} + a(x_1,z)x_2^2 + k(z - x_2) - \underline{u} \\ &\geq \beta - k\bar{v} - \underline{u} > 0. \end{aligned} \tag{5}$$

Whereas, for $x \in P(z)$,

$$\dot{L}_1(x,z) = -a(x_1,z)x_2^2 + \overline{u} \geq -\alpha + \overline{u} > 0 \tag{6}$$

$$\begin{aligned} \dot{L}_2(x,z,\dot{z}) &= \dot{z} + a(x_1,z)x_2^2 + k(z - x_2) - \overline{u} \\ &\leq \gamma + \alpha + k\bar{v} - \overline{u} < 0. \end{aligned} \tag{7}$$

By (4), Property 1 holds.

We will show Property 2.a. To this end, we notice that $u(t) = \overline{u}$ whenever $x(t) \in P_{z(t)}$. We consider two cases $z(t) > x_2(t)$ and $z(t) \leq x_2(t)$. If $z(t) > x_2(t)$ then $\dot{x}_1(t) = z(t) - x_2(t) > 0$ and Property 2.a follows. Suppose that $z(t) \leq x_2(t)$. Then $0 < k(x_1(t) - d) \geq z(t) - x_2 + k(x_1(t) - d) = L_2(x(t), z(t)) \geq 0$, which is a contradiction.

To show Property 2.b, we observe that by Inequalities (4)–(7), any flow line of (2) intersects the boundary of P at a point say $\tilde{x}$ (transversally), i.e., there is $t_1 \geq 0$ such that $x(t_1) = \tilde{x}$. If $L_1(\tilde{x}) = 0$, then the solution (in a Filippov sense) $x(\cdot)$ is such that $L_1(x(t)) = 0$ for all $t \in [t_1, t_2]$, where t_2 is the time at which $L_2(x(t_2), z(t_2)) = 0$. Subsequently, the Fillipov solution $x(\cdot)$ is such that $L_2(x(t), z(t)) = 0$ for all $t \geq t_2$. As a consequence, $z(t) - x_2(t) + k(x_1(t) - d) = 0$, which is equivalent to

$$\frac{\mathrm{d}}{\mathrm{d}t}(x_1(t) - d) = -k(x_1(t) - d).$$

Hence, $\lim_{t\to\infty} x_1(t) = d$. ∎

The above proposition shows that there is a control that keeps the distance from the vehicle C to the vehicle in front safe while the velocity of C does not exceed the reference. Also whenever the vehicle C accelerates, $u(t) = \overline{u}$, and initially the distance $x_1(0)$ is less than d then the distance increases, i.e., the traffic situation is not less safe than it was at the beginning. If the distance between C and D was greater than d then there is no future time that they will hit each other.

To avoid discontinuous control and hence abrupt switches between acceleration $\overline{u}$ and deceleration $\underline{u}$, the control (3) can be replaced by a continuous approximation. The following notation will be instrumental:

$\mathbb{L}_1 \equiv L_1^{-1}(0) = \{x \in \mathbb{R}^2 |\ L_1(x) = 0\}$,
$\mathbb{L}_{2,z} \equiv \{x \in \mathbb{R}^2 |\ L_2(x,z) = 0\}$, and
$\mathbb{H}_1 \equiv \{x \in \mathbb{R}^2 |\ L_1(x) \leq 0\}$, $\mathbb{H}_{2,z} \equiv \{x \in \mathbb{R}^2 |\ -L_2(x,z) \leq 0\}$.

For an $\varepsilon > 0$, we define a map $h : [-\varepsilon, \varepsilon] \to [0,1]$ by $y \mapsto \frac{1}{2}\left(\frac{1}{\varepsilon}y + 1\right)$. Let $\mathbb{L}_1^\varepsilon$ be the ε-neighborhood of $\mathbb{L}_1$ (with respect to the Hausdorff metric), $\mathbb{L}_{2,z}^\varepsilon$ be the ε-neighborhood of $\mathbb{L}_{2,z}$, $\mathbb{H}_1^\varepsilon$ be the ε-neighborhood of $\mathbb{H}_1$, and $\mathbb{H}_{2,z}^\varepsilon$ be the ε-neighborhood of $\mathbb{H}_{2,z}$. Furthermore, we define $P^\varepsilon(z)$ by

$$P^\varepsilon(z) \equiv \mathbb{H}_1^\varepsilon \cap \mathbb{H}_{2,z}^\varepsilon.$$

Let $x^i(x) = x - \pi_{\mathbb{L}_i}(x)$, where $\pi_{\mathbb{L}_1}$ and $\pi_{\mathbb{L}_2}$ are the projection on $\mathbb{L}_1$ and $\mathbb{L}_{2,z}$, respectively.

For $l \equiv l(x) = \text{argmax}\{|x^i(x)||\ i \in \{1,2\}\}$ let

$$y(x) = |x^l| \,\text{sign}(\langle n^l, x^l \rangle),$$

where $\langle \cdot, \cdot \rangle$ is the scalar product on $\mathbb{R}^2$, n^1 and n^2 are the normal vectors to $\mathbb{L}_1(\cdot)$ and $\mathbb{L}_{2,z}(\cdot)$,

$$n^1 = (0,-1), n^2 = (k,-1).$$

Let $P^{-\varepsilon}(z) \equiv P^\varepsilon(z) \setminus (\mathbb{R}^2 \setminus (\mathbb{H}_1^\varepsilon \cup \mathbb{H}_{2,z}^\varepsilon))$. We define $\bar{h} : P^{-\varepsilon}(z) \to [0,1]$ by

$$\bar{h}(x) = h(y(x)).$$

The control is then $u(x,z) =$

$$\begin{cases} \underline{u} & \text{for} \quad x \in \mathbb{R}^2 \setminus P^\varepsilon(z) \\ (1 - \bar{h}(x))\underline{u} + \bar{h}(x)\overline{u} & \text{for} \quad x \in P^{-\varepsilon}(z) \\ \overline{u} & \text{for} \quad x \in P(z) \setminus P^{-\varepsilon}(z). \end{cases}$$

The parameter ε is to be chosen as a tradeoff between accuracy of tracking the distance d and "evenness" of the control.

A. Lateral Motion

So far, we have not discussed lateral motion. For the details of modeling, we refer to [15]. In short, the kinematic model of the vehicle C is given by the global position

$$\dot{X} = v_C \cos(\psi + \beta) \quad \text{and} \quad \dot{Y} = v_C \sin(\psi + \beta), \tag{8}$$

where v_C is the velocity of the vehicle C, β is the slip angle of the vehicle defined below, and ψ is the yaw angle, which defines the orientation angle of the vehicle w.r.t. the x-axis

$$\dot{\psi} = \frac{v_C}{l} \cos(\beta) \tan(\theta). \tag{9}$$

In (9), l is the vehicle base, the distance between the rear and the front wheels, and θ is the angle between the front wheel and the longitudinal axis of the vehicle, with $\theta \in [\underline{\theta}, \overline{\theta}]$ for $\underline{\theta} < 0$ and $\overline{\theta} > 0$; θ as the control input.

The slip angle of the vehicle is given be the relation

$$\beta \equiv \beta(\theta) = \tan^{-1}\left(\frac{l_{\mathrm{r}} \tan(\theta)}{l}\right),$$

where l_{r} is the distance between the center of gravity and the rear wheel.

The control for lateral motion is discussed in [15]. For completeness of our study, we propose a facile feedforward control for changing the lane. To this end, we introduce

$$b(\theta) \equiv \frac{v_C}{l} \cos(\beta(\theta)) \tan(\theta).$$

With this notation, the solution of (8) and (9) belongs to the graph of the function F_{θ,y_0,ψ_0}

$$F_{\theta,y_0,\psi_0} : \psi \mapsto \tilde{y}_0(y_0, \psi_0) - \frac{v_C}{b(\theta)} \cos(\psi + \beta(\theta)),$$

where $\tilde{y}_0(y_0, \psi_0) = y_0 + \frac{v_C}{b(\theta)} \cos(\psi_0 + \beta(\theta))$, and y_0 is the initial lateral position, and ψ_0 is the initial orientation angle.

To change the lane, we change the state (Y, ψ) from $(y_0, 0)$ to $(y_1, 0)$, where without loss of generality it is assumed that $y_0 > y_1$. We suppose that the velocity v_C during the entire maneuver is kept constant. Suppose that $(\theta_0, \theta_1) \in [\underline{\theta}, 0) \times (0, \overline{\theta}]$ are such that the equation $F_{\theta_0,y_0,0}(\psi) = F_{\theta_1,y_1,0}(\psi)$, or equivalently

$$\begin{aligned} &\tilde{y}_0(y_0, 0) - \tilde{y}_0(y_1, 0) \\ &+ v_c \left(\frac{\cos(\psi + \beta(\theta_1))}{b(\theta_1)} - \frac{\cos(\psi + \beta(\theta_0))}{b(\theta_0)} \right) = 0, \end{aligned}$$

has the solution $\hat{\psi}$. The proposed maneuver turns the front wheels from 0 to the angle $\theta_0 > 0$, waiting until the orientation angle ψ is $\hat{\psi}$, and then turns the wheels to the angle $\theta_1 < 0$, waiting until the orientation angle ψ reaches 0, and finally turns the front wheels back to 0. The proposed control is feedforward, thus a linear control [15] is to be implemented to remove deviations from the lateral reference y_1.

The time t_{lc} of the maneuver depends on the vehicle velocity, v_C, and it is used in the guard of the abstract controller LCP depicted in Fig. 1. To avoid a collision during the maneuver of changing the lanes, it is assumed that the minimum distance d to the front cars in the current lane and the neighboring target lane is big enough, i.e., greater than the sum of the maximal braking distance of the car C and the distance $v_C t_{lc}$ traveled by C during the lane change.

VII. Conclusion

The paper has presented an approach to hybrid systems modelling where an abstract model is built in theories that are decidable modulo symbolic guards and actions while a concrete model uses conventional continuous time for which controllers are developed. The key point is that these two worlds are linked by *linking predicates*, so the concrete model is a refinement of the abstract one. Several approaches to controller design for hybrid systems have pursued a *separation* of the dynamics from the control layer. Specifically, Raisch et al. [13], [12] introduce abstraction and refinement to support a hierarchical design. However, this line of work stays within the same underlying model, whereas the work here operates with separate models, more in accordance with the work in [16], which treats semantic alignment of heterogeneous models. The linking predicates used in the current work may make alignment easier, because they relate specific quantities and not full models.

Symbolic Models are well known from a controller side, which can be built using timed automata. Also the use of symbolic guards and actions is intuitively easy. When this is done, it is feasible to use model checking with a simplified environment model that assigns values from (very) finite domains to predicates, and actions.

Defining a suitable state space is intrinsically difficult. We have used a spatial logic to structure it, also for other types of roads [6]. The logic gives a compact formulation of properties and configurations, and an ability to compose and decompose them as well as a potential for deductions [9]. Work on spatial logic often focuses on qualitative spatial reasoning [20] as exemplified in the region connection calculus [17]. The logic used here is inspired by the Duration Calculus [23], and the Shape Calculus [18]. In [19], hybrid automata are considered where invariants and guards are expressed in a spatio-temporal logic $\mathrm{S4}_u$, yet without separation of space and dynamics as here.

Concrete Models are highly application dependent. In the current presentation, the modelling and controller design is kept general. For real applications, there is much detailed engineering to do, but this is not in the scope of this paper. During the development, one must have an eye on the predicates of the symbolic model, so it is feasible to construct observers that match the guards, and handle set points presented by the actions.

Linkage. The linking predicates are the formal outcome of negotiating the interface between the two models. They represent the point where many real application projects fail, because engineering traditions from software development and control system development meet without explicit negotiation. The transfer of temporal properties from abstract to concrete transitions systems via (bi)simulations is well-understood in the area of model checking, see [5].

Overall the approach seems well suited for application areas, where semi-autonomous entities have to coordinate to achieve common objectives. In a tightly coupled application, it is most likely easier to stay with a one level concrete model, typically a conventional hybrid automaton.

Traffic Maneuvers There are various other approaches dealing with the safety of traffic maneuvers. A very influential effort was the California PATH (Partners for Advanced Transit and Highways) project on automated highway systems for cars driving in groups called platoons [21]. The maneuvers include joining and leaving the platoon, and lane change. Lygeros et al. [11] sketch a safety proof for car platoons taking car dynamics into account, but admitting *safe collisions*, i.e., collisions at a low speed. Not all scenarios of multi-lane traffic are covered in their proof. More recently, Platzer et al. [3], [10] represent traffic applications in a differential dynamic logic $d\mathscr{L}$ that is supported by the theorem prover KeYmaera [14]. This logic does not separate space (symbolic model) from dynamics (concrete model), which is at the heart of our approach. The paper [1] proposes a bottom-up strategy, where a concrete model is gradually abstracted to Markov chains, of which the set of reachable states is analyzed.

Acknowledgments: This work was partially supported by the German Research Foundation (DFG) in the Transregional Collaborative Research Center (SFB/TR 14) AVACS (www.avacs.org).

REFERENCES

[1] M. Althoff, O. Stursberg, and M. Buss. Safety assessment of autonomous cars using verification techniques. In *American Control Conference (ACC) 2007*, pages 4154–4159. IEEE, 2007.

[2] R. Alur and D. L. Dill. A theory of timed automata. *TCS*, 126(2):183 – 235, 1994.

[3] N. Arechiga, S. M. Loos, A. Platzer, and B. H. Krogh. Using theorem provers to guarantee closed-loop system properties. In *American Control Conference (ACC) 2012*, pages 3573–3580. IEEE, 2012.

[4] W.-P. de Roever and K. Engelhardt. *Data Refinement: Model-Oriented Proof Methods and their Comparison*. Cambridge Univ. Press, 1998.

[5] O. Grumberg. Abstraction and reduction in model checking. In H. Schwichtenberg and R. Steinbrüggen, editors, *Proof and System-Reliabilty*, volume 62 of *Nato Science Series II. Math., Physics and Chemistry*, pages 213–260. Kluwer Academic Publishers, 2002.

[6] M. Hilscher, S. Linker, and E.-R. Olderog. Proving safety of traffic manoeuvres on country roads. In Zhiming Liu, Jim Woodcock, and Huibiao Zhu, editors, *Theories of Programming and Formal Methods*, volume 8051 of *LNCS*, pages 196–212. Springer, 2013.

[7] M. Hilscher, S. Linker, E.-R. Olderog, and A. Ravn. An abstract model for proving safety of multi-lane traffic manoeuvres. In Shengchao Qin and Zongyan Qiu, editors, *ICFEM 2011*, volume 6991 of *LNCS*, pages 404–419. Springer, 2011.

[8] C. A. R. Hoare and Jifeng He. *Unifying Theories of Programming*. Prentice Hall, 1998.

[9] S. Linker and M. Hilscher. Proof theory of a multi-lane spatial logic. In Zhiming Liu, Jim Woodcock, and Huibiao Zhu, editors, *Int'l Conf. on Theoret. Aspects of Comput. (ICTAC)*, volume 8049 of *LNCS*, pages 231–248. Springer, 2013.

[10] S. M. Loos, A. Platzer, and L. Nistor. Adaptive cruise control: Hybrid, distributed, and now formally verified. In M. J. Butler and W. Schulte, editors, *FM 2011*, volume 6664 of *LNCS*, pages 42–56. Springer, 2011.

[11] J. Lygeros, D. N. Godbole, and S. S. Sastry. Verified hybrid controllers for automated vehicles. *IEEE Trans. on Automatic Control*, 43(4):522–539, 1998.

[12] T. Moor, J. Raisch, and J.M Davoren. Admissiblity criteria for a hierarchical design of hybrid systems. In *Proc. IFAD Conf. on Analysis and Design of Hybrid Systems*, pages 389–394, St. Malo, France, 2003.

[13] T. Moor, J. Raisch, and S. O'Young. Discrete supervisory control of hybrid systems based on l-complete approximations. *Discrete Event Dynamic Systems*, 12:83–107, 2002.

[14] A. Platzer. *Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics*. Spinger, 2010.

[15] R. Rajamani. *Vehicle Dynamics and Control*. Mechanical engineering series. Springer Science, 2006.

[16] A. Rajhans and B. H. Krogh. Compositional heterogeneous abstraction. In *HSCC 2013*, pages 253–262. ACM, 2013.

[17] D. A. Randell, Zhan Cui, and A. G. Cohn. A spatial logic based on regions and connection. In *Proc. 3rd Int'l Conf. Knowledge Representation and Reasoning*, 1992.

[18] A. Schäfer. A calculus for shapes in time and space. In Z. Liu and K. Araki, editors, *ICTAC 2004*, volume 3407 of *LNCS*, pages 463–478. Springer, 2005.

[19] Zhucheng Shao and Jing Liu. Spatio-temporal hybrid automata for cyber-physical systems. In Z. Liu, J. Woodcock, and H. Zhu, editors, *ICTAC 2013*, volume 8049 of *LNCS*, pages 337–354. Springer, 2005.

[20] J. van Benthem and G. Bezhanishvili. Modal logics of space. In M. Aiello, I. Pratt-Hartmann, and J. Benthem, editors, *Handbook of Spatial Logics*, pages 217–298. Springer Netherlands, 2007.

[21] P. Varaija. Smart cars on smart roads: problems of control. *IEEE Transactions on Automatic Control*, AC-38(2):195–207, 1993.

[22] M. Zabat, N. Stabile, S. Farascaroli, and F. Browand. The aerodynamic performance of platoons: A final report. http://escholarship.org/uc/item/8ph187fw, UC Berkeley, 1995.

[23] C. Zhou, C.A.R. Hoare, and A.P. Ravn. A calculus of durations. *IPL*, 40(5):269–276, 1991.