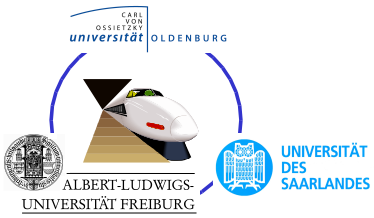

AVACS – Automatic Verification and Analysis of Complex Systems

REPORTS

of SFB/TR 14 AVACS

Editors: Board of SFB/TR 14 AVACS

Undecidability Results for Multi-Lane Spatial Logic

by
Heinrich Ody

AVACS Technical Report No. 112
September 2015
ISSN: 1860-9821

Publisher: Sonderforschungsbereich/Transregio 14 AVACS (Automatic Verification and Analysis of Complex Systems)
Editors: Bernd Becker, Werner Damm, Bernd Finkbeiner, Martin Fränzle, Ernst-Rüdiger Olderog, Andreas Podelski
ATRs (AVACS Technical Reports) are freely downloadable from www.avacs.org

Author(s) contact: Heinrich Ody (heinrich.ody@uni-oldenburg.de).

Undecidability Results for Multi-Lane Spatial Logic*

Heinrich Ody

Department of Computing Science, University of Oldenburg, Germany
heinrich.ody@uni-oldenburg.de

Abstract. We consider (un)decidability of Multi-Lane Spatial Logic (MLSL), a multi-dimensional modal logic introduced for reasoning about traffic manoeuvres. MLSL with length measurement has been shown to be undecidable. However, the proof relies on exact values. This raises the question whether the logic remains undecidable when we consider robust satisfiability, i.e. when values are known only approximately. Our main result is that robust satisfiability of MLSL is undecidable. Furthermore, we prove that even MLSL without length measurement is undecidable. In both cases we reduce the intersection emptiness of two context-free languages to the respective satisfiability problem.
This is the full version of a paper with the same title published at the ICTAC 2015.

Keywords. Robustness, interval logic, spatial logic, undecidability.

1 Introduction

To reason about configurations of cars on a motorway in a formal manner we need an abstract representation of a motorway and the cars on it. In [13] the authors model a motorway as a set of lanes, where the extension of a lane is represented by the real numbers. The space occupied by a car then is a subinterval of the real numbers, together with the lane the car is located on. Additionally, the authors define Multi-Lane Spatial Logic (MLSL) to express topological properties of the cars on the motorway from the local perspective of a car. Possible properties are, e.g. 'there is a car on an adjacent lane' or 'there is some free space in front of us' or 'there is a collision', i.e. an overlap of occupied spaces. MLSL then has been extended to support length measurement and dynamic modalities [16,12]. With the dynamic modalities specifications of, e.g. a distance controller can be expressed in the logic. With length measurement properties such as 'there are five units of free space in front of me' can be expressed. MLSL with length measurement has been proven to be undecidable [16].

However, in all variants of MLSL considered it was assumed that data are known exactly, e.g. the position of every car is known exactly. Considering that the logic is used to express spatial properties of traffic on a motorway and that data are obtained by imperfect sensors this assumption is too idealistic. This raises the question, whether the logic becomes decidable when values are known only approximately. Consider Metric Temporal Logic (MTL) [15] as an example where weakening the assumption of working with exact values leads to decidability. MTL is undecidable [10] and the proof heavily relies on formulas of the form $\Box(P \implies \Diamond_{=1} Q)$, which specifies that always exactly 1 time unit after P holds Q will hold. However, the fragment Metric Interval Temporal Logic, where exact timing properties are not expressible, is decidable [3].

As an intermediate result we show that even without length measurement MLSL is undecidable. This is surprising, because in this fragment we can only express topological relations between cars. Even though this fragment does not contain length measurement it still assumes exact positions. To formalize that values are known only approximately we consider a *robust* satisfaction relation, where a model robustly satisfies a formula iff all similar models also satisfy the formula. Our main

* Work of the author is supported by the Deutsche Forschungsgemeinschaft (DFG) within the Research Training Group DFG GRK 1765 SCARE and the Transregional Collaborative Research Center SFB/TR 14 AVACS.

result is that robust satisfiability of MLSL without length measurement is undecidable. To prove this we reduce the intersection emptiness of two context-free languages to the robust satisfiability problem. Our proof also works for a variety of restrictions on MLSL, e.g. when the extension of the lanes is represented by natural numbers instead of real numbers or when the minimal or maximal size of cars is bounded or any combination of these.

Robustness of spatial logics received little attention so far, because most spatial logics consider qualitative properties. However, as MLSL is inspired by temporal logics, robustness of timed systems is related. Our definition of robust satisfaction is similar to other approaches to introduce robustness into a formalism. A robust timed automaton accepts a trajectory iff it also accepts all slightly perturbed trajectories [8]. They were defined with the hope that by making exact timing properties inexpressible, the universality problem (does an automaton accept all trajectories) would become decidable, similarly to the satisfiability problem for MITL. However, the universality problem remains undecidable for robust timed automata [11]. In [7] a robust interpretation for a fragment of Duration Calculus has been defined. Whether this fragment is decidable with this robust interpretation is not known. For a survey on robustness in timed systems we suggest [1].

MLSL is inspired by DC [6], Interval Temporal Logic [17] and Shape Calculus [18]. Other logics similar to MLSL are CDT [19] and Halpern-Shoam-logic (HS) [9]. CDT and HS are one dimensional modal interval logics used to express temporal properties. In both of these logics topological properties of intervals can be expressed. However, while in MLSL there is a constant number of atomic propositions, in CDT and HS there is an arbitrary but finite number of atomic propositions. Further, both of these logic leave the temporal structure quite unconstrained, i.e. time may be branching or linear, dense or discrete.

2 Multi-Lane Spatial Logic

With Multi-Lane Spatial Logic (MLSL) we reason about static motorway traffic configurations. To this end we model the configuration of a motorway at a specific point in time as a *traffic snapshot*. In such a traffic snapshot the motorway is represented in a two dimensional manner, i.e. one vertical and one horizontal dimension. The vertical dimension represents the lane a car is located on and the horizontal dimension represents the position along the lane. A car then *reserves* the part of a lane where it is currently driving. When a car is changing lanes it has multiple reservations on adjacent lanes.

In the original definition of MLSL the model contains information about whether a car intents to change lanes (there called claims), the perspective from which formulas are evaluated (there called ego) and how a traffic snapshot changes as time progresses. We do not need this expressiveness to show undecidability. Hence, we consider a simplified logic without this information. As original MLSL is a conservative extension of our simplified logic our results also hold for the original logic.

With small adaptions our definitions in this section are taken from [13,16].

2.1 The Model

To formally define a traffic snapshot, we assume a countably infinite set of globally unique *car identifiers* $\mathbb{I}$ and an arbitrary but fixed set of lanes $\mathbb{L} = \{0, \dots, k\}$, for some $k \in \mathbb{N}_{\geq 1}$.

Definition 1 (Traffic snapshot). *A* traffic snapshot $\mathcal{TS}$ *is a structure* $\mathcal{TS} = (res, pos, br\text{-}dis)$*, where* $res, pos, br\text{-}dis$ *are functions*

- $res : \mathbb{I} \to \mathcal{P}(\mathbb{L})$ *such that* $res(C)$ *is the set of lanes the car* C *reserves,*
- $pos : \mathbb{I} \to \mathbb{R}$ *such that* $pos(C)$ *is the position of the car* C *along the lanes,*
- $br\text{-}dis : \mathbb{I} \to \mathbb{R}_{>0}$ *such that* $br\text{-}dis(C)$ *comprises the braking distance and the physical size of* C*,*

where $\mathcal{P}(\mathbb{L})$ *denotes the powerset of* $\mathbb{L}$*.*

In the original definition the authors imposed sanity conditions on a traffic snapshot, such as $1 \leq |res(C)| \leq 2$ holds for all cars C [13]. Our results hold regardless of whether these conditions are assumed. The extension of a lane occupied by a reservation is given by the *safety envelope* of a car, which is defined as

$$se(C) = [pos(C), pos(C) + br\text{-}dis(C)]\,.$$

We reason only about a finite part of a traffic snapshot. In the original definition of MLSL this is called a *view* and we keep this name here.

Definition 2 (View). *For a given traffic snapshot $\mathcal{TS}$ with a set of lanes $\mathbb{L}$, a* view *V is defined as a structure $V = (L, X)$, where*

- *$L \subseteq \mathbb{L}$ is an interval of lanes that are visible in V,*
- *$X \subseteq \mathbb{R}$ is an interval representing the extension of the lanes visible in V.*

A subview *of V is obtained by restricting the lanes and extension we observe. Let L', X' be subintervals of L and X, then we define*

$$V^{L'} = (L', X) \qquad \text{and} \qquad V_{X'} = (L, X')\,.$$

If the interval of lanes is empty or the extension is a point interval we say that the view is empty*.*

To give the semantics of MLSL in a uniform way we define chopping of views into subviews.

Definition 3 (Chopping Views). *Let $V_i = (L_i, X_i)$ be views with $i \in \{0, 1, 2\}$ and $X_i = [r_i, t_i]$. Then we define* vertical chopping *(denoted by $\ominus$) and* horizontal chopping *(denoted by $\oplus$) of V_0 into V_1 and V_2 as*

$$\begin{aligned} V_0 = V_1 \ominus V_2 \;\text{ iff }\; & L_0 = L_1 \cup L_2 \;\text{ and }\; L_1 \cap L_2 = \emptyset \;\text{ and }\; X_0 = X_1 = X_2 \;\text{ and} \\ & (L_1 = \emptyset \;\text{ or }\; L_2 = \emptyset \;\text{ or }\; \max(L_1) + 1 = \min(L_2))\,, \\ V_0 = V_1 \oplus V_2 \;\text{ iff }\; & t_1 = r_2 \;\text{ and }\; r_0 = r_1 \;\text{ and }\; t_0 = t_2 \;\text{ and }\; L_0 = L_1 = L_2\,. \end{aligned}$$

The intuition is that after vertical chopping the lane intervals of the subviews are adjacent and non-overlapping because each lane should belong to exactly one subview. After horizontal chopping the new subviews are adjacent and share a common point, which is the endpoint of the left subview and the startpoint of the right subview. Note that a nonempty view can be chopped into an empty and a nonempty subview and that an empty view can be chopped into two empty subviews.

Let *CVar* be the set of variables ranging over $\mathbb{I}$. A valuation ν is a function $\nu : CVar \to \mathbb{I}$ that maps car variables to car identifiers. Additionally we define valuation updates with the override notation $\oplus$ from Z [20] as $\nu \oplus \{c \mapsto C\}(c') = C$ if $c = c'$ and $\nu(c')$ otherwise.

Definition 4 (Model). *Let $\mathcal{TS}$ be a traffic snapshot, V a view and ν a valuation, then we call $\mathcal{M} = (\mathcal{TS}, V, \nu)$ a model of MLSL.*

2.2 The Logic

The atoms of MLSL are used to express that some part of a lane is filled by a reservation or completely free of reservations. The chop operators in the logic are defined using the chop operators on views. The horizontal chop formula $\phi_0 \frown \phi_1$ expresses that on the left subview ϕ_0 and on the right subview ϕ_1 holds. With the vertical chop formula $\begin{smallmatrix}\phi_1\\ \phi_0\end{smallmatrix}$ we require that the lower subview satisfies ϕ_0 and that the upper subview satisfies ϕ_1. Note that for both chop formulas the satisfying subviews might be empty. Additionally, the logic is closed under first order operators.

Definition 5 (Syntax). *Given variables $c, c' \in CVar$ the syntax of MLSL is given by*

$$\phi ::= c = c' \mid \mathit{free} \mid \mathit{re}(c) \mid \neg\phi \mid \phi \wedge \phi \mid \exists c.\, \phi \mid \phi \frown \phi \mid \begin{smallmatrix}\phi\\ \phi\end{smallmatrix}\,.$$

Definition 6 (Semantics). *Let $c, c' \in CVar$. Then, given a traffic snapshot $\mathcal{TS}$, a view $V = (L, X)$ with $X = [r, t]$, and a valuation ν we define the* satisfaction *of a formula by a model $\mathcal{M} = (\mathcal{TS}, V, \nu)$ as*

$$
\begin{aligned}
\mathcal{M} &\models c = c' && \text{iff} \quad \nu(c) = \nu(c'),\\
\mathcal{M} &\models \mathit{free} && \text{iff} \quad |L| = 1 \text{ and } r < t \text{ and}\\
&&& \quad (\forall C \in \mathbb{I}.\, L \not\subseteq \mathit{res}(C) \text{ or } \mathit{se}(C) \cap (r, t) = \emptyset),\\
\mathcal{M} &\models \mathit{re}(c) && \text{iff} \quad |L| = 1 \text{ and } r < t \text{ and}\\
&&& \quad L \subseteq \mathit{res}(\nu(c)) \text{ and } X \subseteq \mathit{se}(\nu(c)),\\
\mathcal{M} &\models \neg\phi && \text{iff} \quad \mathcal{M} \not\models \phi,\\
\mathcal{M} &\models \phi_0 \wedge \phi_1 && \text{iff} \quad \mathcal{M} \models \phi_0 \text{ and } \mathcal{M} \models \phi_1,\\
\mathcal{M} &\models \exists c.\, \phi && \text{iff} \quad \exists C \in \mathbb{I}.\, \mathcal{TS}, V, \nu \oplus \{c \mapsto C\} \models \phi,\\
\mathcal{M} &\models \phi_0 \frown \phi_1 && \text{iff} \quad \exists V_0, V_1.\, V = V_0 \mathbin{\odot} V_1 \text{ and}\\
&&& \quad \mathcal{TS}, V_0, \nu \models \phi_0 \text{ and } \mathcal{TS}, V_1, \nu \models \phi_1,\\
\mathcal{M} &\models \begin{matrix}\phi_1\\ \phi_0\end{matrix} && \text{iff} \quad \exists V_0, V_1.\, V = V_0 \ominus V_1 \text{ and}\\
&&& \quad \mathcal{TS}, V_0, \nu \models \phi_0 \text{ and } \mathcal{TS}, V_1, \nu \models \phi_1\,.
\end{aligned}
$$

In addition we make use of the standard abbreviations such as $\mathit{true}, \vee, \forall$. Additionally, we use a derived modality to express that *somewhere* on the motorway ϕ holds. It is defined by using both chop operators as

$$\langle\phi\rangle \stackrel{\text{def}}{=} \mathit{true} \frown \begin{pmatrix}\mathit{true}\\ \phi\\ \mathit{true}\end{pmatrix} \frown \mathit{true}\,.$$

3 Undecidability of MLSL

Undecidability of Duration Calculus was proven by a reduction to the halting problem of a two counter machine [5]. This construction has been adapted to prove undecidability of MLSL with length measurement, where length measurement allows to compare the size of the current view to a constant [16]. The authors defined that the representation of a configuration of a two counter machine is of length $5k$, where k is a constant. The value m of a counter was represented in an interval of length k and consisted of m reservations and the remaining space of a configuration was used for markers to separate the counters. To increase the value of a counter they required that for all reservations part of the representation of the counter there is a reservation $5k$ space units later (in the next representation of the counter's value) and additionally, there is exactly one reservation in the later representation for which there is no reservation $5k$ space units earlier. For this construction it is important to be able to specify the distance between reservations.

The construction from [16] does not work in our setting, as we consider MLSL without length measurement. Instead, we reduce the emptiness problem of the intersection of two context-free languages, which is undecidable [14], to the satisfiability problem of MLSL. For the reduction we create a formula such that satisfying models represent derivations of two context-free grammars for the same word. A letter is represented as a fixed number of successive reservations. Different letters are represented by a different number of reservations in their representation. Letters are separated by free space, and rewrite steps are represented by different lanes.

Definition 7 (Context-Free Grammar). *A* context-free grammar *(CFG) is a tuple $G = (\mathcal{N}, \mathcal{T}, \mathcal{R}, S)$, where $\mathcal{N}$ is the set of* nonterminals*, $\mathcal{T}$ with $\mathcal{T} \cap \mathcal{N} = \emptyset$ is the set of* terminals*, $S \in \mathcal{N}$ is the* starting *nonterminal and $\mathcal{R} \subseteq \mathcal{N} \times (\mathcal{T} \cup \mathcal{N})^*$ is the set of* rewrite rules*. A rewrite rule $(N, w) \in \mathcal{R}$ is usually written as $N \to w$. We extend $\to$ as follows. Let $\mathcal{R}^N = \{w \mid (N, w) \in \mathcal{R}\}$, i.e. the set of words that may replace N. Let $u, v, w \in (\mathcal{T} \cup \mathcal{N})^*, N \in \mathcal{N}$ then $uNv \to uwv$ with $(N, w) \in \mathcal{R}$ is a* rewrite step*. A sequence of rewrite steps is a* derivation*. The* language *$\mathcal{L}(G)$ of*

a CFG G is the set of terminal words reachable with a derivation that starts with S. We refer to $\sigma \in \mathcal{N} \cup \mathcal{T}$ as letter.

All grammars we consider are in Chomsky normal form. With the definition we use this implies that the empty word is not in the language of a grammar in Chomsky normal form. Still the language intersection problem of these grammars remains undecidable.

Definition 8 (Chomsky Normal Form). *A CFG $G = (\mathcal{N}, \mathcal{T}, \mathcal{R}, S)$ is in* Chomsky normal form *(CNF) iff all rewrite rules have the form $N_0 \to \tau$ or $N_0 \to N_1 N_2$, where $N_0, N_1, N_2 \in \mathcal{N}, \tau \in \mathcal{T}$.*

Definition 9 (Derivation Tree). *For a grammar $G = (\mathcal{N}, \mathcal{T}, \mathcal{R}, S)$ in CNF a* derivation tree *is a $\mathcal{N} \cup \mathcal{T}$-labelled tree, where the following conditions hold. The label of the root is S. All leaves and no other nodes are labelled by terminals. Further, for any node labelled by a nonterminal N let $w = \sigma_0 \sigma_1 \dots \sigma_{k-1}$ be the word formed by its k children, then $(N, w) \in \mathcal{R}$.*

3.1 Construction

First we give an intuitive explanation of how a model satisfying the formula we construct in this section looks like. We encode two CFGs G_D and G_U in one formula such that a satisfying model represents two derivation trees, one from each grammar. The representation of a derivation tree for G_D has its root on the top lane and grows *downward*, whereas the tree for G_D has its root on the bottom lane and grows *upward*. In a satisfying model every letter is represented as a number of successive reservations without space in between. Representations of adjacent letters are separated by free space and different letters are represented by a different number of reservations. If a nonterminal from the downwards growing grammar is replaced by a word, the word is represented on the lane below the nonterminal such that the horizontal space used to represent the word is strictly contained in the space used for the nonterminal. Equality of the derived words is represented as horizontal alignment of terminals that differ only in their subscripts (e.g. a_U and a_D in Fig. 1). In Fig. 1 we depict an MLSL model representing the derivation trees of the derivations from Example 1. Note that we never show the view in our visualisations of models.

Example 1. Let $G_D = (\mathcal{N}_D, \mathcal{T}, \mathcal{R}_D, S_D)$ and $G_U = (\mathcal{N}_U, \mathcal{T}, \mathcal{R}_U, S_U)$ be two grammars in CNF, where $\mathcal{N}_D = \{A_D, B_D, S_D\}$, $\mathcal{T} = \{a, b\}$, $\mathcal{N}_U = \{C_U, S_U\}$ and $\mathcal{R}_D, \mathcal{R}_U$ are given in BNF-like notation:

$$S_D \to A_D B_D \qquad S_U \to C_U C_U$$
$$A_D \to A_D A_D \mid a \qquad C_U \to C_U C_U \mid a \mid b$$
$$B_D \to B_D B_D \mid b$$

Two derivations of G_D and G_U are $S_D \to A_D B_D \to A_D A_D B_D \to A_D A_D b \to a A_D b \to aab$ and $S_U \to C_U C_U \to a C_U \to a C_U C_U \to aa C_U \to aab$.

Given two CFGs $G_D = (\mathcal{N}_D, \mathcal{T}, \mathcal{R}_D, S_D)$ and $G_U = (\mathcal{N}_U, \mathcal{T}, \mathcal{R}_U, S_U)$ such that $\mathcal{N}_D \cap \mathcal{N}_U = \emptyset$, we assume two sets $\mathcal{T}_D$, $\mathcal{T}_U$ and bijective functions $\pi_D : \mathcal{T} \to \mathcal{T}_D$ and $\pi_U : \mathcal{T} \to \mathcal{T}_U$ such that $\mathcal{T}_D$, $\mathcal{N}_D$, $\mathcal{T}_U$ and $\mathcal{N}_U$ are all disjoint. The idea behind the two functions is that we want to differentiate between the MLSL encoding of terminals from G_U and from G_D.

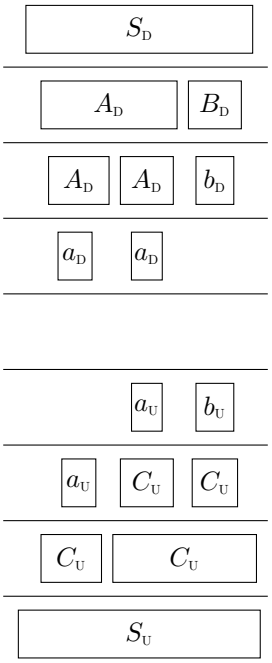

Fig. 1. Visualisation of an MLSL model representing two derivation trees of derivations taken from Example 1. The boxes correspond to letters and reservations are not shown

Letter Let $\mu : \mathcal{T}_D \cup \mathcal{N}_D \cup \mathcal{T}_U \cup \mathcal{N}_U \to \mathbb{N}_{>0}$ be an injective function. In MLSL we represent every letter $\sigma \in \mathcal{T}_D \cup \mathcal{N}_D \cup \mathcal{T}_U \cup \mathcal{N}_U$ as $\mu(\sigma)$ successive reservations from different cars, without free space in between. We formalize this as

$$\begin{aligned} \mathit{letter}(\sigma, c) \stackrel{\text{def}}{=} \quad & \mathit{re}(c) \frown \exists c_1, \dots, c_{\mu(\sigma)-1}. \\ & \mathit{re}(c_1) \frown \dots \frown \mathit{re}(c_{\mu(\sigma)-1}) \wedge \bigwedge_{i,j \in \{1,\dots \mu(\sigma)-1\}}^{i \neq j} c_i \neq c_j \,, \end{aligned}$$

where $c \in \mathit{CVar}$ is a car variable. We use c as an identifier to uniquely differentiate letters within a formula.

Assume that the letter a_D is represented by one reservation, and that the letter b_D is represented by two reservations. We have to distinguish between two occurrences of a_D and one occurrence of b_D. To be able to recognize letters we demand that before and after every letter there is some free space. For this we define

$$\mathit{letter}_{\text{free}}(\sigma, c) \stackrel{\text{def}}{=} \mathit{free} \frown \mathit{letter}(\sigma, c) \frown \mathit{free} \,.$$

The formulas *letter* and $\mathit{letter}_{\text{free}}$ are used in other formulas, which will always bind the car variable, here c.

Start To ensure that there is a starting letter we express that the topmost lane contains the starting nonterminal S_D as

$$\mathit{start}_D \stackrel{\text{def}}{=} \exists c. \begin{matrix} \mathit{letter}_{\text{free}}(S_D, c) \\ \mathit{true} \end{matrix} .$$

Step Now we encode the rewrite relation as a formula. Recall that we consider grammars in Chomsky normal form, so any right-hand side of a rewrite rule has one or two letters. For a word w we define

$$\mathit{word}(w) \stackrel{\text{def}}{=} \begin{cases} \exists c_0. \mathit{letter}_{\text{free}}(\sigma_0, c_0) & \text{if } |w| = 1 \,, \\ \exists c_0, c_1. c_0 \neq c_1 \wedge \mathit{letter}_{\text{free}}(\sigma_0, c_0) \frown \mathit{letter}_{\text{free}}(\sigma_1, c_1) & \text{if } |w| = 2 \,, \end{cases}$$

where σ_j is the j-th letter of w.

To define that a nonterminal N is replaced by a word w, according to the rewrite rule $\mathcal{R}_D$, we define

$$\mathit{step}_D(N, c) \stackrel{\text{def}}{=} \begin{matrix} \mathit{free} \\ \mathit{free} \end{matrix} \frown \left(\begin{matrix} \mathit{letter}(N, c) \\ \bigvee_{w \in \mathcal{R}_D^N} \mathit{word}(w) \end{matrix} \right) \frown \begin{matrix} \mathit{free} \\ \mathit{free} \end{matrix} .$$

This means that we replace a nonterminal on the lane below it with any of the words from the rewrite relation. As we use $\mathit{letter}_{\text{free}}$ in the definition of *word*, we ensure that the replaced letter is horizontally larger than its replacement. Note that we subscripted the formula with D, because we only use it to encode derivation trees growing downward.

As all nonterminals should be replaced on the next lane we define

$$\mathit{step\ all}_D \stackrel{\text{def}}{=} \forall c. \bigwedge_{N \in \mathcal{N}_D} (\langle \mathit{letter}_{\text{free}}(N, c) \rangle \implies \langle \mathit{step}_D(N, c) \rangle) \,.$$

In the premise we test, whether somewhere the car variable c is used as identifier for an occurrence of the nonterminal N. Intuitively, we bind the variable c to the occurrence matched in the premise. For this to work as intended we have to assume that c is used as identifier for only this one occurrence. We formalize this later. In the conclusion we use this c, bounded to one specific occurrence of a nonterminal, to state that below this occurrence there should be a word as defined by the rewrite relation.

Side conditions As already mentioned we do not consider cars with two reservations or overlapping reservations. To exclude these we define

$$\mathit{mutex} \stackrel{\text{def}}{=} \neg\exists c, c'.\, c \neq c' \wedge \langle \mathit{re}(c) \wedge \mathit{re}(c') \rangle\,, \qquad \mathit{no\ 2\ res} \stackrel{\text{def}}{=} \neg\exists c.\, \begin{matrix} \langle \mathit{re}(c) \rangle \\ \langle \mathit{re}(c) \rangle \end{matrix}.$$

We define $\mathit{asm} \stackrel{\text{def}}{=} \mathit{mutex} \wedge \mathit{no\ 2\ res}$.

As we want to encode derivations, all reservations should be part of the representation of the derivation. Thus, all reservations should belong to the representation of some letter, as ensured by

$$\mathit{all\ res\ in\ letter} \stackrel{\text{def}}{=} \forall c.\, \langle \mathit{re}(c) \rangle \implies \exists c'. \bigvee_{\substack{\sigma \in \mathcal{N}_D \cup \mathcal{N}_U \cup \\ \mathcal{T}_D \cup \mathcal{T}_U}} \langle \mathit{letter}_{\text{free}}(\sigma, c') \wedge (\mathit{true} \frown \mathit{re}(c) \frown \mathit{true}) \rangle\,,$$

under the assumption that *asm* holds. We point out that $\mathit{re}(c)$ in the premise and $\mathit{re}(c)$ in the conclusion refer to the same reservation, as every car has only one reservation. Further, $\mathit{letter}_{\text{free}}(\sigma, c')$ is evaluated on the same view as $(\mathit{true} \frown \mathit{re}(c) \frown \mathit{true})$. Thus, this formula ensures that every reservation is inside the representation of a letter.

We want to ensure that all representations of letters are part of a derivation, i.e. we want to forbid orphaned letters. For this we demand that for all letters not on the topmost lane, there is a nonterminal above them. The formula

$$\begin{aligned} \mathit{letter\ next\ to\ letter}_D \stackrel{\text{def}}{=}\ & \forall c. \bigwedge_{\sigma \in \mathcal{N}_D \cup \mathcal{T}_D} \Big(\Big\langle \begin{matrix} \langle \exists c'.\, \mathit{free} \vee \mathit{re}(c') \rangle \\ \mathit{letter}_{\text{free}}(\sigma, c) \end{matrix} \Big\rangle \implies \\ & \exists c''. \bigvee_{N \in \mathcal{N}_D} \Big(\langle \mathit{letter}_{\text{free}}(N, c'') \rangle \wedge \Big\langle \begin{matrix} \mathit{letter}(N, c'') \\ \mathit{letter}_{\text{free}}(\sigma, c) \end{matrix} \Big\rangle \Big) \Big) \end{aligned}$$

ensures this, where we use $\langle \exists c'.\, \mathit{free} \vee \mathit{re}(c') \rangle$ to match at least one lane, without regard for what is on the lane. Similar as in $\mathit{step\ all}_D$ we bind a car variable c to the occurrence of a letter σ in the premise of the implication. However, in the premise we additionally require that the letter is not located on the topmost lane. This is necessary, because we do not want to demand that there is another nonterminal above the starting nonterminal. In the conclusion we bind a new variable c'' to the occurrence of a nonterminal N and require that the representation of N is above and strictly larger than the representation of σ. Intuitively, $\mathit{letter\ next\ to\ letter}_D$ ensures that from any representation of a letter there is a sequence of vertically adjacent representations leading to S_D on the top lane. As $\mathit{step\ all}_D$ requires that all of these sequences obey the rewrite rules, now we can extract derivations from satisfying models.

Second grammar The formulas so far can be used to ensure that the MLSL representation of a derivation from the grammar G_D starts at the top lane and grows downwards. Now we add formulas that demand that a derivation from G_U starts at the bottom and grows upwards.

For all formulas ϕ_D defined so far we create a formula ϕ_U by replacing indices D with U in ϕ and swapping lower and higher formulas in vertical chop operators. For example we define

$$\mathit{start}_U \stackrel{\text{def}}{=} \exists c. \begin{matrix} \mathit{true} \\ \mathit{letter}_{\text{free}}(S_U, c) \end{matrix}, \qquad \mathit{step}_U(N, c) \stackrel{\text{def}}{=} \begin{matrix} \mathit{free} \\ \mathit{free} \end{matrix} \frown \Big(\begin{matrix} \bigvee_{w \in \mathcal{R}_U^N} \mathit{word}(w) \\ \mathit{letter}(N, c) \end{matrix} \Big) \frown \begin{matrix} \mathit{free} \\ \mathit{free} \end{matrix},$$

and the other formulas are similarly defined.

The formula

$$\mathit{free\ lane} \stackrel{\text{def}}{=} \begin{matrix} \mathit{true} \\ \mathit{free} \\ \mathit{true} \end{matrix}$$

requires that there is at least one lane without any reservations. This, together with the formula *letter next to letter*$_D$, ensures that below this free lane there are no representations of letters from G_D. If there was such a letter, then from that letter the sequence of vertically adjacent representations would be interrupted by a free lane. Symmetrically, there is no letter from G_U above this free lane.

We say for two words w, w' that w is a *subsequence* of w' iff we can create w from w' by only removing letters from w'. We now define that the derived word of one grammar is a subsequence of the derived word of the other grammar. For $i \in \{D, U\}$ and $\tau \in \mathcal{T}$ we use τ_i as abbreviation for $\pi_i(\tau)$. Let c, c' be car variables, then we define

$$\phi(\tau, c, c') \stackrel{\text{def}}{=} \left\langle \begin{matrix} \mathit{free} \\ \mathit{true} \\ \mathit{free} \end{matrix} \frown \begin{pmatrix} \mathit{letter}(\tau_D, c) \\ \mathit{true} \\ \mathit{letter}(\tau_U, c') \end{pmatrix} \frown \begin{matrix} \mathit{free} \\ \mathit{true} \\ \mathit{free} \end{matrix} \right\rangle,$$

which requires that the representations of the terminal τ using the variables c, c' are horizontally aligned. The horizontal alignment is enforced by ensuring that the formulas $\mathit{letter}(\pi_D(\tau), c_D)$ and $\mathit{letter}(\pi_U(\tau), c_U)$ are evaluated on the same extension. This is done by evaluating the horizontal chops before the vertical chops. Further, we define

$$\mathit{subseq}_D \stackrel{\text{def}}{=} \bigwedge_{\tau \in \mathcal{T}} \forall c. \left(\langle \mathit{letter}_{\text{free}}(\tau_D, c)\rangle \implies \exists c'. \left(\langle \mathit{letter}_{\text{free}}(\tau_U, c')\rangle \wedge \phi(\tau, c, c')\right)\right),$$

$$\mathit{subseq}_U \stackrel{\text{def}}{=} \bigwedge_{\tau \in \mathcal{T}} \forall c'. \left(\langle \mathit{letter}_{\text{free}}(\tau_U, c')\rangle \implies \exists c. \left(\langle \mathit{letter}_{\text{free}}(\tau_D, c)\rangle \wedge \phi(\tau, c, c')\right)\right).$$

Note that in subseq_D and subseq_U the subformula $\phi(\tau, c, c')$ is the same. However, the car variable names and the subscripts D and U outside $\phi(\tau, c, c')$ are swapped. The formula subseq_D ensures that for every terminal τ, when the downward derivation contains the downward encoding $\pi_D(\tau)$ of τ, then the upwards derivation contains the upward encoding $\pi_U(\tau)$ of τ, horizontally aligned. In other words subseq_D requires that each terminal from the downwards derivation has an horizontally aligned corresponding terminal from the upwards derivation. The horizontal alignment prevents that two downwards terminals share the same corresponding upwards terminal. The explanation for subseq_U is symmetric. Because one word is a subsequence of the other word and vice versa, the two derived words are equal.

For the final formula we conjoin the downward and the upward formulas:

$$F(G_D, G_U) \stackrel{\text{def}}{=} \bigwedge_{i \in \{U,D\}} \mathit{step\ all}_i \wedge \mathit{start}_i \wedge \mathit{letter\ next\ to\ letter}_i \wedge \mathit{subseq}_i \wedge \mathit{free\ lane} \wedge \mathit{all\ res\ in\ letter} \wedge \mathit{asm}.$$

Now we can state our first lemma. We can create an MLSL formula that is satisfiable iff there is a word derivable in two CFGs.

Lemma 1. *Given CFGs $G_D = (\mathcal{N}_D, \mathcal{T}, \mathcal{R}_D, S_D)$ and $G_U = (\mathcal{N}_U, \mathcal{T}, \mathcal{R}_U, S_U)$ we can create a formula $F(G_D, G_U)$ such that*

$$\exists w \in \mathcal{L}(G_D) \cap \mathcal{L}(G_U) \text{ iff } \exists \mathcal{M}. \mathcal{M} \models F(G_D, G_U).$$

Proof. In Appendix A.

As the intersection emptiness of two context-free languages is undecidable [14], we obtain our first main theorem.

Theorem 1. *The satisfiability problem of MLSL is undecidable.*

4 Undecidability of Robust Satisfiability

The undecidability result for MLSL by Hilscher and Linker relies on length measurement [16] and in their construction the authors rely on exact values. Even though in our construction we do not use length measurement, we still rely on correct positional information. For example, we represented letters as non-overlapping, successive reservations without free space in between, i.e. when the position of a single car is shifted a small amount the resulting model does not satisfy the formula.

Here we show that MLSL remains undecidable, even when the position of cars along the lanes are perturbed. We chose positional perturbations as it seems realistic, considering that the position could be obtained via GPS and the lane via more accurate means. We say that a model $\mathcal{M}$ *robustly* satisfies a formula iff there is some $\epsilon > 0$ such that all models differing by at most ϵ (w.r.t. to a *metric*) from $\mathcal{M}$ also satisfy the formula. A formula is robustly satisfiable iff there is a model that robustly satisfies the formula. In the following we adapt our construction from Sect. 3 to this definition of robust satisfiability.

Our definition of robust satisfiability is similar to the definition of tube acceptance from [8], where a robust timed automaton accepts a trajectory iff the automaton also accepts all similar trajectories.

4.1 Robust Satisfiability of MLSL

We define a metric on models of MLSL and robust satisfiability of MLSL formulas w.r.t. to this metric. A metric can be understood to assign to every two models a distance.

Definition 10 (Metric on Models). *Let $\mathcal{M} = (\mathcal{TS}, V, \nu)$, $\mathcal{M}' = (\mathcal{TS}', V', \nu')$ with $\mathcal{TS} = (res, pos, br\text{-}dis)$, $\mathcal{TS}' = (res', pos', br\text{-}dis')$, $V = (L, [r, t])$ and $V = (L', [r', t'])$. Then we define $d(\mathcal{M}, \mathcal{M}') = \infty$ if $L \neq L'$ or $res \neq res'$ or $br\text{-}dis \neq br\text{-}dis'$ and*

$$\begin{aligned} d(\mathcal{M}, \mathcal{M}') = \max_{C \in \mathbb{I}} \{ & |r - r'|, |t - t'|, |pos(C) - pos'(C)|, \\ & |(pos(C) + br\text{-}dis(C)) - (pos'(C) + br\text{-}dis'(C))| \} \end{aligned}$$

otherwise.

Since we only use absolute values, the distance between two models always is positive. Further, it is not difficult to show that the triangle inequality is satisfied. Thus, d is indeed a metric. This means that the distance of two models is infinite, if they disagree on discrete values. Otherwise the distance is the greatest difference of any dense value.

With this we can define robust satisfaction of a formula.

Definition 11 (Robust Satisfaction). *Let $\mathcal{M}$ be a model and let ϕ be an MLSL formula. Then we say that $\mathcal{M}$* robustly satisfies *ϕ, denoted by $\mathcal{M} \models^{\mathrm{R}} \phi$, if and only if*

$$\exists \epsilon \in \mathbb{R}_{>0}. \forall \mathcal{M}'. d(\mathcal{M}, \mathcal{M}') \leq \epsilon \text{ implies } \mathcal{M}' \models \phi \,.$$

A formula is robustly satisfiable *iff there is a model that robustly satisfies it.*

Example 2. Consider the following formulas:

$$\begin{aligned} \phi_0 &\stackrel{\text{def}}{=} c_0 \neq c_1 \wedge \langle re(c_0) \frown re(c_1) \rangle \,, \\ \phi_1 &\stackrel{\text{def}}{=} \phi_0 \wedge \neg \exists c_2, c_3. c_2 \neq c_3 \wedge \langle re(c_2) \wedge re(c_3) \rangle \,. \end{aligned}$$

The formula ϕ_0 requires that there are two successive reservations from different cars without free space in between. The model $\mathcal{M}_0$, depicted in Fig. 2(a), robustly satisfies ϕ_0 because the positions of the cars can be perturbed by a small amount without affecting satisfaction by the model. Hence, ϕ_0 is robustly satisfiable. The model, $\mathcal{M}_1$ (Fig. 2(b)) does not robustly satisfy ϕ_0 because if the

position of c_1 is increased (moved to the right) by an arbitrary small amount there is free space between the reservations, which violates ϕ_0.

The formula ϕ_1 additionally requires that there are no overlapping reservations. Thus, $\mathcal{M}_0$ does not satisfy the formula. While $\mathcal{M}_1$ satisfies ϕ_1, moving a single car in any direction either creates overlapping reservations or free space, both of which violate the formula. In general ϕ_1 requires that the reservation of c_0 ends exactly where the reservation of c_1 starts. Naturally, this is not robustly satisfiable.

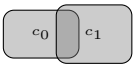

Fig. 2(a). The model $\mathcal{M}_0$, which contains two overlapping reservations

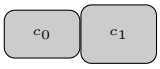

Fig. 2(b). The model $\mathcal{M}_1$, which contains two reservations. The second reservation starts exactly where the first ends

4.2 Construction

We need to replace those parts of our construction, that are affected by small perturbations of positions. For this we adapt our representation of letters, because they are represented as successive reservations starting exactly where another reservation ends. Additionally, we have to adapt the *subseq*-formulas, because they ensured perfect alignment of letters, which is not possible in a setting with perturbations.

Letter To represent letters we remove the assumption that there are no overlapping reservations. However, we still do not consider cars with multiple reservations. For a finite set $\mathcal{C} \subseteq CVar$ of car variables we define

$$\begin{aligned} \mathit{only}(\mathcal{C}) &\stackrel{\text{def}}{=} \bigwedge_{c\in\mathcal{C}} \mathit{re}(c) \wedge (\forall c'.\, (\mathit{true} \frown \mathit{re}(c') \frown \mathit{true}) \implies \bigvee_{c''\in\mathcal{C}} c'' = c')\,, \\ \mathit{only}_{\text{free}}(\mathcal{C}) &\stackrel{\text{def}}{=} \mathit{free} \frown \mathit{only}(\mathcal{C}) \frown \mathit{free}\,, \end{aligned}$$

to ensure that the current view is filled by reservations from all car variables in $\mathcal{C}$, but does not contain reservations from any other cars. See Fig. 3 for a visualisation. Now we can define our representation of letters as

$$\begin{aligned} \mathit{letter}(\sigma, c) &\stackrel{\text{def}}{=} \mathit{startmarker}(c) \frown \exists c_0, \dots, c_{\mu(\sigma)-1}. \bigwedge_{i,j\in\{0,\dots\mu(\sigma)-1\}}^{i\neq j} c_i \neq c_j \wedge \\ &\qquad \mathit{only}_{\text{free}}(\{c_0\}) \frown \dots \frown \mathit{only}_{\text{free}}(c_{\mu(\sigma)-1}) \frown \exists e.\, \mathit{endmarker}(e)\,, \\ \mathit{startmarker}(c) &\stackrel{\text{def}}{=} \mathit{only}(\{c\}) \frown \exists c'.\, c \neq c' \wedge \mathit{only}(\{c, c'\}) \frown \mathit{only}(\{c'\})\,, \\ \mathit{endmarker}(e) &\stackrel{\text{def}}{=} \exists c'.\, e \neq c' \wedge \mathit{only}(\{c'\}) \frown \mathit{only}(\{e, c'\}) \frown \mathit{only}(\{e\})\,, \end{aligned}$$

where σ is either a terminal or a nonterminal and c, e are car variables. Our representation of letters begins and ends with two different markers. In between these markers the representation contains $\mu(\sigma)$ reservations. This representation of letters does not depend on exact positions of cars. In Fig. 3 we depict two adjacent letters with free space in between them.

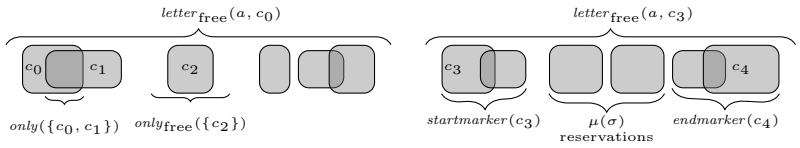

Fig. 3. Visualization of a model satisfying $letter_{\text{free}}(a, c_0) \frown letter_{\text{free}}(a, c_3)$ with $\mu(a) = 2$, where $letter_{\text{free}}$ uses the new *letter*-formula. Reservations are shown as rectangles with rounded corners; different heights are used for better visualisation

Subsequence In Sect. 3 we ensured that terminals at the same position in the derived words are horizontally aligned. With imprecise positions we cannot ensure such an alignment. We define the new subsequence formulas similar to their definition in Sect. 3. For $i \in \{D, U\}$ and $\tau \in \mathcal{T}$ we use τ_i as abbreviation for $\pi_i(\tau)$. Let c, c' be car variables, then we define

$$\psi(\tau, c, c') \stackrel{\text{def}}{=} \left\langle \begin{matrix} free \\ true \frown \\ free \end{matrix} \left(\begin{matrix} letter_{\text{free}}(\tau_D, c) & & letter(\tau_D, c) \\ true & \vee & true \\ letter(\tau_U, c') & & letter_{\text{free}}(\tau_U, c') \end{matrix} \right) \frown \begin{matrix} free \\ true \\ free \end{matrix} \right\rangle,$$

which requires that one representation of τ is horizontally strictly contained within the other representation. Horizontal containment is ensured by aligning *letter* with $letter_{\text{free}}$. The disjunction represents that it does not matter which representation is the larger one. Further, we define

$$subseq_D \stackrel{\text{def}}{=} \bigwedge_{\tau \in \mathcal{T}} \forall c. \left(\langle letter_{\text{free}}(\tau_D, c)\rangle \implies \exists c'. \left(\langle letter_{\text{free}}(\tau_U, c')\rangle \wedge \psi(\tau, c, c')\right)\right),$$
$$subseq_U \stackrel{\text{def}}{=} \bigwedge_{\tau \in \mathcal{T}} \forall c'. \left(\langle letter_{\text{free}}(\tau_U, c')\rangle \implies \exists c. \left(\langle letter_{\text{free}}(\tau_D, c)\rangle \wedge \psi(\tau, c, c')\right)\right).$$

As before, the subformula $\psi(\tau, c, c')$ is the same in $subseq_D$ and $subseq_U$ and we swapped the car variable names and the subscripts D and U outside $\psi(\tau, c, c')$.

All other formulas remain as they are, only that they use the new *letter* formula. The final formula looks almost as before, with the exception that we do not forbid overlapping reservations. This does not pose a problem, because still all reservations are required to be inside a letter and there we exactly specified which reservations we allow. As already mentioned the final formula does not contain *mutex* anymore. We define

$$F_{\text{robust}}(G_D, G_U) \stackrel{\text{def}}{=} \bigwedge_{i \in \{U,D\}} step\ all_i \wedge start_i \wedge letter\ next\ to\ letter_i \wedge subseq_i \wedge$$
$$free\ lane \wedge all\ res\ in\ letter \wedge no\ 2\ res\,.$$

The following lemma reduces the intersection problem of two CFGs to the robust satisfiability problem.

Lemma 2. *Given CFGs $G_D = (\mathcal{N}_D, \mathcal{T}, \mathcal{R}_D, S_D)$ and $G_U = (\mathcal{N}_U, \mathcal{T}, \mathcal{R}_U, S_U)$ we can create a formula $F_{\text{robust}}(G_D, G_U)$ such that*

$$\exists w \in \mathcal{L}(G_D) \cap \mathcal{L}(G_U) \text{ iff } \exists \mathcal{M}. \mathcal{M} \models^{\text{R}} F_{\text{robust}}(G_D, G_U)\,.$$

Proof. In Appendix B.

Thus, we get our second theorem.

Theorem 2. *The robust satisfiability problem of MLSL is undecidable.*

5 Discussion

For our constructions we do not require some of the features of original MLSL, i.e. claims and the ego constant. Hence, we consider a simplified logic without these features. As original MLSL is a conservative extension of our simplified logic, our results also apply to original MLSL. Further, our constructions do not require that a car has multiple reservations. Still we allowed for multiple reservations to not deviate from the original logic too far.

In the following we discuss some possible restrictions on the model side, to perhaps arrive at a decidable logic. Further, we discuss whether the resulting restricted logic remains undecidable.

As both of our constructions rely only on topological arguments, the constructions still can be used to prove undecidability after various restrictions on the model are imposed. However, both of our constructions require that all terminals of a derivation are horizontally contained in the representation of the starting nonterminal of that derivation without overlapping with another letter. As a derivation contains an unbounded amount of terminals, for our constructions to work the maximum size of the representation of the starting nonterminal needs to be unbounded, or the minimum size of terminals needs to be unbounded.

Our construction from Sect. 3 requires that there is no free space in between reservations of the same letter. This implies that the minimum and maximum size of letters are bounded by the minimum and maximum size of reservations. Hence, when we impose bounds on both, by the above argument the construction does not work anymore, i.e. there are context-free grammars for which $F(G_D, G_U)$ is unsatisfiable even though the intersection of the languages is not empty.

Our construction from Sect. 4 does not restrict the size of reservations or the free space in between reservations in any way. Thus, the maximum size of a letter is not bounded by the maximum size of a reservation. To restrict the maximum size of a letter we could bound the maximum size of a view. Then, by the above argument we can not use our construction to answer whether the robust satisfiability problem of the resulting restricted logic is undecidable. Note that undecidability of robust satisfiability implies undecidability of classical satisfiability.

Below we name the mentioned restrictions, and we introduce discreteness of the horizontal dimension as new restriction:

P_0**:** The maximum size of reservations (given by *br-dis*) is bounded.
P_1**:** The maximum size of the view is bounded.
Q_0**:** The minimum size of reservations is bounded.
Q_1**:** The horizontal domain is discrete.

Note that in effect P_1 implies P_0, and that Q_1 implies Q_0. In Table 1 we summarize our arguments from above using the names introduced for our restrictions.

Table 1. The table shows under which combination of restrictions on the model our constructions still serve to prove undecidability of the (robust) satisfiability problem (✓). With X we mark combinations where our constructions do not work. In the table let $R \in \{P_0, P_1, Q_0, Q_1\}$ and $i \in \{0, 1\}$

	R	P_0 and Q_i	P_1 and Q_i
Construction from Sect. 3	✓	X	X
Construction from Sect. 4	✓	✓	X

At last we point out that our constructions strongly depend on the unboundedness of the number of lanes and cars.

6 Conclusion

As our first result we proved undecidability of MLSL without length measurement via a reduction from the emptiness of the intersection of two context free grammars. As our second result we

proved that the logic remains undecidable even when the position along the lane is known only approximately. This proof also works with restrictions of MLSL, e.g. when the extension of the lanes is discrete instead of dense or when the minimal or maximal size of reservations are bounded or any combination of these.

In future work it is worthwhile to create a connection between MLSL and other well studied logics. This may lead to a better understanding of MLSL and increase interest in this logic. Consider for example the modal logic of intervals Halpern-Shoam-logic (HS) [9]. There we have a set of atomic propositions and intervals over a domain, e.g. the real numbers. Every interval satisfies an atomic proposition or its negation. Additionally, we can use for each of Allen's interval relations (before, after, begins etc.) [2] a modal operator that captures the semantics of the relation. In a reduction from HS to MLSL, car reservations might correspond to intervals, (negated) propositions to lanes and the horizontal chop operator to modal operators in HS.

Additionally, we are interested in decidable fragments of MLSL. Possible fragments could be obtained by restricting the nesting of vertical and horizontal chops or imposing an upper bound on the possible number of lanes. Further, it might be interesting to consider simpler modal operators, such as the unary left and right neighbourhood modalities from [4,9] instead of the chop operators.

Acknowledgements I thank Manuel Gieseking, Martin Hilscher, Sven Linker, Ernst-Rüdiger Olderog and Maike Schwammberger for helpful discussions and proofreading. Additionally, I would like to thank the anonymous reviewers of this paper and of a previous version of this paper for valuable feedback.

References

1. Akshay, S., Bérard, B., Bouyer, P., Haar, S., Haddad, S., Jard, C., Lime, D., Markey, N., Reynier, P.A., Sankur, O., Thierry-Mieg, Y.: Overview of Robustness in Timed Systems. Citeseer (2012)
2. Allen, J.F.: Maintaining knowledge about temporal intervals. Commun. ACM 26(11), 832–843 (1983)
3. Alur, R., Feder, T., Henzinger, T.A.: The benefits of relaxing punctuality. J. ACM 43(1), 116–146 (1996)
4. Chaochen, Z., Hansen, M.R.: An adequate first order interval logic. In: Compositionality: the Significant Difference, pp. 584–608. Springer (1998)
5. Chaochen, Z., Hansen, M.R., Sestoft, P.: Decidability and undecidability results for duration calculus. In: Enjalbert, P., Finkel, A., Wagner, K. (eds.) STACS 93, LNCS, vol. 665, pp. 58–68. Springer (1993)
6. Chaochen, Z., Hoare, C.A.R., Ravn, A.P.: A calculus of durations. Inf. Process. Lett. 40(5), 269–276 (1991)
7. Fränzle, M., Hansen, M.R.: A robust interpretation of duration calculus. In: Theoretical Aspects of Computing - ICTAC 2005, pp. 257–271. Springer (2005)
8. Gupta, V., Henzinger, T., Jagadeesan, R.: Robust timed automata. In: Maler, O. (ed.) Hybrid and Real-Time Systems, LNCS, vol. 1201, pp. 331–345. Springer (1997)
9. Halpern, J.Y., Shoham, Y.: A propositional modal logic of time intervals. J. ACM 38(4), 935–962 (1991)
10. Henzinger, T.: The Temporal Specification and Verification of Real-time Systems. Ph.D. thesis, Stanford University (1991)
11. Henzinger, T.A., Raskin, J.F.: Robust undecidability of timed and hybrid systems. In: Hybrid systems: computation and control, pp. 145–159. Springer (2000)
12. Hilscher, M., Linker, S., Olderog, E.R.: Proving safety of traffic manoeuvres on country roads. In: Liu, Z., Woodcock, J., Zhu, H. (eds.) Theories of Programming and Formal Methods. LNCS, vol. 8051, pp. 196–212. Springer (2013)
13. Hilscher, M., Linker, S., Olderog, E.R., Ravn, A.P.: An abstract model for proving safety of multi-lane traffic manoeuvres. In: Qin, S., Qiu, Z. (eds.) ICFEM. LNCS, vol. 6991, pp. 404–419. Springer (2011)
14. Hopcroft, J., Ullman, J.: Introduction to Automata Theory, Languages, and Computation. Addison Wesley (1979)
15. Koymans, R.: Specifying real-time properties with metric temporal logic. Real-Time Syst. 2(4), 255–299 (1990)
16. Linker, S., Hilscher, M.: Proof theory of a multi-lane spatial logic. In: Theoretical Aspects of Computing - ICTAC 2013. pp. 231–248. Springer (2013)

17. Moszkowski, B.: A temporal logic for multi-level reasoning about hardware. IEEE Comput. 18(2), 10–19 (1985)
18. Schäfer, A.: A calculus for shapes in time and space. In: Liu, Z., Araki, K. (eds.) Theoretical Aspects of Computing - ICTAC 2004, LNCS, vol. 3407, pp. 463–477. Springer (2005)
19. Venema, Y.: A modal logic for chopping intervals. J. Log. Comput. 1(4), 453–476 (1991)
20. Woodcock, J., Davies, J.: Using Z – Specification, Refinement, and Proof. Prentice Hall (1996)

A Proof of Lemma 1

For the 'only if'-direction we first extend the nodes of the derivation trees to have intervals representing the parent-child and the sibling relation. This is explained in Lemma 3. With these extended derivation trees we can create a model satisfying $F(G_D, G_U)$.

In the 'if'-direction we create the derivation trees, extended with intervals, by induction on the number of lanes. Then we use the intervals at the nodes of the trees to show equality of the derived words.

We give a more formal definition of derivation trees than in Sect. 3.

Definition 12 (Derivation Tree). *A binary* tree *is a prefix closed set $\Upsilon \subseteq \{0,1\}^*$. A node is defined as the path to the node, which is a word over $\{0,1\}$. A* labelled tree *is a tuple (Υ, f), where $f : \{0,1\}^* \to \Sigma$ is the labelling function and Σ is some set of labels. We lift the labelling function f to ordered sequences of nodes. A* derivation tree *for a grammar $G = (\mathcal{N}, \mathcal{T}, \mathcal{R}, S)$ in CNF is a labelled tree (Υ, f), where $\Upsilon \neq \emptyset$, $f : \{0,1\}^* \to (\mathcal{N} \cup \mathcal{T})$ and*

$$f(\epsilon) = S \wedge \forall x \in \Upsilon.\, (f(x) \in \mathcal{T} \iff x.children = \emptyset) \wedge \\ (f(x) \in \mathcal{N} \implies (f(x), f(x.children)) \in \mathcal{R}),$$

where $x.children$ is the sequence of children of x such that $x.children[v] = xv$ with $v \in \{0,1\}$. For a tree Υ we define $depth(\Upsilon) = \max_{x \in \Upsilon}\{|x|\}$. Let $x, y \in \Upsilon$, then we define the prefix *relation as $x \sqsubseteq y \stackrel{\text{def}}{=} \exists z \in \{0,1\}^*. xz = y$. Additionally, let xy, xz be nodes, then $dis(xy, xz) = |y| + |z|$, i.e. the* distance *of two nodes is the length of the shortest path from one node to the other. For nodes $x0y, x1z$ of a tree we say that $x0y$ is smaller than $x1z$. This defines a total order on nodes.*

The derivations from Example 1 are shown in Fig. 4 as derivation trees.

Fig. 4. Derivation trees for the derivations from Example 1. The set of nodes for the left tree is is $\{\epsilon, 0, 1, 00, 01, 10, 000, 010\}$ and for the right tree it is $\{\epsilon, 0, 1, 00, 10, 11, 100, 110\}$

Definition 13 (Interval). *Let $\mathbb{IR}$ be the set of all closed, non-point intervals over the non-negative real numbers, i.e. $\mathbb{IR} = \{[r,t] \mid r, t \in \mathbb{R} \wedge r < t\}$. For intervals $[r,t], [r',t'] \in \mathbb{IR}$ we define*

$$[r',t'] \subset [r,t] \stackrel{\text{def}}{=} r < r' \wedge t > t',$$
$$\underline{[r,t]} \stackrel{\text{def}}{=} r,$$
$$\overline{[r,t]} \stackrel{\text{def}}{=} t.$$

Because we consider grammars in CNF without rewrite rules that replace a nonterminal by the empty word, we can be sure that two derivation trees of the same word have the same number of leafs. This allows us to prove the following lemma:

Lemma 3. *Let (Υ_D, f_D) and (Υ_U, f_U) be two derivation trees from grammars in CNF. Further, let both trees have the same number $k \in \mathbb{N}_{\geq 1}$ of leafs. We can create two labelling functions $g_D : \Upsilon_D \to \mathbb{IR}$ and $g_U : \Upsilon_U \to \mathbb{IR}$ such that*

$$\underline{g_i(x)} < \overline{g_i(x)}\,, \tag{1}$$
$$g_i(x0) \subset g_i(x)\,, \tag{2}$$
$$g_i(x1) \subset g_i(x)\,, \tag{3}$$
$$\underline{g_i(x1)} > \overline{g_i(x0)}\,, \tag{4}$$
$$g_D(y_j) = g_U(z_j)\,, \tag{5}$$

where $i \in \{D, U\}$, $x \in \Upsilon_i$ and $y_j \in \Upsilon_D, z_j \in \Upsilon_U$ are leafs representing the j-th leaf by their total ordering.

Proof. To construct the functions we start to define them for the leafs first. Let there be $2k$ values $r_0, \dots, r_{k-1}, t_0, \dots, t_{k-1} \in \mathbb{R}$ such that

$$r_0 < t_0 < r_1 < t_1 < \dots < r_{k-1} < t_{k-1} \qquad \text{and} \tag{6}$$
$$r_{j+1} - t_j = 2 \cdot \max\{|\Upsilon_D|, |\Upsilon_U|\} + 1 \qquad \text{and} \tag{7}$$
$$g_D(y_{j'}) = g_U(z_{j'}) = [r_{j'}, t_{j'}]\,, \tag{8}$$

with $j \in [0, k-2]$ and $j' \in [0, k-1]$. This means that the distance of the rightmost point of $g_i(x_j)$ to the leftmost point of $g_i(x_{j+1})$ is defined by the number of nodes in the derivation trees.

Let $x \in \Upsilon_i$ be a node with children, then we define

$$g_i(x) = \begin{cases} [r-1, t+1] & \text{if } |x.\mathit{children}| = 1 \text{ with } g_i(x0) = [r,t] \\ [r-1, t'+1] & \text{else with } g_i(x0) = [r,t] \text{ and } g_i(x1) = [r',t']\,. \end{cases} \tag{9}$$

Essentially, the interval of a parent is the interval of the children increased by one in each direction. This ensures that the parent of a node x is a strict superset of x and a possible sibling of x. Thus, so far it is clear that our construction satisfies Equations (1), (2), (3) and (5). In Fig. 5 a visualisation of g_D and g_U for the derivations from Example 1 is depicted.

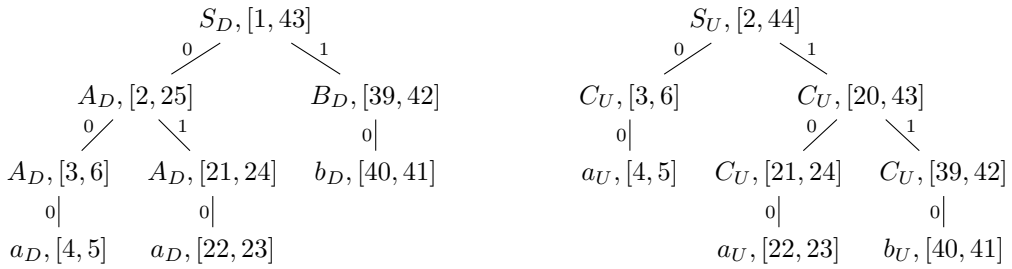

Fig. 5. Derivation trees for the derivations from Example 1. We added interval information as defined in the proof

We say that $x0$ is the left child of x and that $x1$ is the right child of x. We define the leftmost (resp. rightmost) leaf of a node $x \in \Upsilon_i$ as

$$\mathit{lf}_L(x) \stackrel{\text{def}}{=} x_j \in \Upsilon_i \text{ such that } x \sqsubseteq x_j \text{ and } \forall j' < j.\, x_{j'} \not\sqsubseteq x\,,$$
$$\mathit{lf}_R(x) \stackrel{\text{def}}{=} x_j \in \Upsilon_i \text{ such that } x \sqsubseteq x_j \text{ and } \forall j' > j.\, x_{j'} \not\sqsubseteq x\,.$$

We reformulate (9) to show that (4) holds. For any node $x \in \Upsilon_i$ the interval labelling is given by

$$g_i(x) = [\underline{g_i(lf_L(x))} - dis(x, lf_L(x)), \overline{g_i(lf_R(x))} + dis(x, lf_R(x))]\,. \tag{10}$$

This formalizes that for a node x the left border of its interval is given by the left border of the leftmost leaf that is a descendant of x and the distance from that leftmost leaf to x, and similarly for the right border.

For the following equations we point out that the distance of a child from its parent is less or equal the number of nodes in the tree, i.e.

$$\forall \Upsilon \subseteq \{0,1\}^*.\, \forall x, x' \in \Upsilon.\, x \sqsubseteq x' \implies dis(x, x') \leq |\Upsilon|\,. \tag{11}$$

Further, note that if $lf_R(x0) = x_j$ then $lf_L(x1) = x_{j+1}$, i.e. for a node x the rightmost leaf of the left child of x is adjacent to the leftmost leaf of the right child of x. Given

$$\underline{g_i(x1)} > \overline{g_i(x0)} \iff \underline{g_i(x1)} - \overline{g_i(x0)} > 0$$

we show that our construction satisfies (4):

$$\begin{aligned}
& \underline{g_i(x1)} - \overline{g_i(x0)} && \text{with (10)} \\
= \;& \underline{g_i(lf_L(x1))} - dis(x1, lf_L(x1)) - (\overline{g_i(lf_R(x0))} + dis(x0, lf_R(x0))) \\
= \;& \underline{g_i(lf_L(x1))} - \overline{g_i(lf_R(x0))} - dis(x1, lf_L(x1)) - dis(x0, lf_R(x0)) && \text{for some } j \\
= \;& \underline{g_i(x_{j+1})} - \overline{g_i(x_j)} - dis(x1, lf_L(x1)) - dis(x0, lf_R(x0)) && \text{with (7)} \\
= \;& 2 \cdot \max\{|\Upsilon_D|, |\Upsilon_U|\} + 1 - dis(x1, lf_L(x1)) - dis(x0, lf_R(x0)) && \text{with (11)} \\
\geq \;& 2 \cdot \max\{|\Upsilon_D|, |\Upsilon_U|\} + 1 - 2 \cdot \max\{|\Upsilon_D|, |\Upsilon_U|\} \\
> \;& 0
\end{aligned}$$

□

We get the following corollary.

Corollary 1. *Let (Υ, f) be a derivation tree and let $g : \Upsilon \to \mathbb{IR}$ be an interval labelling function as defined in Lemma 3. Then*

$$\forall x, x' \in \Upsilon.\, (x \not\sqsubseteq x' \wedge x' \not\sqsubseteq x) \implies g(x) \cap g(x') = \emptyset\,.$$

Lemma 1. *Given CFGs $G_D = (\mathcal{N}_D, \mathcal{T}, \mathcal{R}_D, S_D)$ and $G_U = (\mathcal{N}_U, \mathcal{T}, \mathcal{R}_U, S_U)$ we can create a formula $F(G_D, G_U)$ such that*

$$\exists w \in \mathcal{L}(G_D) \cap \mathcal{L}(G_U) \text{ iff } \exists \mathcal{M}.\, \mathcal{M} \models F(G_D, G_U)\,.$$

Proof. First we give a definition used in both directions of the proof. We remind that we used $\pi_i : \mathcal{T} \to \mathcal{T}_i$ to distinguish terminals from the downward and the upward derivation. Let $\pi_i' : \mathcal{N}_i \cup \mathcal{T} \to \mathcal{T}_i \cup \mathcal{N}_i$ be an extension of π_i that maps nonterminals to their identity. We define π_i' as

$$\pi_i'(\sigma) = \begin{cases} \pi_i(\sigma) & \text{if } \sigma \in \mathcal{T}\,, \\ \sigma & \text{otherwise}\,. \end{cases}$$

Case 'only if': From $w \in \mathcal{L}(G_D) \cap \mathcal{L}(G_U)$, where G_D, G_U are in CNF, we can create two derivations of w, one according to the grammar G_D and one according to the grammar G_U. Thus, there are derivation trees (Υ_D, f_D) and (Υ_U, f_U). Let the model we create be $\mathcal{TS}, V, \nu$ with $\mathcal{TS} = (res, pos, br\text{-}dis)$ and $V = (L, X)$. The valuation ν does not matter as our formula does not contain free variables.

Let $i \in \{D, U\}$. We create additional labelling functions $g_i : \{0,1\}^* \to \mathbb{IR}$ satisfying the restriction from Lemma 3. We will use g_i to define the extension X of the view and *pos* of the traffic snapshot.

We can define the interval of lanes as $L = [0, \mathit{depth}(\Upsilon_D) + \mathit{depth}(\Upsilon_U) + 2]$. The $+2$ is necessary because the root of a tree does not count towards its depth but still is represented on its own lane. Let $r_{\text{all}} = \min\{g_D(\epsilon) \cup g_U(\epsilon)\}$, $\delta, \delta' \in \mathbb{R}_{>0}$ and $t_{\text{all}} = \max\{g_D(\epsilon) \cup g_U(\epsilon)\}$ then we define the extension of the view as $X = [r - \delta, t + \delta']$. Now we assign to all nodes $x \in \Upsilon_i$ an MLSL representation using reservations. Let $k = \mu(\pi_i'(f_i(x)))$ be the number of cars needed to represent the label of x in MLSL. Further, let $C_0^{i,x}, \dots, C_{k-1}^{i,x}$ be different cars for which we did not assign *res* or *pos* yet and let $s_0, \dots, s_k \in \mathbb{R}$ such that $\underline{g_i(x)} = s_0^{i,x} < s_1^{i,x} < \dots < s_k^{i,x} = \overline{g_i(x)}$. Then we define

$$\begin{aligned}
\mathit{res}(C_j^{i,x}) &= \begin{cases} \{|L| - |x| - 1\} & \text{if } x \in \Upsilon_D \\ \{|x|\} & \text{if } x \in \Upsilon_U \end{cases}, \\
\mathit{pos}(C_j^{i,x}) &= s_j^{i,x}, \\
\mathit{br\text{-}dis}(C_j^{i,x}) &= s_{j+1}^{i,x} - s_j^{i,x},
\end{aligned}$$

for $j \in [0, k-1]$. All cars which we do not use to represent nodes in the trees have reservations outside the view. The idea behind the definition of *res* is that Υ_D is represented in the upper part of the view growing downward, and Υ_U is represented in the lower part of the view growing upward. The number of lanes is chosen large enough so the representations do not interfere with each other.

Remark 1. For some node $y \in \Upsilon_D$ let $\sigma = \pi_D'(f_D(y))$. Then the traffic snapshot $\mathcal{TS}$, the view $(L', g_i(y))$ with $L' = [|L| - |y| - 1, |L| - |y| - 1]$ and the valuation $\nu' = \nu \oplus \{c \mapsto C_0^{D,y}\}$ satisfy the formula $\mathit{letter}(c, \sigma)$, where E is an arbitrary car identifier. Further, from Corollary 1 we can conclude that there are $\delta, \delta' \in \mathbb{R}_{>0}$ such that $\mathcal{TS}, (L', [\underline{g_i(y)} - \delta, \overline{g_i(y)} + \delta']), \nu' \models \mathit{letter}_{\text{free}}(c, \sigma)$. This holds similarly for $z \in \Upsilon_U$.

Now we have to show $\mathcal{TS}, V, \nu \models F(G_D, G_U)$. Clearly, the formula *no 2 res* is satisfied by our construction. For *mutex*, we first observe that none of the reservations used to represent a node overlap each other. Further, note that for all nodes from Υ_i of the same depth level g_i assigns disjoint intervals to those nodes and that representations of nodes from different trees are always located on different lanes. The model satisfies *all res in letter*, because all reservations are part of a sequence of reservations representing a letter and because the functions g_i ensure that different representations are separated by free space. The lane $\mathit{depth}(\Upsilon_U) + 1$ contains no reservation, which ensures satisfaction of *free lane*.

In the following we argue why formulas ϕ_D are satisfied. The arguments apply symmetrically to formulas ϕ_U. The model satisfies start_D, because lane $|L| - 1$ contains the representation of S_D and nothing else.

When a view $V = (L, [r, t])$ is decomposed into subviews $V_{[r,s]}$ and $V_{[s,t]}$ then we say that s is a *chop point*, where we defined $V_{[r,s]} = (L, [r, s])$ (see Definition 2). To see that the model satisfies $\mathit{step\ all}_D$ consider any subview $([n, n], [r, t])$ with $n \in L$ and $[r, t] \subseteq X$ satisfying the premise of $\mathit{step\ all}_D$, i.e. $\mathit{letter}_{\text{free}}(N, c)$ for some $N \in \mathcal{N}_D$ and car variable c. This view contains an MLSL representation of the label of some node $y \in \Upsilon_D$. Assume that y has two children (the case where y has one child works analogously), then let $\sigma_0 = \pi_D'(f_D(y0))$, $\sigma_1 = \pi_D'(f_D(y1))$ and $w = \sigma_0\sigma_1$. We show that the view $([n-1, n], [r, t])$ satisfies $\mathit{step}(N, c)$. Note that we only have to show that $V' = ([n-1, n-1], [r, t])$ satisfies the lower line of $\mathit{step}(N, c)$ because we already know that the premise is satisfied by V. We evaluate $\mathit{word}(w)$ on the view $V'_{g_D(y)}$. Let c_0, c_1 be the quantified car variables in $\mathit{word}(w)$, then we define

$$\begin{aligned}
s &= \frac{(\mathit{pos}(C_{\mu(\sigma_0)-1}^{D,y0}) + \mathit{br\text{-}dis}(C_{\mu(\sigma_0)-1}^{D,y0},)) + \mathit{pos}(C_0^{D,y1})}{2}, \\
\nu' &= \nu \oplus \{c_0 \mapsto C_0^{D,y0}, c_1 \mapsto C_0^{D,y1}\}.
\end{aligned}$$

Note that the first two summands together are the end of the reservation of $C_{\mu(\sigma_0)-1}^{D,y0}$. Thus, s lies in between the representations of σ_0 and σ_1. It is always possible to define such a point because the

horizontal domain is dense. This ensure that both $letter_{\text{free}}(\sigma_0, c_0)$ and $letter_{\text{free}}(\sigma_1, c_1)$ have free space at the borders of their subviews. Now we have $\mathcal{TS}, V'_{\underline{[g_D(y),s]}}, \nu' \models letter_{\text{free}}(\sigma_0, c_0)$, where the $\mu(\sigma_0) + 1$ chop points within $letter_{\text{free}}(\sigma_0, c_0)$ can be assigned to $s_j^{D,y0}$ for $j \in \{0, \ldots, \mu(\sigma_0)\}$. Similarly, $\mathcal{TS}, V'_{\overline{[s,g_D(y)]}}, \nu'$ satisfies $letter_{\text{free}}(\sigma_1, c_1)$. This means that $step(N, c)$ is satisfied and hence, also $step\ all_D$ is satisfied.

With similar arguments we can argue that *letter next to letter*$_D$ is satisfied. At last, we consider $subseq_D$. Let $([n, n], [r, t])$ be a view satisfying the premise of $subseq_D$ for a terminal τ, a car variable c and a valuation which maps c to the car with the first reservation in V.

Equation (5) from Lemma 3 ensures that nodes representing the same position in the derived word are labelled by the same interval. Hence, there is $V' = ([l, l], [r', t'])$ with $l < n$ and $[r', t'] \subset [r, t]$, a car variable c' and a valuation ν' such that $\mathcal{TS}, V', \nu' \models letter(\pi'_U(\tau), c')$. As pointed out in Remark 1 we can extend V' to $V'' = ([l, l], [r'', t''])$ with $[r', t'] \subset [r'', t'']$ such that $\mathcal{TS}, V'', \nu' \models letter_{\text{free}}(\pi'_U(\tau), c')$, which satisfies the first conjunct of the conclusion of $subseq_D$. The second conjunct $\phi(\tau, c, c')$ is satisfied by the traffic snapshot $\mathcal{TS}$, the view $([l, n], [r, t] \cap [r'', t''])$ and the valuation ν'. Hence, $\mathcal{TS}, V, \nu$ satisfies $F(G_D, G_U)$.

Case 'if': In this direction we first reason about the structure of a satisfying model. Then we show that we can create a derivation tree of G_D from the model and a derivation tree of G_U, which works symmetrical to the case for G_D. At the end we show that both trees represent the same word.

We are given $\mathcal{TS}, V_{\text{all}}, \nu \models F(G_D, G_U)$, with $V_{\text{all}} = ([l_{\text{all}}, n_{\text{all}}], [r_{\text{all}}, t_{\text{all}}])$. Note that as all references to car variables are quantified ν does not contain any useful information. The formula *mutex* ensures that V_{all} contains no overlapping reservations and *no 2 res* ensures that every car has at most one reservation inside the view. The formula *all res in letter* ensures that all reservations are part of the representation of a letter, and thus we reason about representations of letters. For $\sigma \in T_D \cup N_D \cup T_U \cup N_U$, $l \in [l_{\text{all}}, n_{\text{all}}]$ and $[r, t] \subset [r_{\text{all}}, t_{\text{all}}]$ we define

$$\begin{aligned} repr(\sigma, l, [r, t]) \stackrel{\text{def}}{=} \mathcal{TS}, ([l, l], [r, t]), \nu \models \exists c.\, letter(\sigma, c) \quad &\text{and} \\ \exists \delta.\, \mathcal{TS}, ([l, l], [r - \delta, r]), \nu \models free \quad &\text{and} \\ \exists \delta.\, \mathcal{TS}, ([l, l], [t, t + \delta]), \nu \models free, \end{aligned}$$

which means that on lane l the interval $[r, t]$ contains $\mu(\sigma)$ successive reservations and some free space to the left and to the right. With $repr_V(\sigma, l, [r, t])$ we refer to the view $([l, l], [r, t])$.

In the following we argue for formulas subscripted with D, however symmetric arguments apply for formulas subscripted with U. The formula $start_D$ ensures that S_D is the only letter with a representation on the topmost lane. Let this representation be given by $repr_V(S_D, n_{\text{all}}, [r_{top}, t_{top}]))$, with $[r_{top}, t_{top}] \subset [r_{\text{all}}, t_{\text{all}}]$.

We show how to construct the derivations, and later we show that they derive the same word. Let $\mathbb{V}_{\text{all}}$ be the set of all subviews of V_{all}, including V_{all}. We define a function $h_D : \mathbb{V}_{\text{all}} \to \{0, 1\}^*$ that assigns every view representing a letter a path over $\{0, 1\}^*$. Then Υ_D is the image of h_D. We start with

$$h_D(repr_V(S_D, n_{\text{all}}, [r_{top}, t_{top}])) = \epsilon\,.$$

Let $\sigma \in \mathcal{T} \cup \mathcal{N}_D$, $l \in [l_{\text{all}}, n_{\text{all}} - 1]$, $[r, t] \in \mathbb{IR}$ such that $repr(\pi'_D(\sigma), l, [r, t])$ holds. Then from *letter next to letter*$_D$ we know that there is a nonterminal $N \in \mathcal{N}_D$ and an interval $[r', t']$ with $[r, t] \subset [r', t']$ such that $repr(\pi'_D(N), l + 1, [r', t'])$ and we define

$$h_D(repr_V(\pi'_D(\sigma), l, [r, t])) = h_D(repr_V(\pi'_D(N), l + 1, [r', t'])) \mid v\,,$$

where $\mid$ is the operator for string concatenation and

$$v = \begin{cases} 1 & \text{if } \exists \sigma'' \in \mathcal{T} \cup \mathcal{N}_D, [r'', t''] \subset [r', t'] \text{ such that } t'' < r \\ & \quad \text{and } repr(\pi'_D(\sigma''), l, [r'', t''])\,, \\ 0 & \text{otherwise}\,. \end{cases}$$

This means that if there is another representation on the same lane left of $repr_V(\pi'_D(\sigma), l, [r,t])$ and that other representation also is enclosed by the same representation on the lane above, then $repr_V(\pi'_D(\sigma), l, [r,t])$ becomes child one, otherwise child zero. Note that *step all*$_D$ ensures that there is at most one such other representation.

Let the image of h_D be Υ_D. Additionally, we define the labelling functions $f_D\colon \Upsilon_D \to \mathcal{T} \cup \mathcal{N}_D$, which assigns to every node a letter and $g_D\colon \Upsilon_D \to \mathbb{IR}$, which assigns to every node an interval. If $h_D(repr(\pi'_D(\sigma), l, [r,t])) = x$, we define $f_D(x) = \sigma$ and $g_D(x) = [r,t]$.

Let $x \in \Upsilon_D$ and assume that $f_D(x) \in \mathcal{N}_D$, then *step all*$_D$ and that the grammars used to create $F(G_D, G_U)$ are in CNF ensure either

- there is $x0 \in \Upsilon_D$ with $f_D(x0) \in \mathcal{T}$ and $(f_D(x), f_D(x0)) \in \mathcal{R}_D$, or
- there are $x0, x1 \in \Upsilon_D$ with $f_D(x0), f_D(x1) \in \mathcal{N}$ and $(f_D(x), f_D(x0)f_D(x1)) \in \mathcal{R}_D$.

Further, if $f_D(x) \in \mathcal{T}$ the formula *letter next to letter*$_D$ ensures that $x0, x1 \notin \Upsilon_D$. Thus, (Υ_D, f_D) is a derivation tree of G_D. In a symmetric manner we can build functions h_U, f_U, g_U and a tree Υ_U.

Let $i \in \{D, U\}$ and let $\bar{i} \in \{D, U\} \setminus \{i\}$. From *letter next to letter*$_i$ and *step all*$_i$ we know that g_i satisfies (1)–(4) from Lemma 3. Thus, g_i defines a total order on the leafs of Υ_i. Let $\tau \in \mathcal{T}$, then from *subseq*$_i$ we know that for every node $x \in \Upsilon_i$ with $f_i(x) = \tau$ and $g_i(x) = [r,t]$ there is a node $x' \in \Upsilon_{\bar{i}}$ such that $f_{\bar{i}}(x') = \tau$ and $g_{\bar{i}}(x') = [r,t]$. Hence, *subseq*$_i$ ensures that the word derived by (Υ_i, f_i) is a subsequence of the word derived by $(\Upsilon_{\bar{i}}, f_{\bar{i}})$, and vice versa for $\bar{i}$. Thus, the two derived words are equal.

□

B Proof of Lemma 2

For the 'only if'-direction let $\sigma \in \mathcal{T}_D \cup \mathcal{N}_D \cup \mathcal{T}_U \cup \mathcal{N}_U$ and let c be a car variable. First we prove as a lemma that for any model satisfying $letter_{\text{free}}(\sigma, c)$ the positions of all cars in the view can be perturbed by a nonzero amount such that the perturbed model still satisfies the formula. Then we show that this is also true for $F_{\text{robust}}(G_D, G_U)$. At last we create a model that satisfies $F_{\text{robust}}(G_D, G_U)$ such that the view and also all cars outside the view can be perturbed such that the perturbed model still satisfies the formula. This proves the 'only if'-direction.

For the 'if'-direction we refer to the proof of Lemma 1 as it remains mostly unchanged.

Let M be a set and let d be a metric on M. Then (M, d) is a *metric space*.

Definition 14 (Open Set). *Given a metric space (M, d) and a set $M' \subseteq M$, then M' is* open *if and only if*

$$\forall x \in M'. \exists \epsilon \in \mathbb{R}_{>0}. \forall y \in M. d(x, y) \leq \epsilon \text{ implies } y \in M' .$$

This means that a set is open iff for every element the set also contains the elements neighbourhood, where the metric is used to determine the elements in the neighbourhood.

For a traffic snapshot $\mathcal{TS}$ and a view $V = (L, X)$ we define the set of cars visible in the view as

$$I_{\mathcal{TS},V} = \{C \mid C \in \mathbb{I} \wedge X \cap se(C) \neq \emptyset \wedge L \cap res(C) \neq \emptyset\} .$$

We introduce a new metric on models. The difference to the metric d from Sect. 4 is that the distance of two models is infinite when their extensions or the cars visible in the view differ. Further, the new metric does not measure perturbations of cars outside the view.

Definition 15. *Let $\mathcal{M} = (\mathcal{TS}, V, \nu)$, $\mathcal{M}' = (\mathcal{TS}', V', \nu')$ be two models with $\mathcal{TS} = (res, pos, br\text{-}dis)$, $\mathcal{TS}' = (res', pos', br\text{-}dis')$. Then we define $d'(\mathcal{M}, \mathcal{M}') = \infty$ if $V \neq V'$ or $\exists C \in I_{\mathcal{TS},V}. (res(C) \cap L) \neq (res'(C) \cap L)$ and*

$$d'(\mathcal{M}, \mathcal{M}') = \max_{C \in I_{\mathcal{TS},V}} \{|pos(C) - pos'(C)|, \\ |(pos(C) + br\text{-}dis(C)) - (pos'(C) + br\text{-}dis'(C))|\}$$

otherwise.

For an MLSL formula ϕ let $sat(\phi)$ be the set of MLSL models satisfying ϕ. We prove that for the metric d' the set $sat(letter_{\text{free}}(\sigma, c))$ is open, where $\sigma \in \mathcal{T}_i \cup \mathcal{N}_i$ with $i \in \{U, D\}$ and $c \in CVar$. Note that this is not the case for the metric d from Sect. 4, because MLSL only restricts cars inside the view. Hence, e.g. exactly where the view ends a satisfying model could have a reservation. Then, if the view is perturbed a little the reservation would be inside the view and $letter_{\text{free}}(\sigma, c)$ would not be satisfied as it requires that the view only contains reservations to represent σ and forbids other reservations.

Lemma 4. *Let $\sigma \in \mathcal{T}_i \cup \mathcal{N}_i$ with $i \in \{U, D\}$ and $c \in CVar$. Then the set $sat(letter_{\text{free}}(\sigma, c))$ is open w.r.t. the metric d'.*

Proof. We give arithmetic constraints on MLSL models that are satisfied by a model $\mathcal{M}$ iff $\mathcal{M} \in sat(letter_{\text{free}}(\sigma, c))$. Then we show that the set of models satisfying these constraints is open w.r.t. the metric d'. This implies that $sat(letter_{\text{free}}(\sigma, c))$ is open.

For two nonempty intervals $[r, t]$, $[r', t']$ we write $[r, t] < [r', t']$ to denote $t < r'$. Let $\mathcal{M} = (\mathcal{TS}, V, \nu)$ with $V = ([l, n], [r, t])$. Assume $\mathcal{M} \in sat(letter_{\text{free}}(\sigma, c))$ then

$$l = n \wedge |I_{\mathcal{TS},V}| = \mu(\sigma) + 4 \tag{12}$$

because $letter_{\text{free}}(\sigma, c)$ ensures that the view V contains a single lane with $\mu(\sigma)$ cars to represent σ and four cars to represent the start-and endmarker. Let $k = \mu(\sigma) + 4$ and let us assume w.l.o.g.

that the cars in $I_{\mathcal{TS},V} = \{C_0, \dots, C_{k-1}\}$ are ordered by their positions. Then we know

$$\nu(c) = C_0\,, \tag{13}$$
$$r < pos(C_0) < pos(C_1) < pos(C_0) + br\text{-}dis(C_0) < pos(C_1) + br\text{-}dis(C_1)\,, \tag{14}$$
$$se(C_1) < se(C_2) < \dots < se(C_{k-3}) < se(C_{k-2}) \text{ and} \tag{15}$$
$$pos(C_{k-2}) < pos(C_{k-1}) < pos(C_{k-2}) + br\text{-}dis(C_{k-2}) < pos(C_{k-1}) + br\text{-}dis(C_{k-1}) < t\,, \tag{16}$$

because $letter_{\text{free}}(\sigma, c)$ requires that

- the first reservation belongs to c (Equation (13)),
- the extension of the view starts with free space, followed by *startmarker* (Inequality (14)),
- *startmarker* is followed by free space, then $\mu(\sigma)$ reservations each separated by free space (Inequality (15)) and
- followed by *endmarker* and then the extension ends with free space (Inequality (16)).

Thus, $\mathcal{M} \in sat(letter_{\text{free}}(\sigma, c))$ implies that $\mathcal{M}$ satisfies (12)–(16).

Now we show the other direction. As in the proof of Lemma 1, when a view $V = (L, [r, t])$ is decomposed into subviews $V_{[r,s]}$ and $V_{[s,t]}$ then we say that s is a *chop point*, where we defined $V_{[r,s]} = (L, [r, s])$ (see Definition 2). Given a model $\mathcal{TS}, V, \nu$ with $V = (L, [r, t])$ satisfying (12)–(16) we show that $\mathcal{TS}, V, \nu \models letter_{\text{free}}(\sigma, c)$. For this the main difficulty is to choose chop points s.t. the subviews satisfy the subformulas. We choose the first few chop points as follows (see also Fig. 6):

$$s_0 = pos(C_0) \qquad s_1 = pos(C_1)$$
$$s_2 = pos(C_0) + br\text{-}dis(C_0) \qquad s_3 = pos(C_1) + br\text{-}dis(C_1)$$
$$s_4 = pos(C_2) \qquad s_5 = pos(C_2) + br\text{-}dis(C_2)$$
$$s_6 = \frac{pos(C_2) + br\text{-}dis(C_2) + pos(C_3)}{2} \qquad s_7 = pos(C_3)$$
$$s_8 = pos(C_3) + br\text{-}dis(C_3)$$

Let $\nu(c) = C_0$ and $\nu(c_j) = C_j$ for $1 \leq j \leq \mu(\sigma) - 1$. The model $\mathcal{TS}, V_{[r,s_0]}, \nu$ satisfies the first *free* in $letter_{\text{free}}(\sigma, c)$. The subformula $startmarker(c)$ in $letter_{\text{free}}(\sigma, c)$ is satisfied by $\mathcal{TS}, V_{[s_0,s_3]}, \nu$, where

$$\mathcal{TS}, V_{[s_0,s_1]}, \nu \models only(\{c\})\,,$$
$$\mathcal{TS}, V_{[s_1,s_2]}, \nu \models only(\{c, c_1\})\,,$$
$$\mathcal{TS}, V_{[s_2,s_3]}, \nu \models only(\{c_1\})\,.$$

Further, the model $\mathcal{TS}, V_{[s_3,s_6]}, \nu$ satisfies the first $only_{\text{free}}(\{c_2\})$. Note that we choose s_6 to be in between $pos(C_2) + br\text{-}dis(C_2)$ and $pos(C_3)$ to ensure that the view the first $only_{\text{free}}$ and the view the second $only_{\text{free}}$ is evaluated on, both have free space left and right of their respective reservation. This is always possible as the horizontal domain is dense. The remaining chop points can be chosen analogously.

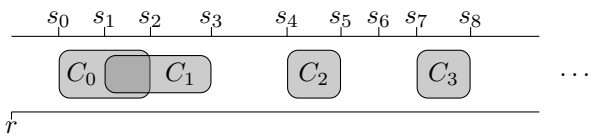

Fig. 6. Visualisation of how chop points can be chosen when evaluating $letter_{\text{free}}(\sigma, c)$ on a model satisfying (12)–(16)

Thus, $\mathcal{M} \in sat(letter_{\text{free}}(\sigma, c))$ iff $\mathcal{M}$ satisfies (12)–(16). The set of MLSL models satisfying (12)–(16) form an open set w.r.t. the metric d' because the positional information of cars is constrained only be strict relations. Hence, the set $sat(letter_{\text{free}}(\sigma, c))$ is open as well. □

A similar argumentation can be used for each conjunct of $F_{\text{robust}}(G_D, G_U)$. Thus, for each conjunct ϕ of $F_{\text{robust}}(G_D, G_U)$ the set $sat(\phi)$ is open w.r.t. the metric d'. From topology theory we know that the intersection of finitely many open sets is an open set. Hence, we get the following corollary.

Corollary 2. *The set $sat(F_{\text{robust}}(G_D, G_U))$ is open w.r.t. the metric d'.*

As in the proof of Lemma 1, we remind that we used $\pi_i : \mathcal{T} \to \mathcal{T}_i$ to distinguish terminals from the downward and the upward derivation. Let $\pi_i' : \mathcal{N}_i \cup \mathcal{T} \to \mathcal{T}_i \cup \mathcal{N}_i$ be an extension of π_i that maps nonterminals to their identity. We define π_i' as

$$\pi_i'(\sigma) = \begin{cases} \pi_i(\sigma) & \text{if } \sigma \in \mathcal{T}\,, \\ \sigma & \text{otherwise}\,. \end{cases}$$

We can now prove Lemma 2.

Lemma 2. *Given CFGs $G_D = (\mathcal{N}_D, \mathcal{T}, \mathcal{R}_D, S_D)$ and $G_U = (\mathcal{N}_U, \mathcal{T}, \mathcal{R}_U, S_U)$ we can create a formula $F_{\text{robust}}(G_D, G_U)$ such that*

$$\exists w \in \mathcal{L}(G_D) \cap \mathcal{L}(G_U) \;\text{ iff }\; \exists \mathcal{M}.\, \mathcal{M} \models^{\text{R}} F_{\text{robust}}(G_D, G_U)\,.$$

Proof. Case 'only if': In this proof we use trees and intervals as they were defined in Appendix A. From $w \in \mathcal{L}(G_D) \cap \mathcal{L}(G_U)$ we know that there are derivation trees (Υ_D, f_D) and (Υ_U, f_U), both deriving w. To create the model and prove that it robustly satisfies $F_{\text{robust}}(G_D, G_U)$ we use an adaptation of Lemma 3, where (5), i.e. $g_D(y_j) = g_U(z_j)$ is replaced by

$$g_D(y_j) \subset g_U(z_j) \vee g_U(z_j) \subset g_D(y_j)\,, \tag{17}$$

where $y_j \in \Upsilon_D, z_j \in \Upsilon_U$ are leafs representing the j-th leaf by their total ordering. The other statements in the lemma remain unchanged. Similarly, the proof of this adapted lemma is mostly the same and we do not give it here. Let g_D, g_U be labelling functions satisfying this adapted lemma.

We create $\mathcal{TS}, V, \nu$ such that $\mathcal{TS}, V, \nu \models^{\text{R}} F_{\text{robust}}(G_D, G_U)$. The definition of $V = (L, X)$ and ν remains the same as in the proof of Lemma 1. Thus, we define $L = [0, \mathit{depth}(\Upsilon_D) + \mathit{depth}(\Upsilon_U) + 2]$, $r = \min\{g_D(\epsilon) \cup g_U(\epsilon)\}$, $t = \max\{g_D(\epsilon) \cup g_U(\epsilon)\}$ and $X = [r - \delta, t + \delta']$ for any $\delta, \delta' \in \mathbb{R}_{>0}$.

We adapt the definitions of $\mathit{pos}, \mathit{br\text{-}dis}, \mathit{res}$. Let $i \in \{D, U\}$, $x \in \Upsilon_i$, $k = \mu(\pi_i'(f_i(x)))$ and let $C_0, \ldots, C_{k-1}$ be different cars for which we did not assign *res* or *pos* yet. Furthermore, let $s_0, t_0, \ldots, s_{k-1}, t_{k-1} \in \mathbb{R}$ such that $\underline{g_i(x)} < s_0 < t_0 < s_1 < \cdots < s_{k-1} < t_{k-1} < \overline{g_i(x)}$. Then we define

$$\begin{aligned} \mathit{res}(C_j) &= \begin{cases} \{|L| - |x| - 1\} & \text{if } x \in \Upsilon_D \\ \{|x|\} & \text{if } x \in \Upsilon_U\,, \end{cases} \\ \mathit{pos}(C_j) &= s_j\,, \\ \mathit{br\text{-}dis}(C_j) &= t_j\,, \end{aligned}$$

for $j \in [0, k-1]$. Additionally, we need reservations for the start- and endmarkers. Let $C_0', \ldots, C_3'$ be four different cars, for which we did not assign *res* or *pos* yet. Let $r_0, \ldots, r_7 \in \mathbb{R}$ such that $\underline{g_i(x)} = r_0 < r_1 < r_2 < r_3 < s_0$ and $t_{k-1} < r_4 < r_5 < r_6 < r_7 = \overline{g_i(x)}$. Then we define:

$$\begin{aligned} \mathit{pos}(C_0') &= r_0 & \qquad \mathit{pos}(C_2') &= r_4 \\ \mathit{br\text{-}dis}(C_0') &= r_2 & \qquad \mathit{br\text{-}dis}(C_2') &= r_6 \\ \mathit{pos}(C_1') &= r_1 & \qquad \mathit{pos}(C_3') &= r_5 \\ \mathit{br\text{-}dis}(C_1') &= r_3 & \qquad \mathit{br\text{-}dis}(C_3') &= r_7 \end{aligned}$$

This means that we assign values which satisfy the *startmarker* and *endmarker* formulas. The lanes of reservations of these cars are (as before)

$$res(C'_j) = \begin{cases} \{|L| - |x| - 1\} & \text{if } x \in \Upsilon_D \\ \{|x|\} & \text{if } x \in \Upsilon_U \,. \end{cases}$$

Any car C which we do not use to represent nodes in the trees have reservations outside the view such that $\overline{X} < pos(C)$.

The model $\mathcal{M}$ we created satisfies $F_{\text{robust}}(G_D, G_U)$ (the proof is similar to the proof given for Lemma 1) and thus $\mathcal{M}$ is in $sat(F_{\text{robust}}(G_D, G_U))$. By Corollary 2 positions and breaking distances of cars inside the view can be perturbed by some $\epsilon_0 > 0$ such that $\mathcal{M}_0 \in sat(F_{\text{robust}}(G_D, G_U))$.

Lemma 2 additionally requires that the extension of the view and the data of cars outside the view can be perturbed. Note that changes to positional data of cars outside the view can not affect satisfaction of an MLSL formula, as long as the cars remain outside the view and that with MLSL only qualitative properties can be expressed. Furthermore, note that for all models $\mathcal{M}'$ in $sat(F_{\text{robust}}(G_D, G_U))$ and all cars C we have either

$$r' < pos'(C) \wedge pos'(C) + br\text{-}dis'(C) < t' \;\text{ or }\; t' < pos'(C)\,. \tag{18}$$

Thus, we can change $\mathcal{M}_0$ further by perturbing the extension of the view and data of cars outside the view by some $\epsilon_1 > 0$ as long as (18) is satisfied and the set of visible cars remains unchanged without affecting satisfaction of $F_{\text{robust}}(G_D, G_U)$. Let this further perturbed model be $\mathcal{M}_1$ and let $\epsilon_2 = \min\{\epsilon_0, \epsilon_1\}$. Now $\mathcal{M}_1$ is an arbitrary model with $d(\mathcal{M}, \mathcal{M}_1) \leq \epsilon_2$ and $\mathcal{M}_1 \in sat(F_{\text{robust}}(G_D, G_U))$. Hence, $\mathcal{M}$ robustly satisfies $F_{\text{robust}}(G_D, G_U)$.

Case 'if': The other direction is mostly the same as in the proof of Lemma 1. □