Unifying Control and Verification of Cyber-Physical Systems (UnCoVerCPS)

WP5 Realization of Cyber-Physical Systems

D5.2 – Report on Conformance Testing of Application Models

WP5	**D5.2 – Report on Conformance Testing of Application Models**
Authors	Marianne Hartung - GE Daniel Hess - DLR Ray Lattarulo - Tecnalia Jens Oehlerking - Bosch Joshué Pérez - Tecnalia Alexander Rausch - Bosch
Short Description	The following document summarizes the conformance testing approaches developed in the UnCoVerCPS project and the results of conformance testing the automated driving use case and the wind turbine use case. Furthermore, we demonstrate the application of the UnCoVerCPS methodology based on a robotics use case.
Deliverable Type	Report
Dissemination level	Public
Delivery Date	June 2017
Contributions by	Matthias Althoff (TUM), Christian Heinzemann (Bosch), Stefan Liu (TUM), Ingo Lütkebohle (Bosch), Hendrik Röhm (Bosch), Bastian Schürmann (TUM)
Internal review by	Axel Busboom (GE), Xavier Fornari (ET), Daniel Hess (DLR), Matthias Althoff (TUM)
External review by	
Internally accepted by	
Date of acceptance	

Document history:

Version	**Date**	**Author/Reviewer**	**Description**
1.0	06/12/17	Alexander Rausch	Final version

Contents

1 Introduction & Motivation

The UnCoVerCPS verification methods, and formal verification methods in general, rely on models of the system under consideration, which are then (formally) checked against a specification. Therefore, all proofs that are obtained with the help of these methods are only relative to the model that was used. In order to ensure that properties of the models transfer to the real system, *conformance* of the model to the system needs to be ensured.
Generally, conformance can come in two flavors:

- model-to-model conformance: This notion of conformance relates two models of the same system (typically on different abstraction levels) to each other, allowing transference of properties from one to the other. This type of conformance can be shown either with the help of formal methods or with the help of testing/simulation. By showing, for example, the conformance of an abstract model to a more concrete one, a hierarchy of models can be built, with different level of detail. Note that we present a formal model-to-model trace conformance approach in deliverable D1.2, which we evaluated for a time-discretized and time-continuous system model.

- model-to-system conformance: This notion of conformance relates a model of the system to the system itself. Obviously, the system can only be characterized by means of observation, through measurements. Here, conformance means that these observations can be explained by the model. In the remainder of this deliverable we focus on model-to-system conformance.

In order to obtain a conformant model of a system, generally two steps are required:

- defining a model structure that is suitable for the system, and

- determining the model parameters that result in the "best" conformance.

What is considered "best" is highly dependent on the specific use case. Since verification models typically contain non-determinism or stochasticity, the parameter identification problem for these models is more complex than for deterministic models. In the scope of this deliverable, we concentrate on conformance testing for non-deterministic models. Typically, the goal of non-deterministic modeling for verification is to encompass all (w.r.t. some reasonable assumptions) behaviors of the system, instead of just providing an approximation. Therefore, parameters that need to be estimated, include ranges of uncertainties capturing the non-determinism.

In general, tighter bounds on uncertainties of the model are better for verification, as long as disturbances are seen as purely adversarial (which is the case in our applications).

Therefore, a "good" conformant model for a given structure of disturbances is one which contains as little uncertainty as possible, while still being able to explain all observed behavior. Non-deterministic error terms capturing the uncertainties can be added to a deterministic model in many different ways – for instance, deviations in the position of a vehicle can be explained by perturbations on velocities or accelerations, and there is no clear answer which is the better choice. This will depend on how the model is to be used, including the properties that should be verified.

Consequently, there is no unique "best" model, but a Pareto front of minimal conformant models with different uncertainty structures. While exploring this Pareto front is a challenge, one of the primary challenges in conformance testing is building suitable initial models that potentially can be conformant to the target system. Here, our main consideration is the quality of the reachset over-approximations obtained by different models.

Another important consideration is the notion of conformance that is used. Depending on the properties one wants to transfer, these notions can be weaker or stronger. This will be detailed in Section 3, where we propose a weaker conformance notion that is tailored purely towards safety properties. The fact that the notion is weaker (i.e., more permissive on what is considered conformant) means that it is easier to show on real measurements. On the other hand, this weaker notion is still strong enough so that safety properties are preserved.

The structure of this deliverable is as follows. First, in Section 2, we will give an overview of related work in conformance testing. Sec. 3 introduces the conformance notions and conformance testing procedures used within the scope of UnCoVerCPS. We then will present the results for the DLR vehicle use case (Sec. 4), the GE wind turbine use case (Sec. 6) and a robotics use case from Bosch (Sec. 7).

2 Related Work

The usage of the term *conformance testing* in a formal sense can be traced back to work by Tretmans [38] in the early 1990s. In general, conformance testing in its broadest sense also includes input generation in order to test conformant models. Since already the construction of abstract but conformant models for formal verification is very challenging if real world measurment data is involved, we focused our effort on the process of deriving conformant abstract models of a system and show the applicability of our approach to different use cases.

In [38], Tretmans defined *input-output conformance (ioco)* for discrete transition systems. This definition essentially requires a behavioral inclusion between two systems S_1 and S_2, such that, for all possible input sequences to S_2, all output traces of S_1 can also be observed in S_2. Dang [16, 15], as well as van Osch [39, 40], defined similar conformance notions for hybrid systems, which were inspired by the work of Tretmans. Following Dang [15], we use the term *trace conformance* for this relation throughout the document. While Dang defines trace conformance for hybrid automata, van Osch follows Tretmans more closely and uses hybrid labeled transition systems as the system modeling formalism. He calls the conformance relation *hybrid input-output conformance (hioco)*. There exist several other names for basically the same notion of conformance. Alur et. al. [8] define *language inclusion* for hierarchical hybrid systems, Henzinger et. al. [19, 9] *refinements* for hierarchical hybrid systems. Lynch et. al. [25] introduce *implementation relation* for hybrid input output automata, whereas Tabuada [37] uses the name *behavioral inclusion*.

In general, the question of test selection is usually highly application specific. Some common patterns include structural model coverage [13], assertion- and code coverage [10, 11] or randomized exploration via rapidly-exploring random trees (RRTs) [16, 15].

In contrast to the concept of trace conformance, within the scope of the project we define another conformance relation called *reachset conformance* [34]. Under the assumption that we are interested in showing safety properties only, it is sufficient to relax the conformance property such that for all possible input sequences, all reachable states of S_1 are also reachable in S_2. This a weaker condition than trace conformance, as different sets of traces can lead to the same reachable set, and therefore easier to show on real measurements.

In the context of this deliverable, we employ both trace and reachset conformance, which are formally defined in the following.

3 Conformance Relations & Conformance Testing

In the following section we will introduce the notation used in the remainder of this document. Furthermore, we will formally define trace conformance and reachset conformance as well as the approaches that allow us to build conformant abstract models of a system.

3.1 Preliminaries & Definitions

A dynamic system with uncertainties is defined by its state vector x, a deterministic input u, a non-deterministic disturbance $w \in \mathcal{W}$, the non-linear differential equation f, the measurement vector y, a non-deterministic measurement error $\nu \in \mathcal{V}$, and the non-linear measurement function h:

$$x(t) \in \mathbb{R}^n \qquad \text{state vector signal} \tag{1}$$

$$u(t) \in \mathbb{R}^m \qquad \text{input vector signal} \tag{2}$$

$$y(t) \in \mathbb{R}^o \qquad \text{measurement vector signal} \tag{3}$$

$$w(t) \in \mathcal{W} \subset \mathbb{R}^q \qquad \text{disturbance vector signal} \tag{4}$$

$$\nu(t) \in \mathcal{V} \subset \mathbb{R}^p \qquad \text{meas. error vect. signal} \tag{5}$$

$$\dot{x}(t) = f(x(t), u(t), w(t)) \qquad \text{differential equation} \tag{6}$$

$$\dot{x}(t) \in \{f\left(x(t), u(t), w(t)\right) | w(t) \in \mathcal{W}\} \qquad \text{differential inclusion} \tag{7}$$

$$y(t) \in \{h\left(x(t), u(t), \nu(t)\right) | \nu(t) \in \mathcal{V}\} \qquad \text{uncertain meas.} \tag{8}$$

In the following, we use the notation $w(\cdot) \in \mathcal{W}$, which is a shorthand for $w(t) \in \mathcal{W}, \forall t \in [0, t_f]$, where $t_f \in \mathbb{R}_0^+$ is the final time. We denote the solution of (6) with initial state $x(0)$, input $u(\cdot)$, and disturbance $w(\cdot)$ at time t as $\xi(x(0), u(\cdot), w(\cdot), t)$. The solution satisfies the following two properties:

$$\xi(x(0), u(\cdot), w(\cdot), 0) = x(0) \tag{9}$$

$$\dot{\xi}(x(0), u(\cdot), w(\cdot), t) = f\big(\xi(x(0), u(\cdot), w(\cdot), t), u(t), w(t)\big), \quad \forall t \in \mathbb{R}_0^+. \tag{10}$$

Sometimes, when we consider the undisturbed system, we use $x^*(t) = \xi(x(0), u^*(\cdot), 0, t)$ to denote the nominal solution without disturbances. The set of states, which are reachable at a point of time t by the non-deterministic system f, under the condition that the initial state is in a given initial set, $x(0) \in \mathcal{X}_0$ and that a control input $u(\cdot)$ is applied, is denoted by $\mathcal{R}^e(t)$. The reachable set is defined using the flow ξ:

$$\mathcal{R}^e(t) := \{\xi(x(0), u(\cdot), w(\cdot), t) | x(0) \in \mathcal{X}_0, w(\cdot) \in \mathcal{W}\} \tag{11}$$

The superscript e denotes the exact reachable set, which usually cannot be computed. Rather, an over-approximation of the true reachable set is denoted $\mathcal{R}(t)$, with $R^e(t) \subseteq \mathcal{R}(t)$. For an implicit sampling of the time, $t \in \{t_0, t_1, \cdots, t_f\}$, we denote the according trace of reachable sets with $\mathcal{R} = \{\mathcal{R}(t_0), \mathcal{R}(t_1), \cdots, \mathcal{R}(t_f)\}$ and the algorithm, which computes the trace up to a point of time t_f by $\mathcal{R} = \textsc{Reach}(\mathcal{X}_0, u(\cdot), t_f)$. In Section 4 and Section 7 we use the CORA toolbox [6] for an implementation of Reach.

Let us assume the system is sampled at points of time $T \in \mathbb{R}^K$. The matrix containing the states of the system at these points of time is denoted by $X \in \mathbb{R}^{n \times K}$. The true evolution of the system state is usually not known, but X can also denote a guess of the system's state evolution. The measurement trace $Y \in \mathbb{R}^{o \times K}$ is recorded during one experiment with the investigated system. Typically, the input trace $U \in \mathbb{R}^{m \times K}$ is also recorded.

$$T = [t_1, ..., t_K] \in \mathbb{R}^K \quad \text{sampled points of time} \tag{12}$$

$$X_i = [x(t_1), ..., x(t_K)] \in \mathbb{R}^{n \times K} \quad \text{state trace} \tag{13}$$

$$U_i = [u(t_1), ..., u(t_K)] \in \mathbb{R}^{m \times K} \quad \text{input trace} \tag{14}$$

$$Y_i = [y(t_1), ..., y(t_K)] \in \mathbb{R}^{o \times K} \quad \text{measurement trace} \tag{15}$$

$$V_i = [v(t_1), ..., v(t_K)] \in \mathbb{R}^{p \times K} \quad \text{measurement error trace} \tag{16}$$

$$W_i = [w(t_1), ..., w(t_K)] \in \mathbb{R}^{q \times K} \quad \text{disturbance trace} \tag{17}$$

3.2 Trace Conformance

One of the difficulties of applying formal methods to cyber-physical systems is the transference of formal properties to the real, physical sub-system. In order to make plausible why results derived for the model also apply to the physical system, the conformance between a physical system and a model is investigated in this section. A model is said to be conformant to a system, if it reacts similarly to the system, when the same inputs are applied. *Testing* conformance refers to applying exemplary inputs to the system, recording observations of the system's behavior and investigating whether the model can reproduce similar observations under these inputs.

We use the following definition of trace conformance: A test-case $\langle U_i, Y_i, T_i \rangle$ is understood as a combination of a control input trace $U_i = [u_i(t_1), ..., u_i(t_K)]$ applied to the system and a measurement trace $Y_i = [y_i(t_1), ..., y_i(t_K)] \in \mathbb{R}^{o \times K}$ recorded from the system at discrete points of time $T_i = [t_{i,1}, ..., t_{i,K}] \in \mathbb{R}^K$. A test suite is defined as a set of test cases

$\{\langle U_1, Y_1\rangle, ..., \langle U_r, Y_r\rangle\}$. A model is defined as the combination $\langle f, h, \mathcal{V}, \mathcal{W}\rangle$.

$$C_i := \langle U_i, Y_i, T_i\rangle \quad \text{test case} \tag{18}$$

$$S := \{\langle U_1, Y_1, T_1\rangle, ..., \langle U_r, Y_r, T_r\rangle\} \quad \text{test suite} \tag{19}$$

$$M := \langle f, h, \mathcal{V}, \mathcal{W}\rangle \quad \text{model} \tag{20}$$

$$\langle X_i, V_i, W_i\rangle \quad \text{model trace} \tag{21}$$

Definition: A model $\langle f, h, \mathcal{V}, \mathcal{W}\rangle$ is said to be trace conformant w.r.t. a test suite S, if for every test case $\langle U_i, Y_i, T_i\rangle$ in a test suite S, a model trace $\langle X_i, V_i, W_i\rangle$ with $X_i = [x(t_1), ..., x(t_K)] \in \mathbb{R}^{n\times K}$, $V_i = [v(t_1), ..., v(t_K)] \in \mathbb{R}^{p\times K}$, $W_i = [w(t_1), ..., w(t_K)] \in \mathbb{R}^{q\times K}$, exists, which is consistent with the differential equation f, the measurement function h and the error sets $\mathcal{V}$ and $\mathcal{W}$:

$$\text{TRACECONF}(M, S) \Leftrightarrow \forall i \in \{1, \dots, r\} : \exists\langle X_i, V_i, W_i\rangle : \forall k = \{1, \dots, K-1\} :$$

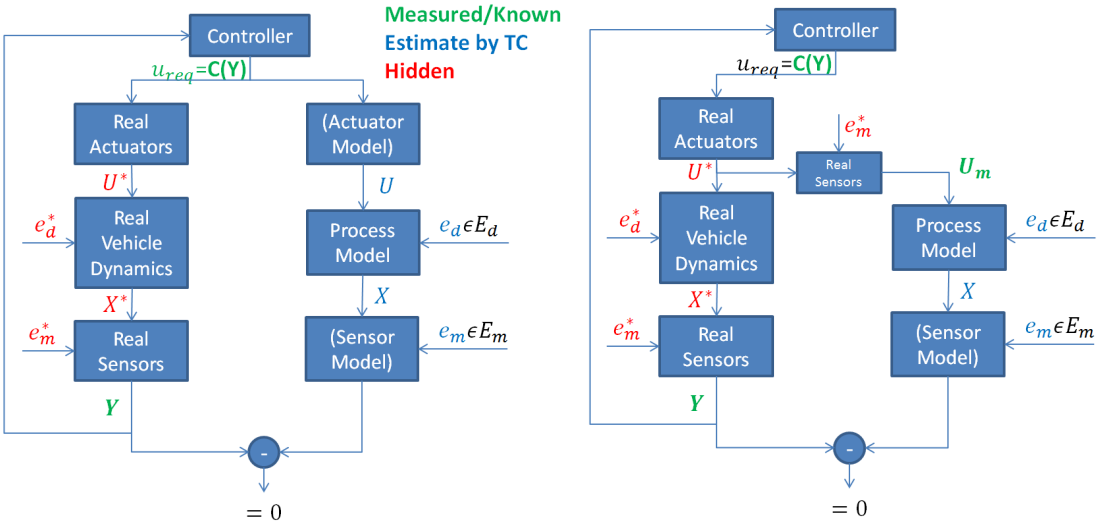
$$x(t_{k+1}) = x(t_k) + \int_{t_k}^{t_{k+1}} f\left(x(\tau), u(t_k), w(t_k)\right) d\tau \tag{22}$$

$$\wedge \quad y(t_k) = h(x(t_k), u(t_k), \nu(t_k)) \tag{23}$$

$$\wedge \quad \nu(t_k) \in \mathcal{V} \wedge w(t_k) \in \mathcal{W} \tag{24}$$

Our conformance testing process involves recording a test-suite with the system and then solving a constraint satisfaction problem for equations (22)-(24) for each test case. If a valid model trace exists for each test case, the model is trace conformant. If the constraint satisfaction problem cannot be solved for one or more test-cases, the model is not conformant. In this case, the system has to be modeled more precisely by choosing a more appropriate f or h, or the uncertainty in the model has to be increased by changing $\mathcal{V}$ and $\mathcal{W}$.

We evaluate two different versions of trace conformance on the system, as depicted in Figure 1: On the left in Figure 1(a) the full trace conformance (TC) relationship is evaluated. A set of measurement errors and disturbances is computed, which leads to equality between measured and simulated trace under consideration of an actuator model, a process model and a sensor model. Whereas on the right in Figure 1(b), the actuator model is skipped by recording the output of the actuators and by supplying this to the simulated process model directly. This second version is a simplification, which on the one hand allows to distinguish between actuator and process errors. On the other hand of course, the simplified version cannot guarantee to identify all errors which might appear between controller and output. Therefore, the final goal is to be conformant according to the approach described in Figure 1(a).



(a) Full TC

(b) TC without actuator model

Figure 1: Two different versions of trace conformance (TC)

We use the following general approach, Alg. 1, to determine trace conformance: Step 2 computes an appropriate disturbance trace W_i and an initial state $x_{i,0}$ for the given test case. Two different solutions for implementation of step 2 are given in Section 3.2.1 and 3.2.2. Step 3 uses the disturbance trace W_i and the initial state $x_{0,i}$ to solve the initial value problem for the system model f, given system input U_i. The result is a state trace X_i, which satisfies (22). Step 4 computes a measurement error trace V_i, which satisfies (23) for the computed X_i at each point of time. This is usually simple, as the solutions for different points of time are independent.

Using this approach, it is possible to guarantee that models are never wrongly classified as conformant if a contradictory trace Y_i exists, (e.g. preventing false positives). In step 3, a numerically sound reachability analysis could be applied to conservatively overapproximate any uncertainties resulting from the numerically imprecise solution of a non-linear differential

Algorithm 1 Test trace conformance between a model $M = \langle f, h, \mathcal{V}, \mathcal{W} \rangle$ and a trace $C_i = \langle U_i, Y_i, T_i \rangle$. Returns *true* in case of conformance.

1: **function** $isConformant \leftarrow$ TraceConformance(M, C_i)
2: $\langle x_{0,i}, W_i \rangle \leftarrow$ DisturbanceTrace(U_i, Y_i, T_i)
3: $X_i \leftarrow$ SolveInitialValueProblem $(f, U_i, W_i, T_i, x_{0,i})$
4: $V_i \leftarrow$ SolveOrdinaryEquations (h, X_i, U_i, Y_i)
5: return *true* if $V_i \in \mathcal{V}^K \wedge W_i \in \mathcal{W}^K$ else return *false*.
6: **end function**

equation. At the same time, step 2 does not have to meet any precision requirements to prevent false positive classification. This allows to employ faster, approximate methods, which can be based on linearization of the non-linear model. We have developed several ideas how step 2 DisturbanceTrace can compute a possible disturbance trace for a given measurement trace. The first, very basic idea is to compute disturbances which make the model state follow the measurement trace exactly. Under the premise that the measurement function h is invertible for x and that the measurement error and the model discrepancy are very small, good results can be achieved in a computationally efficient manner. The idea is detailed in Section 3.2.1. When the assumption of small measurement errors is not applicable, the first method tends to hugely overestimate the magnitude of the disturbances. Therefore a second approach is introduced in Section 3.2.2, which estimates measurement errors and disturbances at the same time.

3.2.1 Trace Conformance w/o Measurement Error

Given a small or no measurement error ($\mathcal{V} \approx \emptyset$) and assuming full observability of system states, the definition of trace conformance for system states in equations (22ff) can be reduced such that a trace conformant model has to satisfy

$$y(t_{k+1}) = \underbrace{y(t_k) + \int_{t_k}^{t_{k+1}} f\left(x(\tau), u(t_k), w(t_k)\right) d\tau}_{\tilde{x}(t_{k+1}):=} . \tag{25}$$

Solving equation (25) for all k can be implemented by solving the optimization problem

$$\forall k : w(t_k) = \arg\min_{w(t_k)} \left\| y(t_{k+1}) - \tilde{x}(t_{k+1}) \right\|_\infty \tag{26a}$$

$$= \arg\min_{w(t_k)} \left(\max_i \left| [y(t_{k+1})]_i - [\tilde{x}(t_{k+1})]_i \right|_2 \right) \tag{26b}$$

with $[y(t_{k+1})]_i$ being the $i-$th component of the reference data vector $y(t_{k+1})$ and $[\tilde{x}(t_{k+1})]_i$ similar the i-th component of $\tilde{x}(t_{k+1})$. In practice, we achieved better performance of the optimization-based approach by changing the objective function such that

$$\forall k : w(t_k) = \arg\min_{w(t_k)} \left(\sum_i (([y(t_{k+1})]_i - [\tilde{x}(t_{k+1})]_i)/[y(t_{k+1})]_i)^2 \right) . \tag{27}$$

To solve the problem stated by equation (27) we used MATLAB's non-linear programming solver *fminunc* (version 2016b). Since the problem stated in (27) solely relies on pairs of neighboring measurement points $y(t_{k+1})$ and $y(t_k)$, and thus follows the "single instruction multiple data" (SIMD) paradigm, solving (27) for all k can be done in parallel. We leverage this advantage in the trace conformance testing of the windturbine model in Section 6 to

archive a low overall runtime of the approach for large collections of reference data. While numerical inaccuracies may arise due to numerical integration of integrals and the objective and step tolerances in the optimization procedure, these can often be neglected with a close to zero measurement error $\mathcal{V}$.

3.2.2 Trace Conformance w/ Measurement Error

The previous section describes a method to compute an initial state and a disturbance trace by exactly following the measurement trace. While this constrains the solution at each discrete point of time and thus leads to small, independent equations for each time step, the disturbance magnitude tends to be overestimated. In order to take measurement errors into account, the disturbance can be chosen in such a way that the model reaches any state which satisfies the measurement equation for one admissible measurement error. Therefore at time t_k, the state $x(t_k)$ must be in a set:

$$x(t_k) \in \{\chi \in \mathbb{R}^n | \exists \nu \in \mathcal{V} : h(\chi, u(t_k), \nu) = y(t_k)\} \tag{28}$$

The ensuing problem is essentially equivalent to a non-linear optimal control problem with state and actuator constraints, in which our disturbances $w(t_k)$ take the place of the usual control inputs and in which our actual (recorded) control inputs $u(t_k)$ constitute pre-determined, time-varying disturbances. As was noted above, there are no formal correctness requirements for the step of determining a disturbance trace W_i and an initial state $x_{0,i}$. Therefore it is sufficient to approximate the solution to the non-linear optimal control problem. A local linearization scheme with a step-wise refinement is chosen, as described in the following algorithm 2. As a model, a non-linear differential equation $\dot{x} = f(x, u, w)$ and a measurement function with additive measurement errors $y = h(x, u) + \nu$ are assumed. In order to simplify satisfaction of measurement error bounds, we assume that the linearization of f is r-step controllable [24] with respect to the disturbance input w at all relevant state space locations, so that $r \cdot q \geq n$. We subdivide W by r times per time step, resulting in $W \in \mathbb{R}^{r \cdot q \times K}$.

The given algorithm starts in step 2 by initializing X with an initial guess X_0 and W with a zero matrix. If the complete state vector can be measured, as in the case of the DLR test vehicle, the initial guess can be $X_0 = Y$. If the state vector is measured only partially, but is fully observable, an initial guess can be attained by employing an observer. Of course, by assigning $W = 0$, the state and disturbance traces X and W can be incongruent at the outset, yet this is resolved by the subsequent iterations of the algorithm. In line 3 a loop is entered, which compares the current linearization error e with a constant bound ϵ, terminating only if e is small enough. Each iteration of the loop, the model dynamics are linearized in line 4

Algorithm 2 Approximates a minimal disturbance trace $W \in \mathcal{W}^K$, which keeps the state trace X inside the bounds defined by h, Y, U and $\mathcal{V}$.

1: **function** $\langle X, W \rangle \leftarrow$ DisturbanceTrace-OptCtrl(X_0, U, Y, T)
2: Initialize $X \leftarrow X_0; W \leftarrow 0; e \leftarrow \infty$
3: **while** $e > \epsilon$ **do**
4: $\langle M, J, j, A, b \rangle \leftarrow$ LocalLinearization$(X, W; f, h, U)$
5: $\Delta X^* \leftarrow \underset{\Delta X \in \mathbb{R}^{n \times K}}{\arg\min} \left\{ \Delta X^{\mathrm{T}} J \Delta X + j \Delta X \quad \text{subj. to} \quad A \Delta X + b \leq 0 \right\}$
6: Update traces: $\tilde{X} \leftarrow X + \Delta X^*; \quad W \leftarrow W + M \Delta X^*$
7: $X \leftarrow$ SolveInitialValueProblem$(f, U, W, T, \tilde{X}_1)$
8: $e \leftarrow |X - \tilde{X}|_\infty$
9: **end while**
10: **end function**

at the current estimate of the state and disturbance trace: A matrix $M \in \mathbb{R}^{r \cdot q \cdot K \times n \cdot K}$ and a vector $m \in \mathbb{R}^{r \cdot q \cdot K}$ are computed, which convert from a change of states, $\Delta X \in \mathbb{R}^{n \times K}$, to a necessary change of disturbance $\Delta W = M \Delta X$ for a linear approximation of the model. The matrices $J \in \mathbb{R}^{K \cdot n \times n \cdot K}, j \in \mathbb{R}^{1 \times n \cdot K}, A \in \mathbb{R}^{(n \cdot K + r \cdot q \cdot K) \times n \cdot K}$ and $b \in \mathbb{R}^{(n \cdot K + q \cdot K) \times 1}$ derive from M and define a constrained, quadratic optimization problem. The matrices and vectors M to b are defined in the following. In line 5 a quadratic optimization problem is solved over the optimization variable ΔX, which corresponds to the change of X in the current iteration and which can be linearly transformed into the corresponding change of the disturbance input ΔW. In order to follow an updated state trace $X + \Delta X$, the disturbances $W + \Delta W = W + M \Delta X$ have to be applied to the system. In line 6, the intermediate state trace $\tilde{X}$ is defined as $\tilde{X} = X + \Delta X^*$ and the disturbance trace is updated to $W + M \Delta X^*$. In order to reduce linearization errors, the intermediate state trace $\tilde{X}$ is used here only for comparison with a more precise update of X. The more precise update to X is gained by solving an initial value problem (IVP). The IVP is defined by the new initial state $\tilde{X}_1$ and the updated disturbance trace W. The IVP can be solved at a finer resolution than the optimization problem and therefore results in a better estimate of the state trace X, which is produced by application of the disturbances W. As a last step in line 8, the error is set to the difference between the initial value solution X and the intermediate solution $\tilde{X}$: If the difference is small enough, convergence is assumed and the latest X and W are returned. Otherwise, the next iteration starts by updating the linearization at the new traces X and W and continues with a new optimization at the updated linearization point.

The local linearizations at X and W are computed for each time-step $1 \leq k \leq K$:

$$[A_k, B_k] := \left.\frac{\partial f}{\partial [x, w]}\right|_{x(t_k), u(t_k), w(t_k)}, \quad C_k := \left.\frac{\partial h}{\partial x}\right|_{x(t_k), u(t_k), \nu(t_k)} \tag{29}$$

For many models, the number of disturbances q is smaller than the number of states n. In that case we subdivide one time step from t_k to t_{k+1} into $r = \lceil n/q \rceil$ equal parts. To steer the model from a state x_k to the exact state x_{k+1}, the r disturbances $w_{k,1}, \cdots, w_{k,r}$ are applied on the subdivided time interval $[t_k, t_{k,1}], [t_{k,1}, t_{k,2}], \cdots, [t_{k,r-1}, t_{k,r}]$, with $t_{k,0} = t_k$, $t_{k,1} = t_k + (t_{k+1} - t_k) \cdot 1/r$, $t_{k,2} = t_k + (t_{k+1} - t_k) \cdot 2/r, \cdots, t_{k,r} = t_{k+1}$. For a time-step with $\Delta t = (t_{k+1} - t_k)/r$, the discrete time system matrices $A_{k,d}$, $B_{k,d}$ are:

$$\begin{bmatrix} A_{k,d} & B_{k,d} \\ 0 & I \end{bmatrix} = \exp\left\{ \begin{bmatrix} A_k & B_k \\ 0 & 0 \end{bmatrix} \Delta t \right\} \tag{30}$$

After using the discrete time system matrices in (30) for the r subintervals,

$$\begin{aligned} &\tilde{A}_k \cdot \Delta x_k + \tilde{B}_k \cdot \Delta w_{k,1:r} := \\ &(A_{k,d})^r \cdot \Delta x_k + \left[(A_{k,d})^{r-1} B_{k,d}, \cdots (A_{k,d})^0 B_{k,d} \right] \cdot \begin{bmatrix} \Delta w_{k,1} \\ \vdots \\ \Delta w_{k,r} \end{bmatrix}, \end{aligned} \tag{31}$$

the approximate solution on the time interval $[t_k, t_{k+1}]$ can be written as:

$$\begin{aligned} x_{k+1} + \Delta x_{k+1} \approx \quad & \xi(\cdots \xi(\xi(x_k, u_k, w_{k,1}, \Delta t), u_k, w_{k,2}, \Delta t) \cdots, u_k, w_{k,r}, \Delta t) \\ & + \tilde{A}_k \cdot \Delta x_k + \tilde{B}_k \cdot \Delta w_{k,1:r}. \end{aligned} \tag{32}$$

We assume that the (pseudo) inverse of each $\tilde{B}_k$ exists, as we require the system to be r-step controllable. This allows to formulate a matrix M over all time-steps which translates from the state change ΔX to the disturbance input change ΔW:

$$\Delta w_{k,1:r} = \begin{bmatrix} \tilde{B}_k^{-1} \tilde{A}_k & \tilde{B}_k^{-1} \end{bmatrix} \begin{bmatrix} \Delta x_k \\ \Delta x_{k+1} \end{bmatrix} \tag{33}$$

$$\Delta W = \begin{bmatrix} \tilde{B}_1^{-1} \tilde{A}_1 & \tilde{B}_1^{-1} & 0 & \cdots & 0 \\ 0 & \tilde{B}_2^{-1} \tilde{A}_2 & \tilde{B}_2^{-1} & 0 & 0 \\ 0 & 0 & \ddots & \ddots & 0 \\ 0 & 0 & 0 & \tilde{B}_{K-1}^{-1} \tilde{A}_{K-1} & \tilde{B}_{K-1}^{-1} \end{bmatrix} \Delta X \tag{34}$$

$$=: M \cdot \Delta X \tag{35}$$

We define a quadratic objective function over the updated disturbances $W + \Delta W$, which incites the trace conformance algorithm to prefer smaller average disturbances. The cost

function is defined with the help of a weighting matrix $S \in \mathbb{R}^{r \cdot q \cdot K \times r \cdot q \cdot K}$, which accounts for the different units of the different disturbances.

$$g(\Delta W) := (W + \Delta W)^{\mathrm{T}} S (W + \Delta W) \tag{36}$$

Setting $J := M^{\mathrm{T}} S M$ and $j := 2W^{\mathrm{T}} S$, we gain a cost function in relation to the variable ΔX, $g = \Delta X^{\mathrm{T}} J \Delta X + j \Delta X$. After linearizing the original constraint $\left|h^{-1}(X, U, Y)\right| < \nu_{max}$ and applying $\Delta W = M \Delta X$ to the original constraints $|(W + \Delta W)| \leq w_{max}$, $\left|h^{-1}(X, U, Y)\right| < \nu_{max}$ one has the following linear inequalities:

$$M\Delta X + W - w_{\max} \leq 0 \tag{37}$$

$$-M\Delta X - W + w_{\max} \leq 0 \tag{38}$$

$$C\Delta X + h(X, U, 0) - Y - \nu_{\max} \leq 0 \tag{39}$$

$$-C\Delta X - h(X, U, 0) + Y + \nu_{\max} \leq 0. \tag{40}$$

The linear inequalities are expressed as $A\Delta X + b < 0$ by defining:

$$A := \begin{bmatrix} M \\ -M \\ C \\ -C \end{bmatrix}, b := \begin{bmatrix} W - w_{\max} \\ -W + w_{\max} \\ h(X, U, 0) - Y - \nu_{\max} \\ -h(X, U, 0) + Y + \nu_{\max} \end{bmatrix} \tag{41}$$

The given approach has been implemented[1] for the ConfTest toolbox and is subsequently applied to and evaluated for the automated vehicle use case.

In contrast to this section, which is based on deriving individual state and error traces for each measurement trace Y_i, the following section takes a combined approach, using reachable sets and a conformance notion based theorem.

3.3 Reachset Conformance

The term *reachset conformance* was first coined in [35] as a weaker conformance notion compared to trace conformance. Here, weaker means that every trace conformant model is also reachset conformant, but not the other way around. This means that reachset conformance in general holds for more models than trace conformance, which has two distinct advantages: a) for a given model, reachset conformance wrt. a measurement is easier to show and b) it is often possible to derive reachset conformant models with less uncertainty than it would be possible under trace conformance. The drawback of reachset conformance is that only the

[1] `https://svn.dlr.de/UnCoVerCPS/T5.3/MATLAB/ConformanceTesting/trace_conformance_optctrl_version03.m`

transference of *safety properties* from a model to the actual system can be shown. In [35], reachset conformance between two systems is defined. In the scope of this deliverable, we only need a special case: conformance of a model with measurements.

A measurement trace Y is reachset conformant with an abstract model of a system if and only if

$$\forall t_i \in T : Y(t_i) \in \{h(\tilde{x}, u(t_i), v(t_i)) | \tilde{x} \in \textsc{Reach}(\mathcal{X}_0, u(\cdot), t_i)\}. \tag{42}$$

If the measurement error v is an additive disturbance on x and independent of u, i.e. $h(x, u, v) = x + v$, then the reachset conformance condition reduces to

$$\forall t_i \in T : Y(t_i) \in \textsc{Reach}(\mathcal{X}_0, u(\cdot), t_i) \oplus \mathcal{V}. \tag{43}$$

This means that it is sufficient to enlarge the reachset by the measurement error and check inclusion of the measured values in that set.
More details on reachset conformance and its testing procedure can be found in [35].

4 Use Case DLR Vehicle

In order to validate the presented conformance testing approach in Sec. 3.2.1, an exemplary test suite has been recorded by the DLR test vehicle. Different vehicle models and their according sets of measurement and disturbance errors have been identified for the vehicle and validated against the recorded test suites. The test setup is depicted in figure 2: The test vehicle (FASCar II) is a Volkswagen Passat TDI from 2009, which is equipped with a differential GPS receiver (DGPS). As it is often the case for conformance testing, an error-free ground truth for the system state is not available. To assess the quality of the position measurements of the DGPS, two maneuvers at lower speed have been additionally recorded with a tracking tachymeter.

Two measurement campaigns that are useful for conformance testing have been recorded on 2016/04/19 at Heinrich-Der-Löwe-Kaserne[2] and on 2016/12/07 at airport Edemissen Edesse[3]. On 2016/04/19 two swerve maneuvers [4] [5] were executed at 10 m/s and 5m/s (see Figure 16) and Figure 17 in Appendix A). The 5 m/s test drive was additionally recorded with the tracking tachymeter, which is unfortunately only available for speeds below 6 m/s. On 2016/12/07 four different maneuvers (lane-change (1), swerve (2), double-lane-change (3), slalom (4)) were recorded at 10 m/s and for two different lateral accelerations [6] [7], $2m/s^2$ (A) and $4m/s^2$ (B), (cf. Appendix A Figure 18-21 for A1-A4 and Figure 22-25 for B1-B4). Each maneuver was executed and recorded at least five times, with the times in the figure caption allowing to identify the corresponding log-files.

An overview of the vehicle-internal measurement and controller setup is given in Figure 4. Our central measurement instrument is the Novatel SPAN inertial navigation system (INS), which uses MEMS to measure accelerations and turn rates and fuses these measurements with DGPS position measurements of two antennas on the vehicle roof. These measurement values are sent at 100Hz via CAN 0 to a data logger. Additional vehicle state measurements are recorded such as individual wheel speeds, motor speed, gear and measured steering angle, which are distributed via CAN 1 at different rates. All maneuvers used for conformance testing have been executed in automated driving mode, i.e., closed loop tracking of a predefined reference trajectory, which is sent via a Dispatcher module from a PC to a closed-loop tracking

[2] 52.249325, 10.575351 `https://goo.gl/maps/DEM8U2gkKXt`

[3] 52.402632, 10.229746 `https://goo.gl/maps/udWhDQHgBN82`

[4] `https://svn.dlr.de/UnCoVerCPS/T5.3/MATLAB/trackingTest_201604/19/A`

[5] `https://svn.dlr.de/UnCoVerCPS/T5.3/MATLAB/trackingTest_201604/19/B`

[6] `https://svn.dlr.de/UnCoVerCPS/T5.3/MATLAB/trackingTest_20161207/A`

[7] `https://svn.dlr.de/UnCoVerCPS/T5.3/MATLAB/trackingTest_20161207/C`

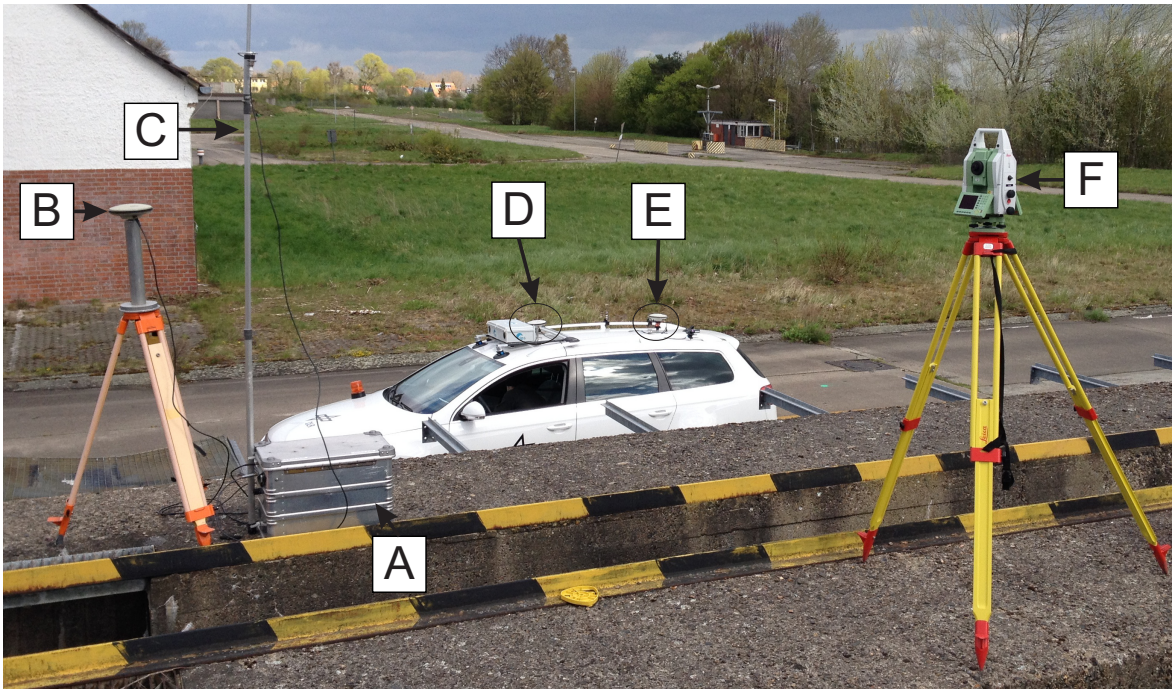


Figure 2: FASCar2 and setup for conformance testing: [A] DGPS base station with receiver and transmitter for correction data, [B] DGPS antenna at known reference position, [C] antenna for transmission of correction data, [D] secondary GPS antenna on vehicle, [E] primary GPS antenna plus marker position for tachymeter, directly above vehicle reference COR, [F] tracking tachymeter.

Figure 3: Final setup on 2016/04/19 for execution of testdrives: Tracking tachymeter placed directly behind vehicle for better visibility of marker on vehicle. FASCar2 facing down reference track (north-to-south) at Heinrich-Der-Löwe-Kaserne, Braunschweig.

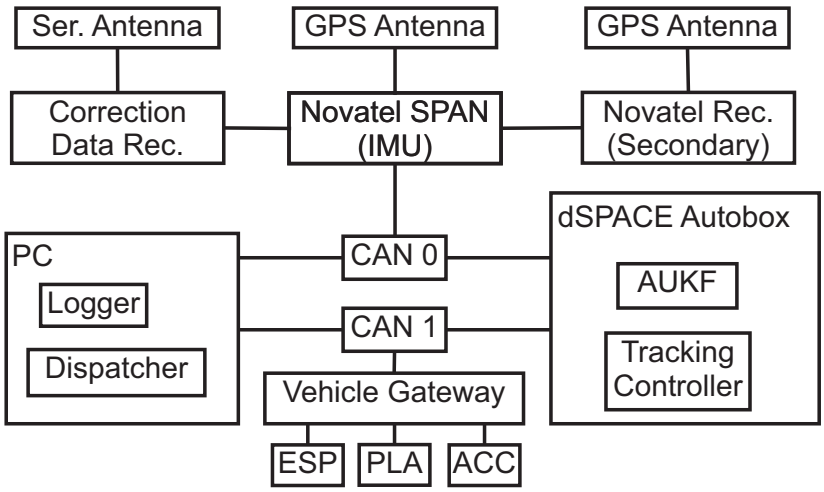


Figure 4: Sensor and data processing setup on FASCar2 vehicle

controller on the dSPACE Autobox. The tracking controller additionally receives inputs from an augmented unscented kalman filter [33] which estimates the state of a dynamic bicycle model. The requested actuator values steering angle, brake pressure and throttle valve, which are generated by the tracking controller, are sent via CAN1 to a CAN-gateway and then to the according vehicle sub-systems, as well as to the data logger on the PC. Methods for reading of CSV encoded measurement data[8], executed reference trajectroy[9] and local reference coordinate system[10] are available. As the tracking controller was under development during the conformance testing investigation, the corresponding version of the tracking controller[11] and the observer[12] are saved with the measurement data. Both s-functions are implemented in MATLAB and can be tested offline as documented in an example[13].

4.1 Abstract Vehicle Model

4.1.1 Dynamic Bicycle Model with Measured Inputs

Our vehicle model and also the applied parameters are equivalent to the model reported in deliverable 1.3, with the addition of disturbance inputs. The state vector consists of the position of the rear axle X_{COR}, Y_{COR}, the yaw angle ψ, the longitudinal velocity v_x, the lateral velocity at the center of the rear axle $v_{y,COR}$ and the yaw rate ω. The model inputs are the longitudinal acceleration $a_{x,m}$ and the steering angle δ_m. The state vector is completely

[8] `https://svn.dlr.de/UnCoVerCPS/T5.3/MATLAB/trackingTest_20161207/readVehicleData.m`

[9] `https://svn.dlr.de/UnCoVerCPS/T5.3/MATLAB/trackingTest_20161207/readTrajectoryData.m`

[10] `https://svn.dlr.de/UnCoVerCPS/T5.3/MATLAB/trackingTest_20161207/readReferenceSystem.m`

[11] `https://svn.dlr.de/UnCoVerCPS/T5.3/MATLAB/trackingTest_20161207/controller_sfunction.m`

[12] `https://svn.dlr.de/UnCoVerCPS/T5.3/MATLAB/trackingTest_20161207/ovserver_sfunction.m`

[13] `https://svn.dlr.de/UnCoVerCPS/T5.3/MATLAB/trackingTest_20161207/reference_implementation_vehicle_sim.m`

measured and we model additive measurement noise in all state dimensions. The disturbances are defined as three normalized forces, with the error force $e^d_{f_x}$ acting in longitudinal direction, $e^d_{f_{y,f}}$ acting in lateral direction at the front axle and $e^d_{f_{y,r}}$ acting in lateral direction at the rear axle.

$$x = [X_{COR}, Y_{COR}, \psi, v_x, v_{y,COR}, \omega]^{\mathrm{T}} \quad \text{state wrt COR} \tag{44}$$

$$u = [a_{x,m}, \delta_m]^{\mathrm{T}} \quad \text{input, measured} \tag{45}$$

$$y = [X^m_{COR}, Y^m_{COR}, \psi^m, v^m_x, v^m_{y,COR}, \omega^m]^{\mathrm{T}} \quad \text{measurement} \tag{46}$$

$$\nu = [e^m_{X_{COR}}, e^m_{Y_{COR}}, e^m_\psi, e^m_{v_x}, e^m_{v_{y,COR}}, e^m_\omega]^{\mathrm{T}} \quad \text{meas. err.} \tag{47}$$

$$w = [e^d_{f_x}, e^d_{f_{y,f}}, e^d_{f_{y,r}}]^{\mathrm{T}} \quad \text{disturbance} \tag{48}$$

In dependence on [32] the differential equation and the measurement function for the dynamic bicycle model are defined:

$$f_B(x,u,w) = \left\{ \begin{array}{l} \dot{x}_1 = x_4 \cos(x_3) - x_5 \sin(x_3) \\ \dot{x}_2 = x_4 \sin(x_3) + x_5 \cos(x_3) \\ \dot{x}_3 = x_6 \\ \dot{x}_4 = u_1 + x_5 \cdot x_6 + w_1 \\ \dot{x}_5 = f_{y,f}(x,u,w) + f_{y,r}(x,w) - x_4 \cdot x_6 - b \cdot \dot{x}_6 \\ \dot{x}_6 = a\frac{m}{J}\left(f_{y,f}(x,u,w)\right) - b\frac{m}{J}\left(f_{y,r}(x,w)\right) \end{array} \right\}. \tag{49}$$

$$f_{y,f}(x,u,w) = -c_f \mu g \frac{b}{a+b}\left(\frac{x_5 + (a+b)\cdot x_6}{x_4} - u_2\right) + w_2 \tag{50}$$

$$f_{y,r}(x,w) = -c_r \mu g \frac{a}{a+b}\frac{x_5}{x_4} + w_3 \tag{51}$$

$$h(x,u,\nu) = x + \nu \tag{52}$$

The state of the system is often expressed in different coordinates, using a polar-coordinate representation of the vehicle velocity, which is defined by the slip-angle $\beta = \arctan(v_y/v_x)$ and the absolute velocity $v = \sqrt{v_x^2 + v_y^2}$, with the direction of motion $\theta = \psi + \beta$. The model is used to investigate trace conformance according to the simplified approach in Figure 1(a), i.e., short-cutting the actuators and comparing only the physical equations. The model parameters given in table 1 are used subsequently.

Table 1: Parameters for Bicycle model

$J/m[m^2/s^2]$	$L[m]$	b/L	c_f	c_r	μ
1.57	2.7	0.57	−9.7	−25.2	1.0

4.1.2 Dynamic Bicycle Model with Requested Inputs

This vehicle model is an extension of the bicycle model f_B: As an addition, the delayed execution of requested inputs is modeled. The delays are defined by first order differential equations.

$$x = [X_{COR}, Y_{COR}, \psi, v_x, v_{y,COR}, \omega, a_x, \delta]^{\mathrm{T}} \quad \text{state wrt COR} \tag{53}$$

$$u = [a_{x,r}, \delta_r]^{\mathrm{T}} \quad \text{input, requested} \tag{54}$$

$$y = [X^m_{COR}, Y^m_{COR}, \psi^m, v^m_x, v^m_{y,COR}, \omega^m, a^m_x, \delta^m]^{\mathrm{T}} \quad \text{measurement} \tag{55}$$

$$\nu = [e^m_{X_{COR}}, e^m_{Y_{COR}}, e^m_{\psi}, e^m_{v_x}, e^m_{v_{y,COR}}, e^m_{\omega}, e^m_{a_x}, e^m_{\delta}]^{\mathrm{T}} \quad \text{meas. err.} \tag{56}$$

$$w = [e^d_{f_x}, e^d_{f_{y,f}}, e^d_{f_{y,r}}, e^d_{\dot{a}_x}, e^d_{\dot{\delta}}]^{\mathrm{T}} \quad \text{disturbance} \tag{57}$$

The differential equation and the measurement function are:

$$f_{BX}(x,u,w) = \begin{cases} [\dot{x}_1, \cdots, \dot{x}_6]^T = f_B\left([x_1, \cdots, x_6]^T, [x_7, x_8]^T, [w_1, \cdots, w_3]^T\right) \\ \dot{x}_7 = -k_{a_x}\left(x_7 - (u_1 + w_4 + a_{x,off}) \cdot a_{x,s}\right) \\ \dot{x}_8 = -k_{\delta}\left(x_8 - (u_2 + w_5 + \delta_{off}) \cdot \delta_s\right) \end{cases} \tag{58}$$

The extended model f_{BX} is used for conformance testing according to Figure 1(a). The conformance testing results validate the model for closed loop use. The control inputs that are calculated by the trajectory tracking controller during closed loop operation (steering angle, longitudinal acceleration) considerably differ from the control inputs that are currently realized by the vehicle. This is due to message delays, actuator dynamics and actuator mis-configuration (in case of longitudinal control). The parameters of the simple, first order ODEs of the actuators have been identified by least squares minimization[14] and are given in Table 2. Figure 5 displays the benefits of using the actuator model: Instead of more than 2° steering angle mismatch under assumption of applying the requested inputs directly, the mismatch between the actuator model and the measured steering angle is approx. 0.5° in the given test-run. In Figure 6 the evaluation[15] for longitudinal actuator delays is given.

$a_{x,off}[m/s^2]$	$a_{x,s}$	$k_{a_x}[1/s]$	$\delta_{off}[^\circ]$	δ_s	$k_{delta}[1/s]$
0.32	0.64	2.96	1.09	0.28	9.83

Table 2: Actuator parameters for extended bicycle model

[14] `https://svn.dlr.de/T5.3/MATLAB/ConformanceTesting/test_steering_angle_delay.m`

[15] `https://svn.dlr.de/T5.3/MATLAB/ConformanceTesting/test_acceleration_delay.m`

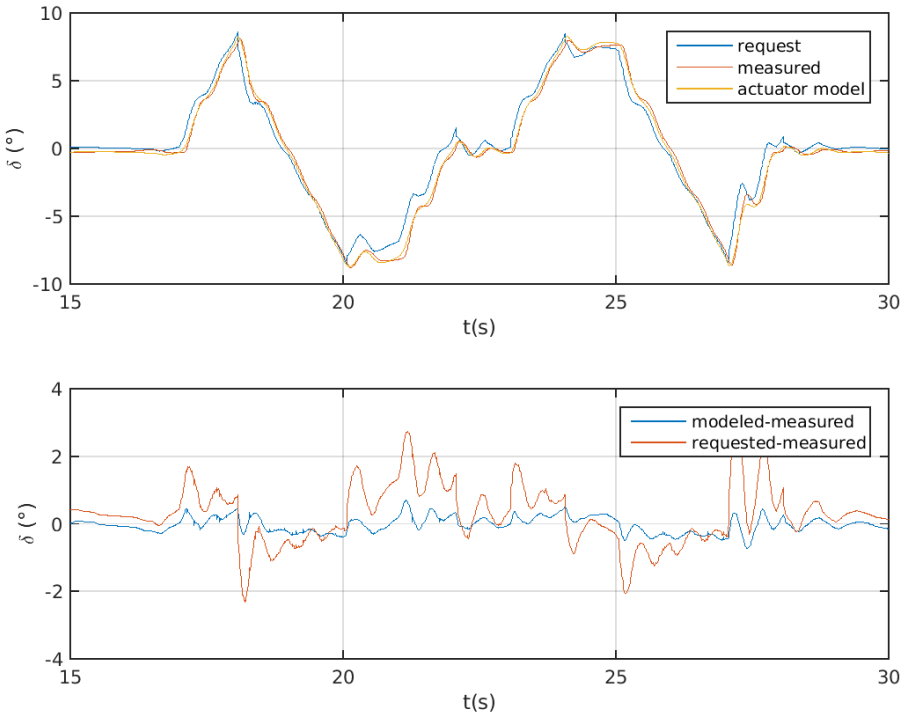


Figure 5: Comparison between modeled and unmodeled delay in requested steering angle.

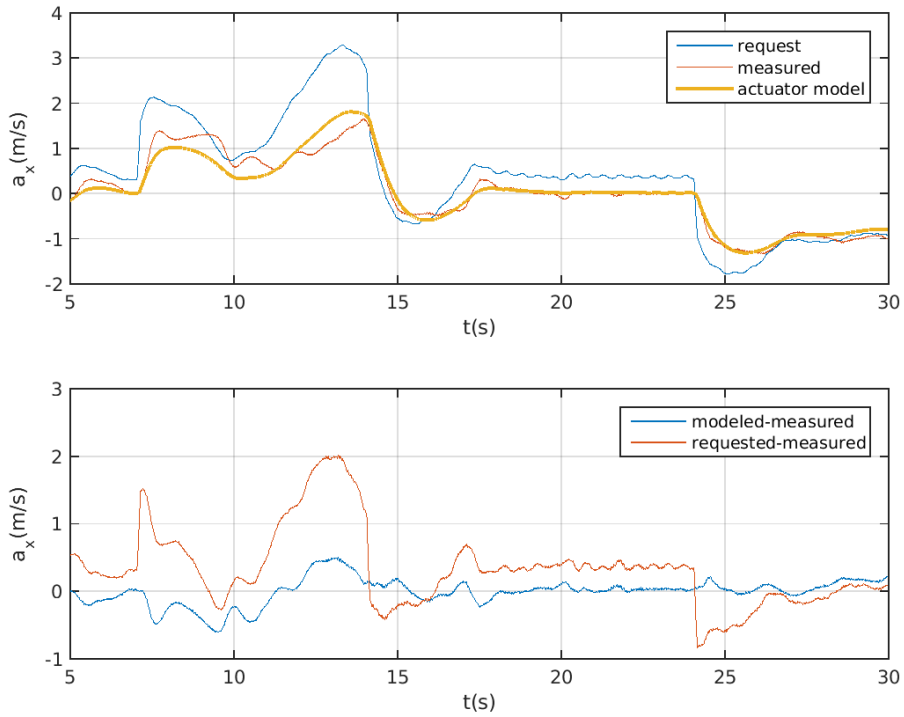


Figure 6: Comparison between modeled and unmodeled delay in requested longitudinal acceleration. A running average of 21 values of the measured acceleration is shown.

4.2 Results Trace Conformance

4.2.1 Trace Conformance without Actuator Model

Using model f_B with the following maximum disturbances and maximum measurement errors given in Table 3, furthermore using measured acceleration and steering angle as inputs for the model, trace conformance is shown for all A1-4, C1-4 December 2016 test drives. The

Table 3: Trace conformance of vehicle w/o actuator model: Measurement errors and disturbances

$\nu_1 : e^m_X[m]$	$\nu_2 : e^m_Y[m]$	$\nu_3 : e^m_\psi[^\circ]$	$\nu_4 : e^m_\theta[^\circ]$	$\nu_5 : e^m_v[m/s]$	$\nu_6 : e^m_\omega[^\circ/s]$
0.04	0.04	1	1	0.05	2
$w_1 : e^d_{f_x}[m/s^2]$	$w_2 : e^d_{f_{y,f}}[m/s^2]$	$w_3 : e^d_{f_{y,r}}[m/s^2]$			
$0.1g$	$0.057g$	$0.043g$			

results of the trace conformance evaluation[16] are plotted in Figure 26 to 33 in Appendix B. The inputs for trace-conformance[17], e.g., measurement vector Y, time T and input vector U, which are generated from the experiment recordings, as well as the outputs[18], which are generated by the trace conformance algorithm, are saved and provided for future use.

4.2.2 Trace Conformance with Actuator Model

Using model f_{BX} with the following maximum disturbances and maximum measurement errors given in Table 4, furthermore using requested acceleration and steering angle as inputs for the model, trace conformance is shown for all A1-4, C1-4 December 2016 test drives. The results of the trace conformance evaluation[19] are plotted in Figure 34 to 41 in Appendix C. The inputs for trace-conformance[20], e.g., measurement vector Y, time T and input vector U, which are generated from the experiment recordings, as well as the outputs[21], which are generated by the trace conformance algorithm, have been saved and provided for future use.

[16] `https://svn.dlr.de/UnCoVerCPS/T5.3/Matlab/ConformanceTesting/D5_2_TC_Bicycle_process_all.m`

[17] `https://svn.dlr.de/UnCoVerCPS/T5.3/Matlab/ConformanceTesting/D5_2_TC_Bicycle_DecA1_tc_input.mat`, etc.

[18] `https://svn.dlr.de/UnCoVerCPS/T5.3/Matlab/ConformanceTesting/D5_2_TC_Bicycle_DecA1_tc_output.mat`, etc.

[19] `https://svn.dlr.de/UnCoVerCPS/T5.3/Matlab/ConformanceTesting/D5_2_TC_Bicycle_Delayed_process_all.m`

[20] `https://svn.dlr.de/UnCoVerCPS/T5.3/Matlab/ConformanceTesting/D5_2_TC_Bicycle_Delayed_DecA1_tc_input.mat`, etc.

[21] `https://svn.dlr.de/UnCoVerCPS/T5.3/Matlab/ConformanceTesting/D5_2_TC_Bicycle_Delayed_DecA1_tc_output.mat`, etc.

Table 4: Trace conformance of vehicle with actuator model: Measurement errors and disturbances

$\nu_1 : e^m_X[m]$	$\nu_2 : e^m_Y[m]$	$\nu_3 : e^m_\psi[^\circ]$	$\nu_4 : e^m_\theta[^\circ]$	$\nu_5 : e^m_v[m/s]$	$\nu_6 : e^m_\omega[^\circ/s]$
0.04	0.04	1	1	0.2	2
$\nu_7 : e^m_{a_{x,m}}[m/s^2]$	$\nu_8 : e^m_{\delta_m}[^\circ]$	$\nu_9 : e^m_{a_{x,r}}[m/s^2]$	$\nu_10 : e^m_{\delta_r}[^\circ]$		
2	1	0.2	0.2		
$w_1 : e^d_{f_x}[m/s^2]$	$w_2 : e^d_{f_{y,f}}[m/s^2]$	$w_3 : e^d_{f_{y,r}}[m/s^2]$	$w_4[m/s^2]$	$w_5[^\circ]$	
$0.02g$	$0.011g$	$0.009g$	0.5	1	

4.2.3 Concluding Remarks on Trace Conformance for the DLR Vehicle

Trace conformance is shown between two vehicle models and the physical FASCar2 vehicle. At first, for a reduced model, we replace actuation requests (actuator inputs) by measured actuator outputs in order to deal with the lateral vehicle dynamics only. In a second step we consider an extended vehicle model with actuator delays, which is representative of the full, closed-loop system. For each model, a single, reasonable set of error bounds is derived, which explains each of the eight different maneuvers with five repetitions per maneuver, i.e. a total of 40 measurement traces.

It was instructive to observe that some maneuvers actually require bigger error bounds than others, especially for higher lateral accelerations. Thus it is useful for a comprehensive safety analysis, to investigate different maneuvers and to densely cover the state space with test drives.

From the perspective of conformance testing, the overall results are satisfactory. First tests with reachability analysis unfortunately show, that the bounds derived here for trace conformance do not lead to convergent reachable sets. We assume that this is in part due to the fact that actuators and sensors on the physical vehicle do not perform as well as could be expected. A worst-case concatenation of sensor and actuator deficiencies could probably destabilize the real vehicle. The quality of measurements outputted by the IMU is not quite sufficient to be used for vehicle control directly (e.g. the yaw-rate is rather noisy, and the yaw-angle appears to be inflicted with an almost constant offset error). Because of this, a model-based observer is already employed on the vehicle for closed-loop control.

A next step could be to quantify the error of the model-based observer instead of the raw IMU measurements, in order to incorporate this smaller error in reachability analysis. Finally, one could consider how to reduce actuator errors by removing offsets and improving performance maps or actuator level controllers.

4.3 Results Reachset Conformance

In the following section we present the results on reachset conformance testing of the DLR vehicle model. The model is a closed-loop model as presented in Section 4.1.1 including a refined time-discretized version of the tracking controller presented in deliverable D5.1. As a structure for the additive disturbance applied to the ODEs of the vehicle model, we chose

$$\begin{aligned} w &= [w_1, w_2, w_3, w_4, w_5, w_6] && (59)\\ &= [e^d_{f_{\dot{X}}}, e^d_{f_{\dot{Y}}}, e^d_{f_{\dot{\Psi}}}, e^d_{f_{\dot{v}_x}}, e^d_{f_{\dot{v}_y}}, e^d_{f_{\dot{\omega}}}] && (60) \end{aligned}$$

such that each ODE $\dot{x}_i = f_i(x,u) + w_i$ with $i = 1 \dots 6$ is disturbed by an additive term w_i.

4.3.1 Taylor Series Models

As reachset conformance testing relies on the computation of reachable sets of a model (cf. Section 3.3), (non-linear) reachability analysis, e.g., using CORA, has to be performed. During our initial investigations we found that the computation time of CORA was very high due to the time-consuming function evaluation capabilities of MATLAB. To be more specific, we found the performance bottleneck to be the evaluation of the symbolic Lagrange remainder of the model linearization computed in CORA.

As a remedy, in order to achieve reasonable performance in reachabilty analysis of the non-linear vehicle model in CORA, we compute a Taylor series expansion of the closed-loop vehicle model. Given $z = [x, u, w]^T \in \mathbb{R}^{n_z}$, a Taylor series expansion of $\dot{x} = f(x,u,w)$ can be computed by

$$\begin{aligned} \dot{x} \approx \; & f(z_0) + \sum_{j=1}^{n_z} \left.\frac{\partial f(z)}{\partial z_j}\right|_{z=z_0} \Delta z_j \\ & + \sum_{j,k=1}^{n_z} \left.\frac{\partial^2 f(z)}{\partial z_j \partial z_k}\right|_{z=z_0} \Delta z_j \Delta z_k \\ & + \sum_{j,k,l=1}^{n_z} \left.\frac{\partial^3 f(z)}{\partial z_j \partial z_k \partial z_l}\right|_{z=z_0} \Delta z_j \Delta z_k \Delta z_l \\ & + \cdots . \end{aligned} \tag{61}$$

Since Taylor series expansions are most accurate around their expansion point z_0, one has to be careful with generalizing a single Taylor series model to explain all the measured maneuvers. In the worst case, a faulty Taylor series model might induce additional disturbances caused by the inaccuracy of the model. As a consequence, the goal in reachset conformance testing is to find a Taylor series model of the original model for each maneuver which explains all measurements of a maneuver.

The expansion point z_0 for the Taylor series of each maneuver is defined by averaged values of the reference input trajectory for the input u and averaged states x of the open-loop model following the reference trajectory. For the expansion point, the disturbance w to the zero vector. Concerning the order of the Taylor series expansion, we found by comparing the simulations of the original model and a model of second order that a second order Taylor series does not sufficiently represent the behaviors of the original model for all maneuvers. Unfortunately, simulation of a third order model using MATLAB is not possible because the file size of the model is around 11MB and thus functions calls of the model are very time consuming. Furthermore, reachability analysis of a third order model in CORA is around 20 times slower compared to the analysis of a second order model. As a remedy, we investigated which state space equations rely on a third order expansion and found that it is sufficient to do a second order Taylor series for the ODEs of all state variables except x_2 (position of the vehicle on the global Y-axis). Therefore, only the ODE of x_3 is approximated with a third order series. Overall, this approach solves all our accuracy and performance problems.

4.3.2 Reuse of Trace Conformance Results for Reachset Conformance

In principle, one could start reachset conformance testing by doing a random search for a set of disturbances $\mathcal{W}$ such that equation (43) is fulfilled. However, in practice, random search does not scale well as either models are not conformant or reachability analysis performs badly due to the splitting of reachable set during analysis. The latter is usually caused by high disturbances although high values for disturbances potentially explain all measurements. As trace conformance implies reachset conformance [35], we can, however, reuse trace conformant models as a starting point, then identify a reachset conformant model with less conservative $\mathcal{W}$. We use the approach from Section 3.2.1 to compute disturbances $w(t_k)$ for trace conformance models for each measurement $\langle U_j, Y_j, T_j \rangle$ independently. Note that we use the original vehicle model to derive the disturbances $w(t_k)$ for performance reasons as the larger file size of the Taylor series model slows down the optimization process in equation (27). We checked that the Taylor series model is trace conformant to the measurements under assumption of the measurement errors from Table 5, using the obtained disturbances on the original model. Therefore, these disturbances can be used as a starting point for the reachset conformance analysis on the Taylor model.

In the following, we follow a three-step approach for each measurement:

1. We start with the disturbances as computed by the approach in Section 3.2.1 and compute a bounding box $\mathcal{W}^{TC}$, which is used as an additive uncertainty for the reachability

analysis. No measurement errors are assumed during this step.

2. We then employ non-linear optimization to minimize the volume of the $\mathcal{W}^{TC}$, such that the resulting model is still trace conformant under the assumption of measurement errors. The disturbances from Step 1 are used as an initial value for the optimization.

3. The resulting minimal bounding box from step 2 is then used as initial value for another optimization step, leading to a model with smaller disturbances that is reachset conformant, but not necessarily trace conformant. This model is a Pareto optimal reachset conformant model, i.e., no reachset conformant model with smaller disturbances exists.

The result is then a reachset conformant model for each measurement. These models per measurement are then merged for all measurements belonging to a single maneuver, resulting in a model which can be used for formal verification.

Step 1 is as follows: In order to compute $\mathcal{W}^{TC}$, we over-approximate the disturbances $w(t_k)$ for each $w_i(t_k)$ from the trace conformant model by time independent interval hulls with the resulting zonotope representation of

$$\mathcal{W}_i^{TC} = c_i^{TC} + \beta \cdot g_i^{TC} \tag{62}$$

with

$$c_i^{TC} = w_i^{min}(t_k) + g_i^{TC} \tag{63}$$

$$g_i^{TC} = (w_i^{max}(t_k) - w_i^{min}(t_k))/2 \tag{64}$$

$$w_i^{min} = \min\left(w_i(t_k)\right) \tag{65}$$

$$w_i^{max} = \max(w_i(t_k)) \tag{66}$$

and $\beta \in [-1, 1]$. Thus, the Cartesian product $\mathcal{W}^{TC}$ of the $\mathcal{W}_i^{TC}$ represents a six dimensional (6D) box.

To check, whether this model already gives reasonable reachsets, we performed reachability analysis with the disturbances $\mathcal{W}^{TC}$ for each measurement $\langle U_j, Y_j, T_j \rangle$ independently where the initial reachset for each analysis $Reach_1^j$ is defined by

$$Reach_1^j = X_j(t_1) \oplus \mathcal{V}. \tag{67}$$

Unfortunately, the reachability analysis computations did not finish within a reasonable amount of time as CORA needs to split reachable set such that the linearization error in the analysis does not become too large. This can be tracked back to a large set of disturbances

$\mathcal{W}^{TC}$. As we do not optimize the values of disturbances $w(t_k)$ in the approach presented in Section 3.2.1, this result could be anticipated. Therefore, we continued with Step 2 of the approach, applying (non-linear) optimization to the volume $V_{\mathcal{W}^{TC}}$ of the convex set $\mathcal{W}^{TC}$. During the optimization process, the non-linear programming solver is able to reduce the disturbances $w_i(t_k)$ by respecting the measurement error $\mathcal{V}$. Table 5 summarizes assumed maximal measurement errors on each state variable. Overall, we implemented the optimization problem with goal $V^{opt}_{\mathcal{W}^{TC}} = \min(V_{\mathcal{W}^{TC}})$ such that trace conformance holds (cf. equations (22)-(24)). In contrast to the approach described in Section 3.2.2, this approach does not require domain knowlege to set potentially satisfiable bounds of maximal assumed disturbances in $\mathcal{W}$, but explicitly minimizes the current maximal values that contribute to $\mathcal{V}_{W^{TC}}$. On the downside, one relies on an implementation of a non-linear programming solver with good performance. We use MATLAB's non-linear programming solver $fmincon$ (version 2016b) and optimize each the initial sets $\mathcal{W}^{TC,j}$ associated with each measurement $\langle U_j, Y_j, T_j \rangle$ for three hours.

We found that the resulting $V^{opt}_{\mathcal{W}^{TC,j}} = \min(V_{\mathcal{W}^{TC,j}})$ are order of magnitude smaller than the initial sets. However, showing reachset conformance based on the optimized volumes still leads to splitting of reachable sets in CORA. Since trace conformance is a more strict notion of conformance than reachset conformance and thus potentially results in higher disturbances, this result could also be anticipated.

Therefore, we proceeded with Step 3 by requiring only reachset conformance instead of trace conformance. We perform a binary optimization of each $V^{opt}_{\mathcal{W}^{TC,j}}$, such that

$$V^{bin-opt}_{\mathcal{W}^{RC,j}} = \underset{\alpha \in [0,1]}{\arg\min} \left(c^{TC,opt,j} + \alpha \cdot (\beta \cdot g^{TC,opt,j}) \right) \tag{68}$$

with the constraint that a model with disturbances $\mathcal{W}^{RC,j} = c^{TC,opt,j} + \alpha \cdot (\beta \cdot g^{TC,opt,j})$ is reachset conformant (RC). Thereby, $c^{TC,opt,j}$ and $g^{TC,opt,j}$ are the center and generator of a 6D axis-aligned box approximation of the disturbances that result from the optimization of trace conformance disturbances w.r.t to a measurement error $\mathcal{V}$ for each measurement $\langle U_j, Y_j, T_j \rangle$. Using this approach with 10 iterations during binary optimization, we were able to find sets of disturbances $\mathcal{W}^{RC,j}$ such that the Taylor models are reachset conformant to each respective measurement of the lane-change and double-lane-change maneuver with lateral

$\nu_1 : e^m_X[m]$	$\nu_2 : e^m_Y[m]$	$\nu_3 : e^m_\Psi[^\circ]$	$\nu_4 : e^m_{v_x}[m/s]$	$\nu_5 : e^m_{v_y}[m/s]$	$\nu_6 : e^m_\omega[^\circ/s]$
0.05	0.05	0.5	0.1	0.1	0.8

Table 5: Measurement errors used in reachset conformance testing

Maneuver	$\mathcal{W}_1^{RC,Mnvr}$ $[m/s]$	$\mathcal{W}_2^{RC,Mnvr}$ $[m/s]$	$\mathcal{W}_3^{RC,Mnvr}$ $[rad/s]$
LC	$[-0.0831, 0.0552]$	$[-0.1173, 0.0274]$	$[-0.0593, 0.0678]$
DLC	$[-0.0564, 0.0227]$	$[-0.1017, -0.0091]$	$[-0.0100, 0.0102]$
Maneuver	$\mathcal{W}_4^{RC,Mnvr}$ $[m/s^2]$	$\mathcal{W}_5^{RC,Mnvr}$ $[m/s^2]$	$\mathcal{W}_6^{RC,Mnvr}$ $[rad/s^2]$
LC	$[-0.3203, 0.7775]$	$[-0.1503, 0.1051]$	$[-0.4414, 0.7231]$
DLC	$[-0.3215, 0.7528]$	$[-0.2148, 0.0530]$	$[-0.3695, 1.0267]$

Table 6: Intervals of disturbances $\mathcal{W}_i^{RC,Mnvr}$ of reachset conformant Taylor models for each maneuver

acceleration of $2m/s^2$. For all other maneuvers, due to splitting of reachsets in CORA for high values of α, we were not yet able to derive reachset conformant models.

Up to this point, we have derived disturbances $\mathcal{W}^{RC,j}$ of reachset conformant vehicle models individually for each measurement corresponding to a lane-change or double-lane-change maneuver with lateral acceleration of $2m/s^2$. As mentioned before, the goal is to derive a single reachset conformant model which explains all measurements of a maneuver. For the lane-change and the double lane change we merged all reachset conformant sets of disturbances $\mathcal{W}^{RC,j}$ that belong to respective maneuver, and again represent this sets by a 6D axis-aligned box approximation. Since the box approximation tends to be too conservative, we again optimize the volume of the box via binary optimization similar to (68) with 10 iterations and denote the resulting set of disturbances $\mathcal{W}^{RC,Mnvr}$. Here it is important to note that $\mathcal{W}^{RC,Mnvr}$ has to explain all the measurements of a maneuver.

In Table 6 we show the resulting set of disturbances $\mathcal{W}^{RC,Mnvr}$ in interval representation for the lane-change (LC) and double-lane-change (DLC) maneuvers. The Cartesian product of all $\mathcal{W}_i^{RC,Mnvr}$ represents the 6D box $\mathcal{W}^{RC,Mnvr}$.

Figure 7 and Figure 8 show the reachable sets of the $X-$ (x_1) and $Y-$position (x_2) of the center of the rear axle of the vehicle model for the two reachset conformant maneuvers. While the vehicle models are reachset conformant, the resulting reachable sets are noticeably conservative. Here, future work might include further exploration of the Pareto front of conformant models, as the model given through the intervals above is only one point on the Pareto front (cf. binary optimization in (68)). Furthermore, one could reduce the horizons of the measurement data and search for conformant Taylor models on smaller segments within maneuvers. Thereby, disturbances on straight and cornering segments do not get mixed. This approach does not interfere with the maneuver database (MDB) approach of UnCoVerCPS since conformance can still be tested for each maneuver in the database.

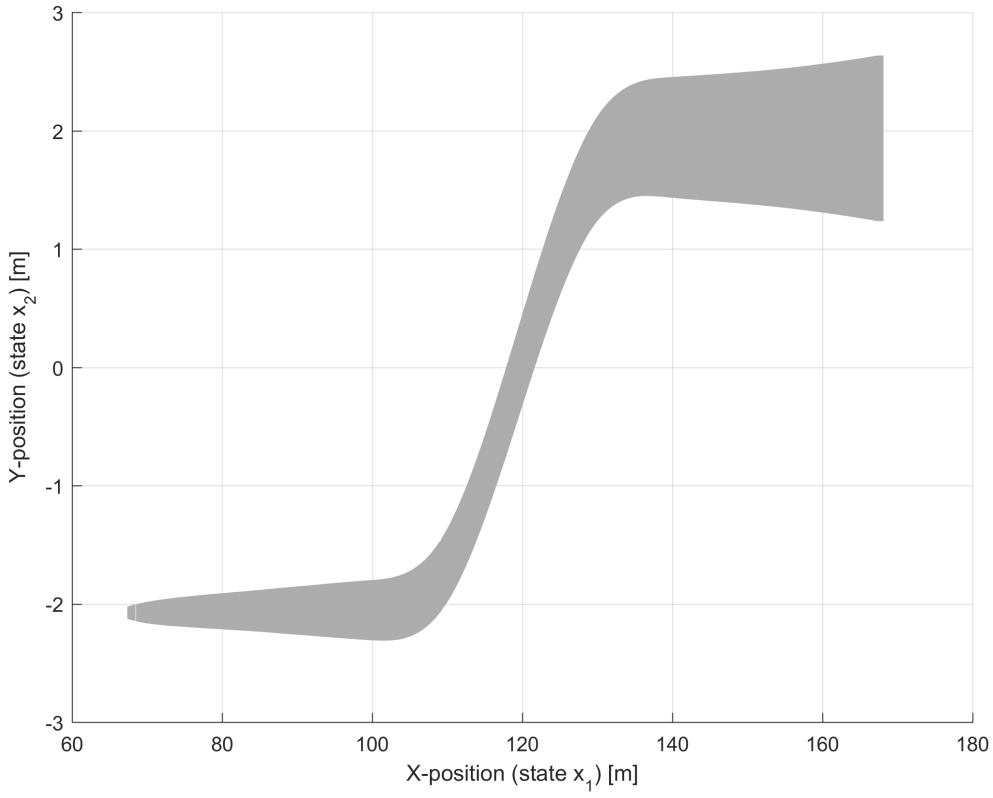


Figure 7: Projection of reachable sets for the Taylor vehicle model with disturbances $\mathcal{W}^{RC,Mnvr}$ for one lane-change maneuver

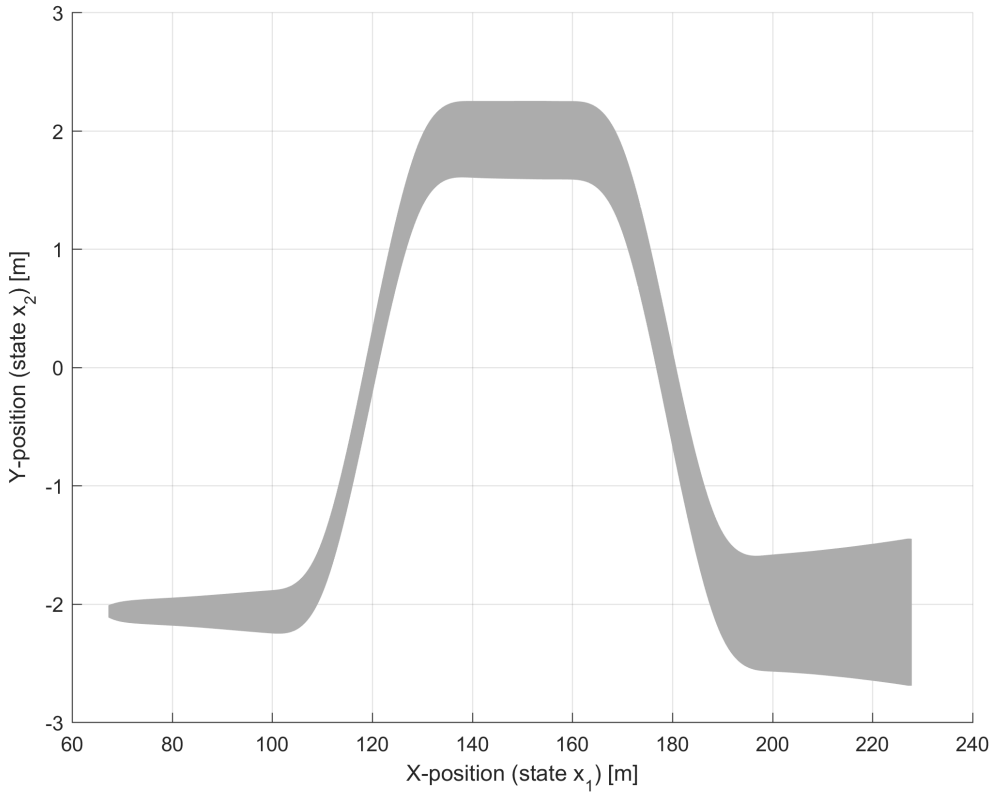


Figure 8: Projection of reachable sets for the Taylor vehicle model with disturbances $\mathcal{W}^{RC,Mnvr}$ for one double-lane-change maneuver

5 Use Case Tecnalia Vehicle

Similar to the DLR use case presented in the previous section, we present the results of the trace conformance testing of the Tecnalia vehicle (see Appendix D for more details on the Twizy platform). Thereby, our reference data is based on high fidelity simulations of a multi-body vehicle model.

5.1 Abstract Vehicle Model

In deliverable D5.1 (Section 4), we described the model used for the lateral dynamics validation (equations 41 and 42, page 45). The lateral dynamic is modeled by a bicycle model (see Section 4.1). In this section, a description of the parameters used in the open-loop estimation of the Twizy Vehicle is presented.

The bicycle model has been implemented in MATLAB and has the same state vector as the DLR vehicles (see deliverable D5.1), where the main parameters are: vehicle total mass, ratio of mass and rotational inertia (J_m), wheelbase, distance from rear axle to center of gravity (COG), COG height, front cornering stiffness, rear cornering stiffness and steering wheel ratio. The main benefit of using the same model is to provide a interchangeable comparison between the vehicles.

The MATLAB bicycle model has been identified with the techniques described in deliverable D5.1 based on test drives of the Twizy vehicle. Steering wheel angle and longitudinal acceleration are taken from raw measurement data. Table 7 shows the values estimated. Furthermore, in Appendix D we show a classical validation of the of open-loop and closed-loop vehicle model (with a mass of the vehicle of 582.5 kg). This validation was made using an abstract vehicle model that compares its states with the ones from the collected data.

$I_z/m[m^2]$	$b/L[m/m]$	c_f	c_r
0.5150	0.4502	4.9906	7.5122

Table 7: Results of the plan parameter identification (based on the Twizy vehicle)

5.2 Reference Data

The reference data used for the trace conformance testing comes from the Dynacar simulator, which uses a multi-body model of the Twizy vehicle (see Appendix D). In our case, this simulator has similar performance as the real vehicle.

The reference data consists of a lane-change and a double-lane-change maneuver. The data is conformed by a state vector which contains position, orientation, speeds and accelerations (lateral and longitudinal); and the data used to defined the trajectory is more related with geometrical aspects. The information used is equivalent to the DLR's data format.

5.3 Results

This section will be used to present the results of conformance testing, based on the the abstract open- and closed-loop model. In this sense, the parameters assumed for the trace conformance are the maximum measurement errors (ν) and disturbances (w) shown in Table 8. This parameters were selected based on real sensor measurement errors.

For the simulation results, the values decrease due to the accuracy of the models (the errors and disturbances are considered equal for the open-loop and closed-loop case). In this sense, the multi-body model of the Twizy allows to reduce some bounds as measurement errors of position, orientation and rate of change of the orientation; and additionally, the disturbances of longitudinal and lateral forces.

$\nu_1 : e^m_X[m]$	$\nu_2 : e^m_Y[m]$	$\nu_3 : e^m_\psi[^\circ]$	$\nu_4 : e^m_\theta[^\circ]$
0.01	0.01	1.00	0.05
$\nu_5 : e^m_v[m/s]$	$\nu_6 : e^m_\omega[^\circ/s]$		
0.05	0.4		
$w_1 : e^d_{f_x}[m/s^2]$	$w_2 : e^d_{f_{y,f}}[m/s^2]$	$w_3 : e^d_{f_{y,f}}[m/s^2]$	$w_4 : e^d_{a_x}[m/s^2]$
$0.009g$	$0.02g$	$0.02g$	0.07
$w_5 : e^d_\delta[^\circ]$			
0.8			

Table 8: Overall resulting trace conformance measurement error and disturbances bounds

Figure 51 and 52 (in Appendix E) show the detailed trace conformance results for the open-loop case (just feed-forward controller). Figure 53 and 54 present the results with the integration of the feedback controller (closed-loop part).

The disturbances ($e^d_{f_x}$, $e^d_{f_{y,f}}$, $e^d_{f_{y,f}}$) show a different behavior in both conformance (open and closed). In the open-loop case, these signals are approximately zero, due it is considering the position errors (distance to the reference trajectory). Adding the feedback controller, the trace conformance on the disturbances are different in the closed-loop case since the values are not close to zero all the time. It is due to the better tracking of the lateral error e^m_Y.

Overall, our result shows a good matching of the model (open-loop control) into the

acceptable bounds of the measurement errors, and for the closed-loop controller (feedback) the results fit into the measurement error bounds.

6 Use Case Windturbine

One of the applications considered in UnCoVerCPS is a wind turbine model provided by General Electric. Upon completion of conformance testing the wind turbine use case will be discontinued. Wind turbine controllers often rely on linearized, reduced-order models. In order to design more performant and more flexible control schemes, it is of high importance to better understand the capabilities and limitations of these models. In this chapter, we quantify the modeling errors of an abstract, open-loop wind turbine model such that a resulting model, including the estimated errors, is trace conformant to a more sophisticated high fidelity wind turbine model. Therefore, similar to the conformance testing presented for the Tecnalia vehicle in this document, the reference data for the conformance testing is not measured data but simulation results of the high fidelity wind turbine model.

6.1 Abstract Wind Turbine Model

In [36] a hybrid wind turbine model has been derived. While the turbine itself is not a hybrid system, it uses a hybrid switching controller. Modeling this hybrid switching controller accurately as a hybrid automaton in reachability analysis tools for non-linear systems, e.g., CORA, is considered to be a tremendous effort as this process is not fully automated by existing tools yet. As a remedy, we use an open-loop abstract wind turbine model. A drawback of this approach is that reachability analysis and analysis of a trace conformant model via uncertainty sampling is very likely to result in instable behavior of the system. Nonetheless, the quality of the abstract open-loop model can be estimated by building a trace conformant model with high-fidelity simulations as reference data. If reasonable disturbances need to be applied to the abstract model in order to achieve trace conformance, the undisturbed abstract model can be considered adequate.

In case of the open-loop model, the control inputs M_g (generator torque) and θ (blade pitch angle) are obtained from the high-fidelity simulation. Since no controller is considered, the resulting model is a non-linear, continuous-time differential equation. The dynamic behavior of rotor speed Ω and tower position x_T is given by the ordinary differential equations (ODEs)

$$\dot{\Omega} = (M_a(\dot{x}_T, \Omega, \theta, v_0) - M_g/i) \cdot 1/J \tag{69a}$$

$$\dot{x}_T = \ddot{x}_T \tag{69b}$$

$$\ddot{x}_T = (F_a(\dot{x}_T, \Omega, \theta, v_0) - c_{Te}\dot{x}_T - k_{Te}x_T) \cdot 1/m_{Te} \tag{69c}$$

with state space vector $x = [\Omega, x_T, \dot{x}_T)]^T$. These ODEs describe drive train shaft dynamics (cf. (76a), Appendix F) and elastic tower fore-aft motion (cf. (76b), Appendix F). Here, v_0 is

the rotor effective wind speed, i is the gearbox ratio, J is moment of inertia about the rotor axis, and m_{Te}, c_{Te} and k_{Te} are the tower equivalent model mass, structural damping and bending stiffness, respectively. The corresponding parameter values are given in Table 15 (Appendix F). The non-linearity in the model is contained in the aerodynamic thrust F_a and the aerodynamic rotor torque M_a. Expressions for F_a and M_a are given in (77) and (78) in Appendix F, which summarizes all details on the derivation of the abstract wind turbine model. Due to initial validations of the continuous open-loop wind turbine model, we know that the model is most suitable for average wind speeds of 8 m/s, and with limitations up to wind speeds of 10 m/s and 12 m/s. This observation mainly tracks back to the open-loop character of the abstract model. Our reference data respects the validity of the model by containing appropriate wind speed ranges.

6.2 Reference Data

6.2.1 High Fidelity Tool

High fidelity wind turbine performance data is generated using the aero-elastic simulator FAST (Fatigue, Aerodynamics, Structures and Turbulence). FAST is an open source software that is distributed by NREL (National renewable energy laboratory) [20]. The FAST model employs a combined modal and multibody dynamics formulation. The model for a three bladed horizontal axis turbine contains up to 24 degrees of freedom. Additionally, unsteady blade element momentum theory is applied to compute the aerodynamic forces on the blades in a turbulent wind field. FAST was evaluated by Germanischer Lloyd WindEnergie and was found suitable for the calculation of onshore wind turbine loads for design and certification.

For trace conformance testing the generator torque M_g, the blade pitch angle θ, the rotor speed Ω, the tower top deflection x_T and the tower base fore-aft bending moment M_{yT} are extracted from the FAST result files. While M_g and θ are control inputs, conformance will be checked for Ω, x_T and M_{yT}. Here, we are especially interested in the results for Ω and M_{yT}. As the tower base fore-aft bending moment M_{yT} has to be compensated by the concrete in the tower base, the quantity M_{yT} directly translates into the costs of building a wind turbine. Thus, the abstract model should reflect this appropriately in order to support cost savings.

6.2.2 Reference Turbine

All simulations have been performed on the NREL 5-MW reference turbine [21]. This wind turbine is a conventional three-bladed upwind variable-speed variable blade-pitch-to-feather-controlled turbine, which is representative of typical utility-scale multimegawatt turbines. The

turbine has been used as a reference by research teams throughout the world to standardize baseline wind turbine specifications and to quantify the benefits of advanced wind energy technologies.

6.2.3 Wind Disturbance

The wind is acting as a disturbance on the turbine. In general, the wind field impacting the turbine is three-dimensional and stochastic. For wind turbine certification, wind fields have to be generated according to the IEC standards [1] with a certain turbulence class. For this study, we have used the tool TurbSim [22] to generate turbulent wind fields with IEC turbulence class A and mean wind speeds of 8 m/s, 10 m/s and 12 m/s. Wind shear was set to zero since no wind shear was considered in the WT_perf simulations (cf. Appendix F). For the abstract wind turbine model, the wind fields are reduced to one-dimensional rotor effective wind speeds v_0 by calculating the average of the longitudinal wind component over the rotor plane.

For the mean wind speeds of 8 m/s we obtained 50,402 reference states for conformance testing, for wind speeds of 10 m/s 22,902 reference states and for wind speeds of 12 m/s 5,000 reference states from the high fidelity simulation. As mentioned before, the reference data respects the validity of the abstract model as conformance testing only makes sense in domains where the abstract model is assumed to be correct.

6.3 Trace Conformance Setup

As structure for the disturbances we selected additive disturbances $w = [e^d_{\dot{\Omega}}, e^d_{\dot{x}_T}, e^d_{\ddot{x}_T}]^T$ to each of the three ODEs in (69). To estimate the disturbances, we apply the stricter trace conformance approach presented in (Sec. 3.2.1) since the high fidelity simulation data does not suffer any noise during measurement. Furthermore, the computation of the disturbances can be done in parallel computations which is necessary for performance reasons since the reference data is around three times as large as in the automated driving use cases.

Without any measurement error, as mentioned in Section 3.2.1, computing a trace conformant model can be implemented by solving the optimization problem of (26). In practice, we achieved a good performance of MATLAB's (Version 2016b) non-linear programming solver *fminunc* by changing the objective function such that

$$\forall k : w(t_k) = \underset{w(t_k)}{\arg\min} \left(\sum_i (([y(t_{k+1})]_i - [\tilde{x}(t_{k+1})]_i)/[y(t_{k+1})]_i)^2 \right). \tag{70}$$

Note that this objective function is possible in our case since any component of the reference data is not equal to zero at any instance of time. As mentioned before, the abstract wind

turbine model should be trace conformance w.r.t. the states Ω and x_T and tower base fore-aft bending moment M_{yT} where the latter is an output and not a state of the abstract model. Thus, $Y = [y(t_1), y(t_2), \ldots, y(t_k), \ldots, y(t_K)]$ is a trace of the tuple $\langle \Omega, x_T, M_{yT} \rangle$. Since disturbances $w = [e^d_{\dot{\Omega}}, e^d_{\dot{x}_T}, e^d_{\ddot{x}_T}]^T$ are applied on the state space equations in (69), we have to rewrite $\tilde{x}(t_{k+1})$ in equation (25):

$$y(t_{k+1}) = \underbrace{g\left(g^{-1}(y(t_k)) + \int_{t_k}^{t_{k+1}} f\left(x(\tau), u(t_k), w(t_k)\right) d\tau\right)}_{\tilde{x}(t_{k+1}):=}. \tag{71}$$

Here, $g(x)$ is the bijective output map ($g : x \in X \rightarrow y \in Y$) that can be computed by using equation (80).

While numerical inaccuracies may arise due to numerical integration in the objective and step tolerances in the optimization procedure, we found that the maximal deviation in the conformant dimensions Ω, x_T and M_{yT} of the simulated abstract wind turbine model under the computed disturbances is less then 0.1% of the respective reference data.

6.4 Results

Figure 9 shows scatter plots of the estimated errors $w(t_k) = [e^d_{\dot{\Omega}}(t_k), e^d_{\dot{x}_T}(t_k), e^d_{\ddot{x}_T}(t_k)]^T$ for different mean wind speeds of 8 m/s, 10 m/s and 12 m/s.

Judging by the distribution of the estimated errors, we conjecture that the abstract model has a similar performance w.r.t. representing the high fidelity model for different mean wind speeds. As we know from earlier validations, the abstract model tends to be less accurate for increasing wind speeds. This fact is also reflected in our results, as the absolute errors, notably $e^d_{\ddot{x}_T}(t_k)$, for wind speeds of 10 m/s increase (green circles in Figure 9). The results for wind speeds of 12 m/s do not show tendencies towards higher absolute disturbances, which might be due to the lower total number of reference states in the reference data set.

Estimating the overall quality of an open-loop model can be considered difficult as neither disturbances can be sampled nor reachable sets can be computed due to the lack of a controller that stabilizes the system. In order to get a rough estimate of the quality of the undisturbed abstract wind turbine model, we compute the maximal and minimal numerical derivatives from the reference high fidelity simulations by

$$\frac{dy}{dt}_{max} = \max_k \frac{y(t_{k+1}) - y(t_k)}{t_{k+1} - t_k} \tag{72a}$$

$$\frac{dy}{dt}_{min} = \min_k \frac{y(t_{k+1}) - y(t_k)}{t_{k+1} - t_k}. \tag{72b}$$

If disturbances are not significantly greater than the estimated dynamics derived from the reference data, the model should not be rejected in general. In Figure 9, we plot the box

that is spanned by the minimal and maximal reference derivatives with solid lines for each wind speed independently. One can observe that the disturbances $e^d_{\dot{x}_T}(t_k)$ and $e^d_{\ddot{x}_T}(t_k)$ are considerably outside or sometimes close to the borders of the box spanned by dy/dt_{max} and dy/dt_{min}. A reason for this is that the abstract model does not capture the dynamics of the tower deflection x_T of the reference data (cf. Figure 10 as an example for mean wind speeds of 8 m/s). Since the coefficient in the tower base fore-aft bending moment M_{yT} in equation (80) is larger for the tower deflection x_T, the disturbance $e^d_{\dot{x}_T}(t_k)$ has to compensate for the missing dynamics to achieve conformance in M_{yT}.

However, despite this observation, we conjecture that the abstract wind turbine model might be a good initial point to start model-based development since the disturbances of $e^d_{\dot{\Omega}}(t_k)$ are mostly centered in the reference derivative box. Furthermore, the rotor speed Ω of the undisturbed wind turbine model follows the reference high fidelity simulation quite closely (cf. Figure 10). Since the reference data is based on high fidelity simulations, it has to be ensured that the high fidelity simulation model can serve as a reference instead of measurement data. If changes in the tower deflection x_T do not occur with high frequency in real world wind turbines, the disturbances applied to the abstract wind turbine model might be considerably smaller.

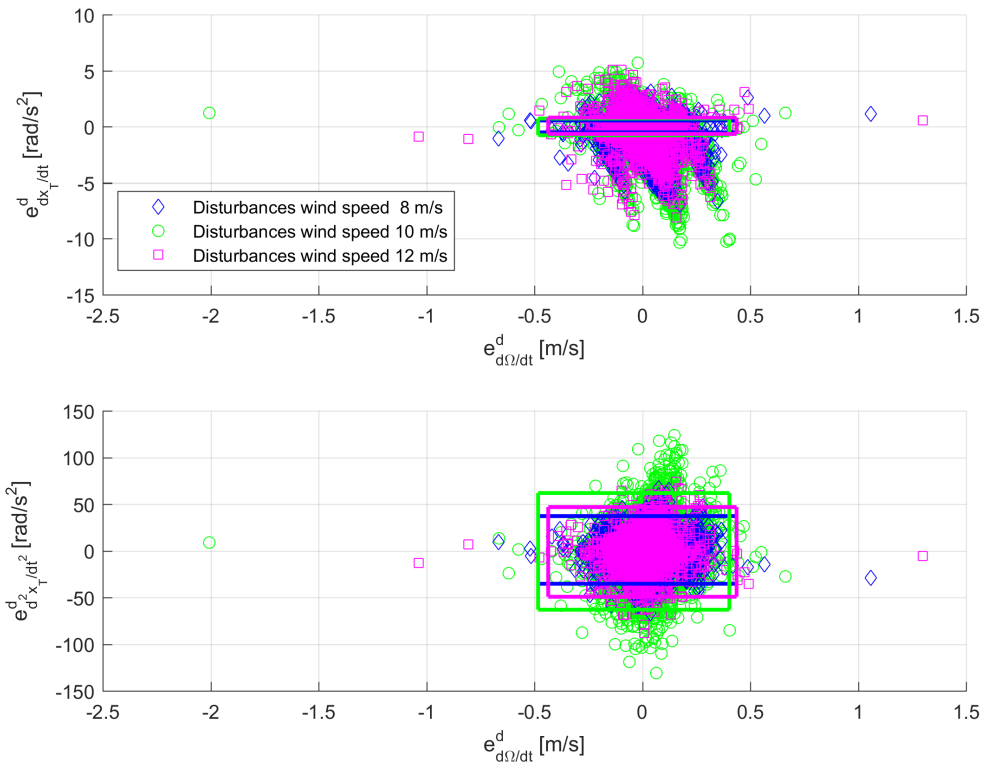


Figure 9: Scatter plot of the estimated errors $w(t_k) = [e^d_{\dot{\Omega}}(t_k), e^d_{\dot{x}_T}(t_k), e^d_{\ddot{x}_T}(t_k)]^T$ for different mean wind speeds

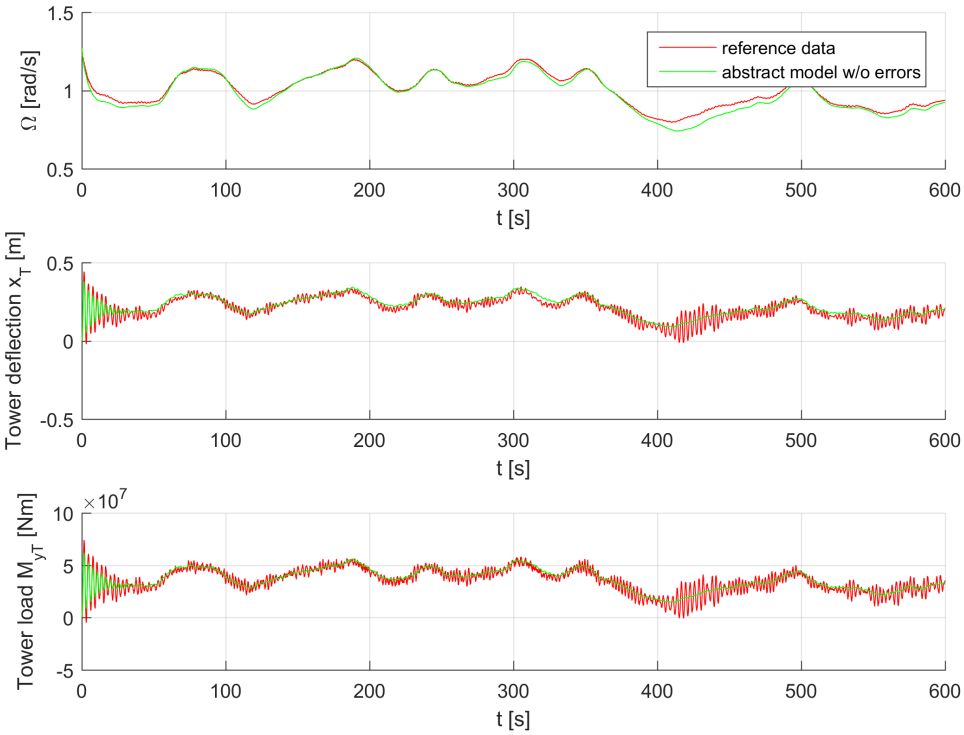


Figure 10: Overview of the desired conformant states/quantities Ω, x_T and M_{yT} based on simulation of the undisturbed wind turbine model (mean wind speeds of 8 m/s)

7 Use Case Robotics

As an additional use case, we applied conformance testing and reachability analysis to a mobile robot used within Bosch. The robot is supposed to operate safely in human populated environments, avoiding collisions with walking pedestrians while still maintaining a reasonable average velocity, even in dense crowds.

7.1 Use Case Overview

Mobile service robots often need to operate freely and flexibly in environments occupied by pedestrians. Because collisions could cause serious harm, particularly in settings with heavier robots, safety mechanisms always have to be considered. Despite the large body of work in path planning and obstacle avoidance [23, 27], in practice, most production robots still rely on hardware safety devices such as certified laser scanners. The main reason is that demonstrating the safety of software to the satisfaction of a safety body is difficult, and difficulty scales with the complexity of the algorithm.

In environments with none or only a few humans, a common way to reduce the problem is through a simple model of human motion: Either assume people will always stop (ISO 3691-4 [3]), or assume they always move at full speed (ISO 13855 [2] and ISO 13482 [4]). The latter is usually applied and results in a circular safety area around the robot.

Unfortunately, a circular field seriously restricts robot motion, including in areas which common sense would indicate as usable, such as beside or following a walking pedestrian. In more populated environments, it leads to frequent stopping of the robot and is therefore almost unusable. This opens up potential for the application of methods developed in UnCoVerCPS. Based on reachability analysis, a kinematicalliy accurate model of human motion can be employed to guarantee the same level of safety, allowing for more effifienct motion.

However, the application of reachability analysis to a safety critical scenario relies on conformant models. Within the scope of UnCoVerCPS, we concentratred on conformance testing for the pedestrian models, as this was perceived as the greater challenge by domain experts within Bosch.

7.2 Pedestrian Modeling

We model a single pedestrian as a point on a two-dimensional plane. The size of the pedestrian is then taken into account after the reachable set computation by enlarging the reachable sets accordingly. We assume that we can measure the pedestrian's position and velocity with

some known uncertainty. Also, we assume that the pedestrian performs a forward walking motion while possibly changing directions and that the pedestrian has a maximum speed and acceleration. We represent these constraints as two separate differential equation models: one constraining the acceleration and one constraining the velocity. Reachable states of the pedestrian are then states which are reachable under both models.

It would be possible to merge these two models into one that includes state constraints. This can be realized in CORA in a hybrid model, for which reachable sets are difficult to obtain. However, it has been shown in [7] that for reachability analysis it is possible to define multiple abstracting models, such that their reachable set intersection overapproximates the reachable sets of the hybrid model.

Therefore, we define the following two models. The acceleration-constrained model

$$\begin{aligned} \dot{p}_x &= v_x \\ \dot{p}_y &= v_y \\ \dot{v}_x &= a_x \\ \dot{v}_y &= a_y \\ \mathcal{U}_{ped}^{(a)} &= \left\{(a_x, a_y) \in \mathbb{R} \times \mathbb{R} | a_x^2 + a_y^2 \leq a_{max}^2\right\} \end{aligned} \tag{73}$$

has the two-dimensional position p and velocity v as its state variables. The input is a set representing all possible two-dimensional accelerations and bounded by a_{max}. The velocity-constrained model

$$\begin{aligned} \dot{p}_x &= v_x \\ \dot{p}_y &= v_y \\ \mathcal{U}_{ped}^{(v)} &= \left\{(v_x, v_y) \in \mathbb{R} \times \mathbb{R} | v_x^2 + v_y^2 \leq v_{max}^2\right\} \end{aligned} \tag{74}$$

has only the two-dimensional position p as its state variables while the velocity v is instead an input with a bound of v_{max}.

The initial position $[p_x(0), p_y(0)]$ and initial velocity $[v_x(0), v_y(0)]$ are assumed to lie in the sets $\mathcal{X}_{ped}^{(a),0}$ and $\mathcal{X}_{ped}^{(v),0}$, respectively. Since both models are used to predict possible pedestrian behavior, their initial states can be interpreted as the currently measured position and velocity of the pedestrian, plus some assumed measurement uncertainty.

The reachable sets of a single pedestrian are obtained by computing the reachable sets $\mathcal{R}_{ped}^{(a)}(t)$ and $\mathcal{R}_{ped}^{(v)}(t)$ (Fig. 11) of both models and then taking their intersection $\mathcal{R}_{ped}(t) = \mathcal{R}_{ped}^{(a)}(t) \cap \mathcal{R}_{ped}^{(v)}(t)$. Lastly, we enlarge all $\mathcal{R}_{ped}(t)$ by a circle in the (p_x, p_y)-dimensions to account for the shape of the human.

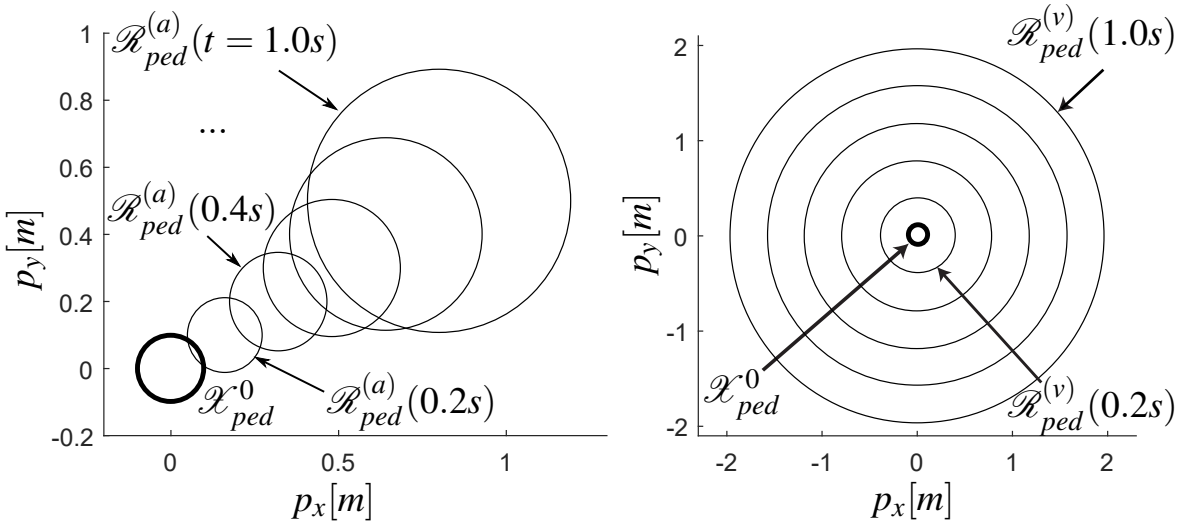


Figure 11: Reachable sets according to the acceleration-constrained (left) and velocity-constrained (right) model

7.3 Robot Modeling

For modeling the mobile robot we use a kinematic model of a differential-drive robot

$$\dot{p}_x = v_{tra}\cos(\phi)$$
$$\dot{p}_y = v_{tra}\sin(\phi)$$
$$\dot{\phi} = v_{rot}.$$

The initial state $[p_x(0), p_y(0), \phi(0)]^T$ represents the current pose of the robot and is bounded by an initial set $\mathcal{X}^0_{rob}$ that accounts for the inaccuracy of the robot's localization algorithm. The input of the system is the vector $[v_{tra}(t), v_{rot}(t)]^T$, consisting of the translational and rotational velocities of the differential drive, and is usually known exactly from the robot's control algorithm. In the same fashion as for the reachable sets of the pedestrians, we add the shape of the robot to the (p_x, p_y)-dimensions of all $\mathcal{R}_{rob}(t)$.

7.4 Conformance Testing

In the following, we present the results of evaluating our pedestrian model. We check whether the pedestrian model overapproximates the real behavior of a walking-only human by performing *reachset conformance testing* using ground truth pedestrian trajectories from a labeled video source of a street scene in Zurich, Switzerland [30] (see Fig. 12 for a screenshot). To be more precise, we test if trajectories lie inside the computed pedestrian reachable sets for time horizons of 1.6 s, which is larger than the maximally required braking time of the robot in our evaluation.

Figure 12: Screenshot of video source

For the pedestrian model, we parameterize $v_{max} = 2.0\,\mathrm{m/s}$ as suggested by [2], because it is the transition speed between walking and running. To set a_{max}, we apply numerical differentiation and filtering on the velocity data of the video source and then set $a_{max} = 0.6\,\mathrm{m/s^2}$ as an overapproximative value. The parameters of our model are shown in Tab. 9.

The results (Tab. 9) show good conformance results. However, there are some unsuccessful tests. A closer look at these failed cases reveals that the unsuccessful tests are caused by special pedestrian behavior such as changing directions too fast (12 cases), and velocities faster than v_{max} (229 cases). These behaviors lie outside of our initial assumptions, and we do not intend to cover them using our model.

This conformance test shows that our pedestrian model is reachset conformant to walking-only pedestrians. This pedestrian model can therefore be used for our verification approach if we are able to constrain human behavior to walking-only, which is possible in a closed environment setting, as in production plants. However, our model is not reachset conformant to all pedestrian behaviors. To have a more general model in the future, one may consider hybrid models switching to more conservative models, as suggested in [7], once the special cases above are detected.

Table 9: Pedestrian model and conformance test results

Pedestrian Model		**Conformance Test**	
Time horizon	1.6 s	Pedestrians	420
a_{max}	0.6 m/s^2	Gener. test cases	20084
v_{max}	2.0 m/s	Passed tests	19843
Ped. diameter	0.54 m	Rate	98.80 %
$\mathcal{X}^0_{ped}$: (p_x, p_y)-uncertain.	$\pm$ 0.1 m		
$\mathcal{X}^0_{ped}$: (v_x, v_y)-uncertain.	$\pm$ 0.1 m/s		

7.5 System Evaluation

In the the following section we present the results of the application of the UnCoVerCPS methodology (cf. automotive use-case) to our mobile service robots use case. We evaluate the performance of the online verification in an ROS[22] simulation for different scenarios where the robot has to navigate in the presence of pedestrians.

We compare three approaches with different obstacle models. First, we use our approach (UnCoverCPS methodology similar to the automotive use-case). Second, we consider an ISO13482-compliant safety field [4] with 360° warning and protective fields. The size of the safety field is fixed and dimensioned based on the maximum speed of the robot and the assumption that a pedestrian may approach the robot at full speed at any point in time. The third approach is based on the obstacle model used in braking ICS [12] and by Mitsch et al. [28]. This obstacle model assumes that obstacles may always move at full speed in any direction if we do not know their future behavior and requires that the robot is able to come to a rest before the obstacle may hit it. We refer to this approach as braking ICS in the following.

We execute our evaluation based on ROS Indigo. The physics simulation is carried out in Gazebo 7[23] and the robot uses a standard move base leveraging the Dynamic Window Approach (DWA, [17]). We use the default parameters from the Indigo release for the move base, except that we set the maximum velocity and acceleration for the differential drive robot to $v_{tra} = 1.5\,\mathrm{m/s}$, $v_{rot} = 2.0\,\mathrm{rad/s}$, $a_{tra} = 1.5\,\mathrm{m/s^2}$, and $a_{rot} = 1.0\,\mathrm{rad/s^2}$. Initially, the robot is stationary. The robot model is based on the Robotino models for Gazebo by RWTH Aachen[24], where we use its laser scanner for navigation and use the standard Planar Move Plugin to steer the robot.

The evaluations are carried out on a map that is illustrated in Fig. 13. The map is 24 m by 30 m from wall to wall and pedestrians walk continuously counter-clockwise along the green area. For obtaining realistic pedestrian motion, we simulate pedestrian motion in a dedicated Pedestrian Simulator[25] (PedSim) that is based on social forces. The simulated pedestrian positions are then transferred to Gazebo, while the robot position is also considered in PedSim such that the pedestrians react to the robot.

In our experiments, the robot will always move from top to bottom through the green area with different starting positions (cf. Fig. 13). Depending on the starting position, we create

[22] `http://www.ros.org`

[23] `http://gazebosim.org/`

[24] `https://git.fawkesrobotics.org/gazebo-models.git`

[25] `https://github.com/srl-freiburg/pedsim_ros`

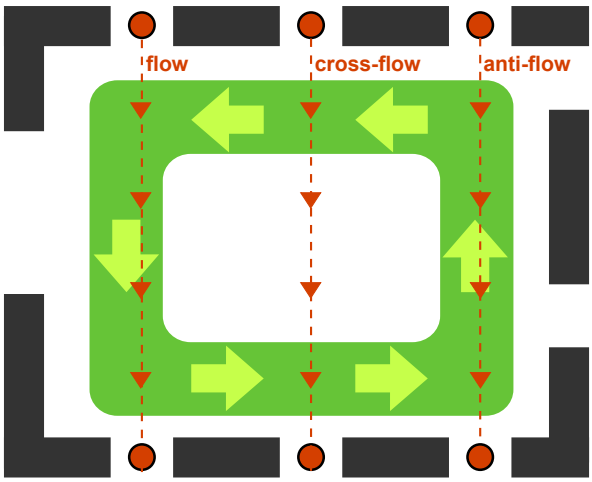


Figure 13: Illustration of the map used for the experiments

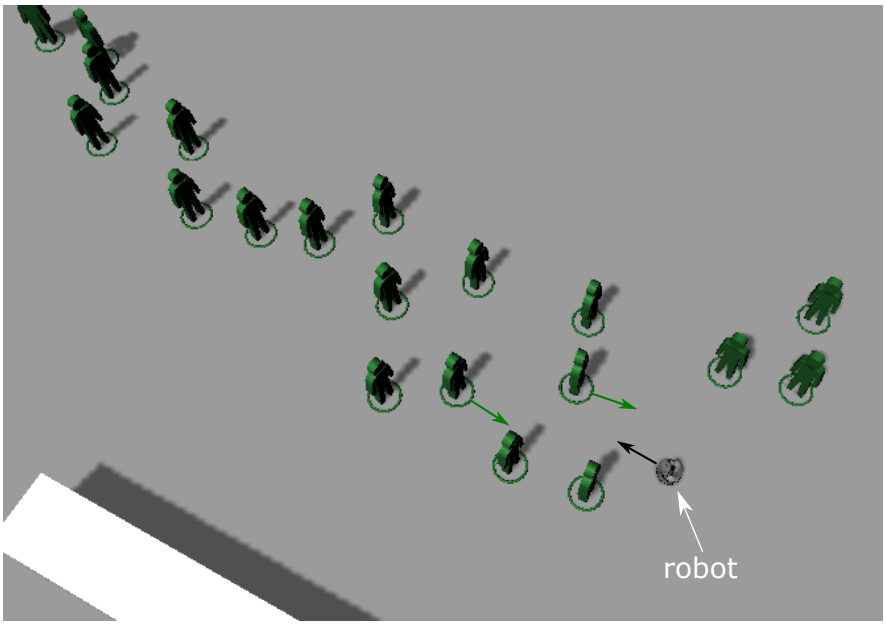


Figure 14: Gazebo screenshot for a dense antiflow scenario

three scenarios for encountering pedestrians: flow (move in same direction as pedestrians), cross-flow (pedestrians coming from left or right), and anti-flow (pedestrians approaching from front). In addition, we consider two pedestrian densities: light population and dense population. For light population, we place 25 pedestrians uniformly at random in the green area; for dense population, we distribute 60 pedestrians. We create 10 different placements of pedestrians for each density with a minimum distance of 0.5 m between any two pedestrians. The example in Fig. 14 shows a Gazebo screenshot for a typical situation in an anti-flow scenario with dense population.

For each scenario (flow, cross-flow, anti-flow), and for both light and dense populations, we generate 10 different pedestrian placements. All three approaches are executed on all of the

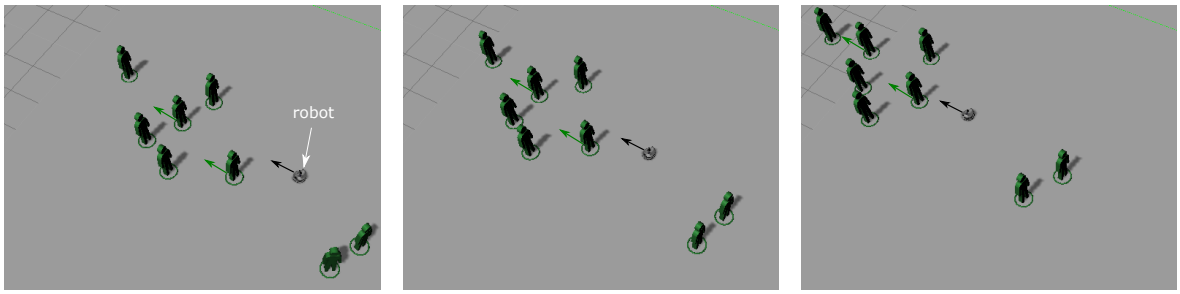


Figure 15: Gazebo screenshots from the same position with a time step of approx. 1 s for a light flow scenario where the robot uses the online verification approach

Table 10: Results from ROS Simulation (Lightly Populated Scenarios)

	Flow			Cross-flow			Anti-flow		
Approach	@Goal	Time(s)	Vel(m/s)	@Goal	Time(s)	Vel(m/s)	@Goal	Time(s)	Vel(m/s)
Braking ICS	10	34.3	0.73	10	40.0	0.63	10	116.1	0.21
Safety Field	10	37.9	0.64	10	35.7	0.68	10	74.8	0.32
Onl. Verif.	10	22.4	1.04	10	26.3	0.91	10	52.3	0.45

situations. Based on the collected data, we compute (1) whether the goal has been reached, (2) how long it took to reach the goal, (3) the average velocity, and (4) whether there were any unsafe collisions. An unsafe collision is one in which the robot's velocity is greater than 0.

The results are summarized in Table 10 for lightly populated situations and in Table 11 for the densely populated ones. Throughout our simulation runs, no unsafe collisions occurred for any of the approaches, so we omitted the corresponding column in the result tables. All values are arithmetic means over all runs.

The results clearly show that our method performs best in all cases by a large margin. Even in the simplest situation, motion with a lightly populated flow, our method is 1.4 times faster, and in the dense situation this even increases to a factor of 3.5. The example in Fig. 15 illustrates how the robot is able to follow a group of pedestrians in a flow scenario with light population when applying our online verification approach.

For dense population, our online verification method provides significant improvements in

Table 11: Results from ROS Simulation (Densely Populated Scenarios)

	Flow			Cross-flow			Anti-flow		
Approach	@Goal	Time(s)	Vel(m/s)	@Goal	Time(s)	Vel(m/s)	@Goal	Time(s)	Vel(m/s)
Braking ICS	10	108.0	0.25	10	114.2	0.21	10	519.5	0.05
Safety Field	10	96.0	0.27	10	76.9	0.31	10	251.5	0.10
Onl. Verif.	10	26.0	0.92	10	37.8	0.65	10	159.2	0.15

average velocity for the flow and cross-flow scenarios. For the anti-flow scenario, the online verification still enables an average velocity that is a factor 2 (safety field) or 3 (braking ICS) higher compared to the other approaches, but in absolute numbers an average velocity of 0.15 m/s still leaves significant room for improvement.

8 Conclusions

In this document we presented the UnCoVerCPS project's approaches for conformance testing of trace conformance and reachset conformance. Furthermore, we evaluated the applicability of our approaches on different use cases.

For the automated driving use case we tested trace conformance and reachset conformance for different vehicle models. Thereby, our studies were based on recorded measurement data of a real vehicle and reference simulations of a high dimensional multi-body model. While the conformance testing results with the multi-body simulation model as a reference are promising, our results show that modeling of a real vehicle for formal verification is still a challenging problem. Here, especially black box components in the real vehicle hinder the exact modeling of observed effects.

Concerning the wind turbine use case, we tested trace conformance based on high fidelity simulations as a reference. The results show that the presented abstract wind-turbine model can be used as a good starting point for model-based development.

We concluded this document by reachset conformance testing a pedestrian model in order to leverage the resulting conformant model for collision avoidance in path planning. Leveraging the UnCoVerCPS methodology, we were able to achieve a considerable reduction of the time it takes a robot to reach its target in densely populated scenarios.

Appendix A: Measurement Campaigns DLR Vehicle

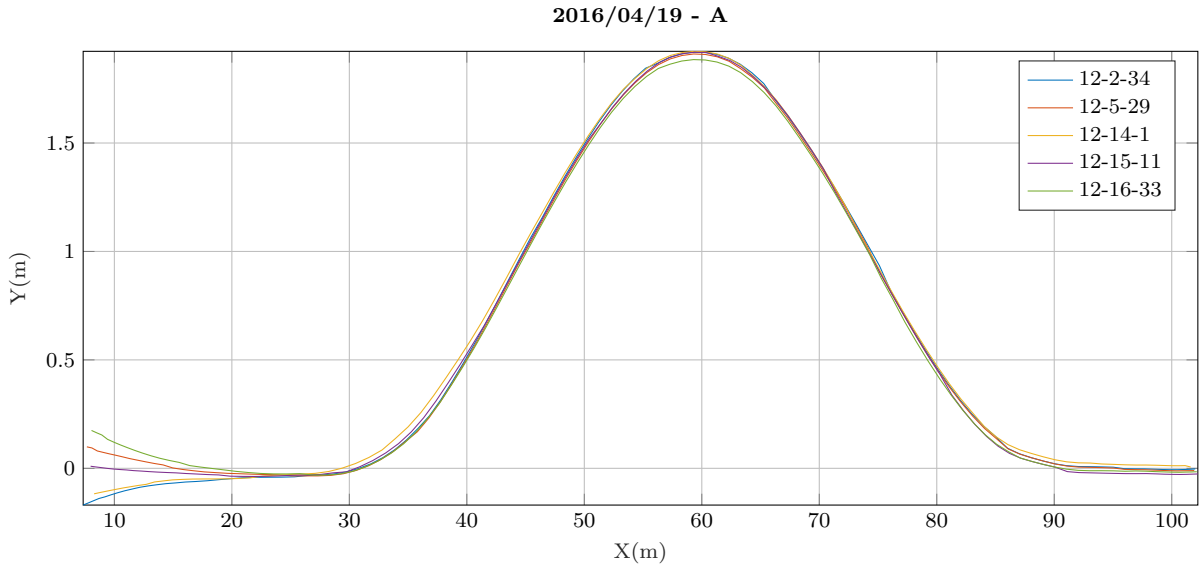


Figure 16: Swerve maneuver recorded with FASCarII at $v_x = 10m/s$. (The legend gives the time of recording of each trace, the figure title contains the date, thus allowing to identfy the corresponding log file)

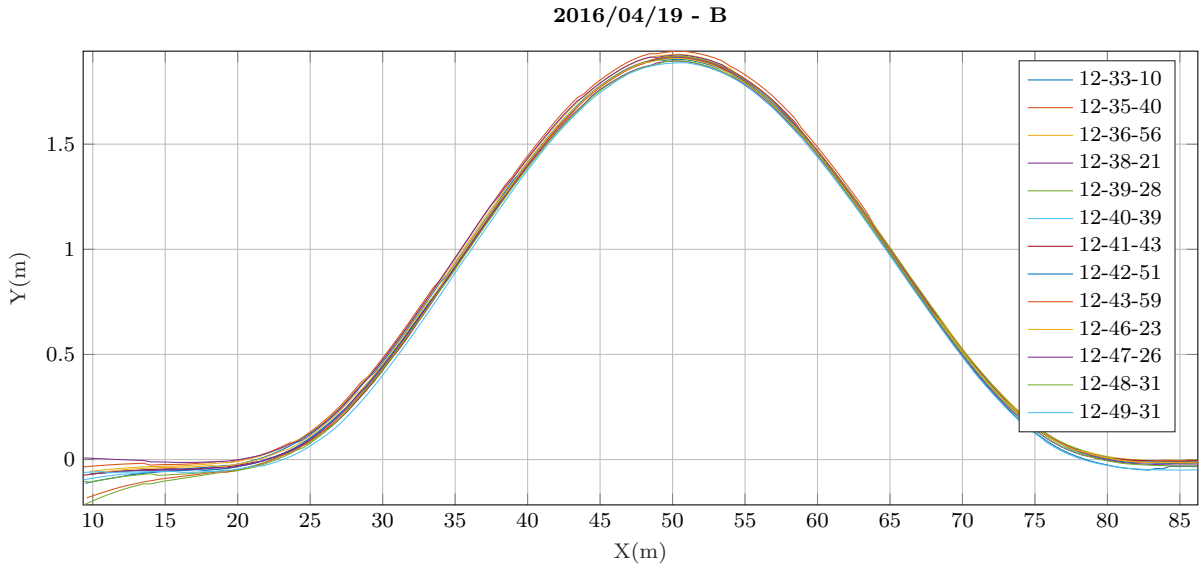


Figure 17: Swerve maneuver recorded with FASCarII at $v_x = 5m/s$.

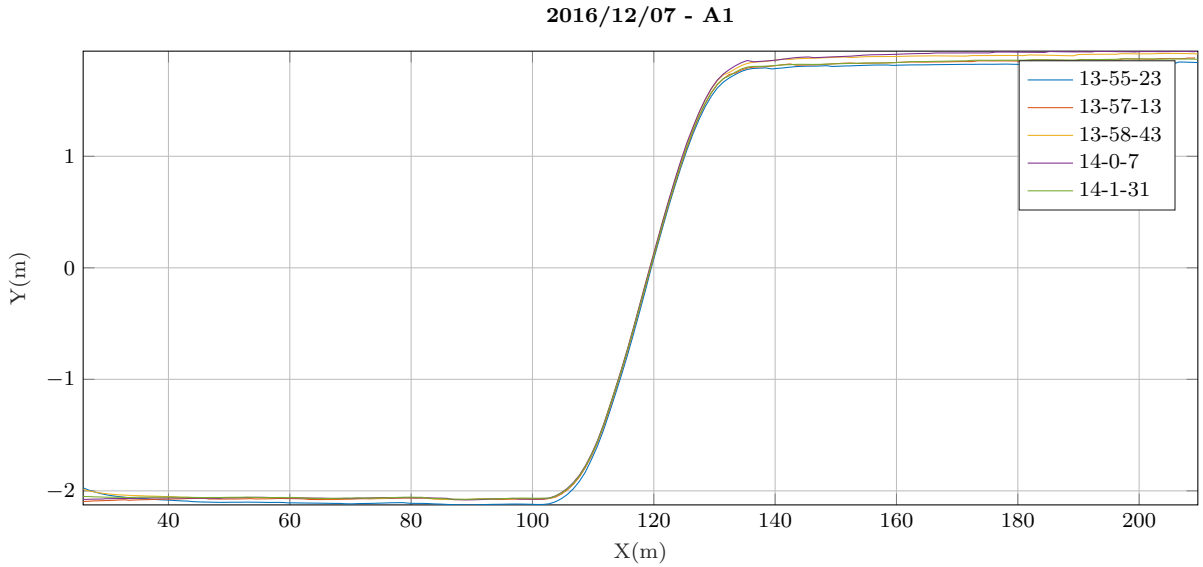


Figure 18: Lane-change maneuver recorded with FASCarII at $v_x = 10m/s$ and $\hat{a}_y = 2m/s^2$.

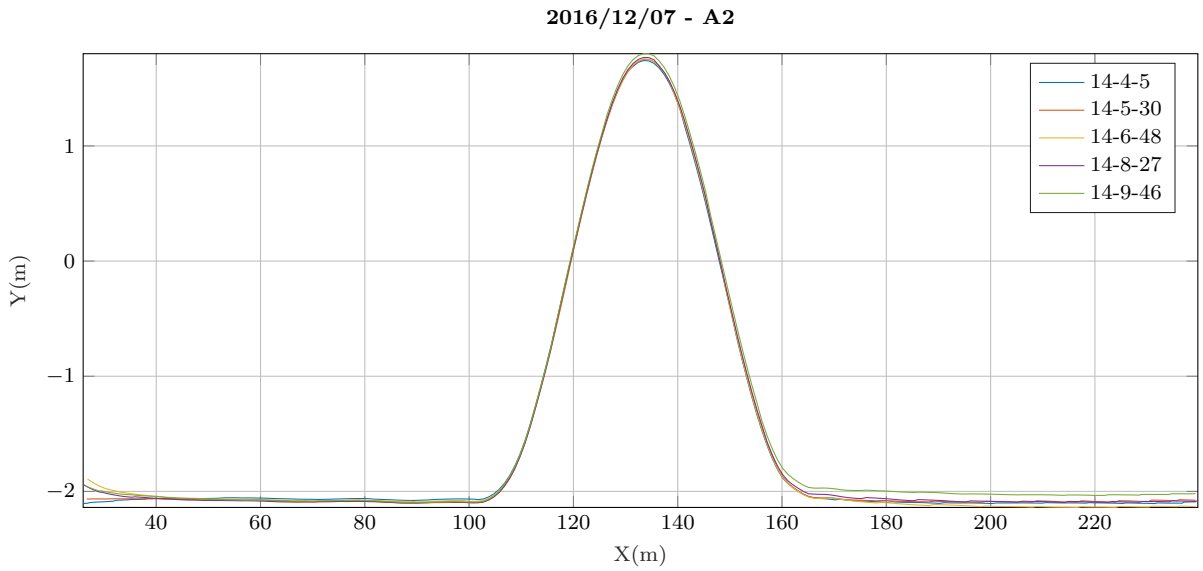


Figure 19: Swerve maneuver recorded with FASCarII at $v_x = 10m/s$ and $\hat{a}_y = 2m/s^2$.

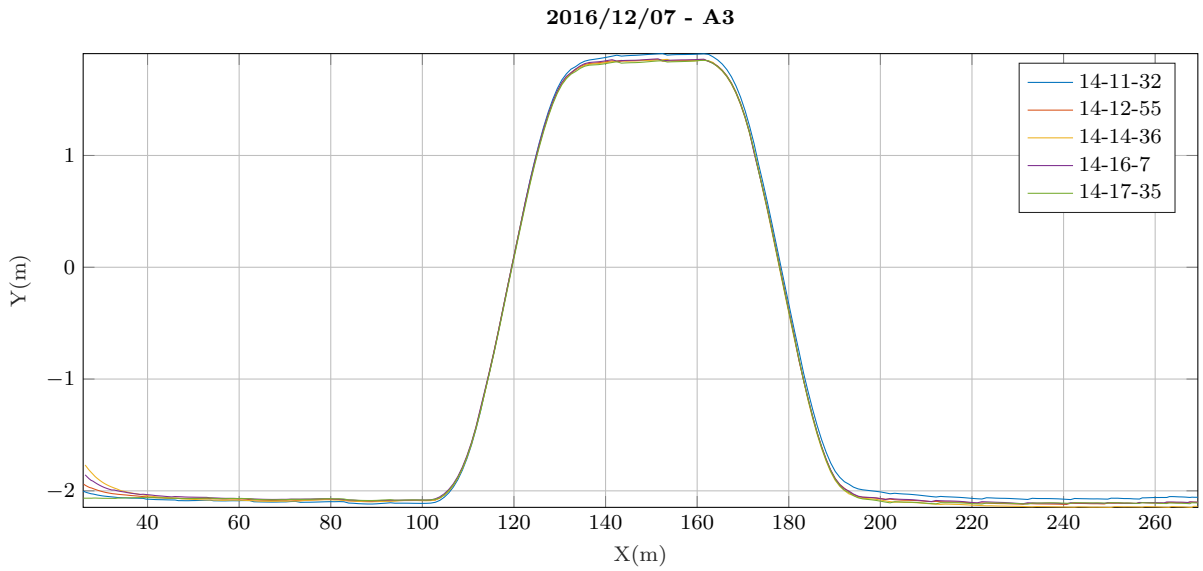


Figure 20: Double-lane-change maneuver recorded with FASCarII at $v_x = 10m/s$ and $\hat{a}_y = 2m/s^2$.

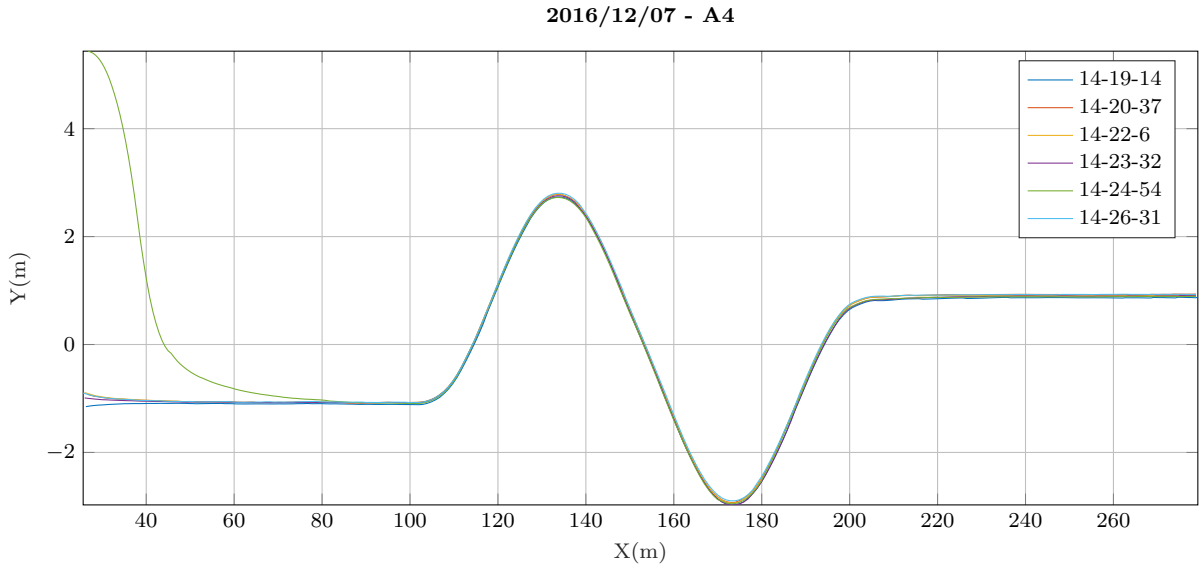


Figure 21: Slalom maneuver recorded with FASCarII at $v_x = 10m/s$ and $\hat{a}_y = 2m/s^2$.

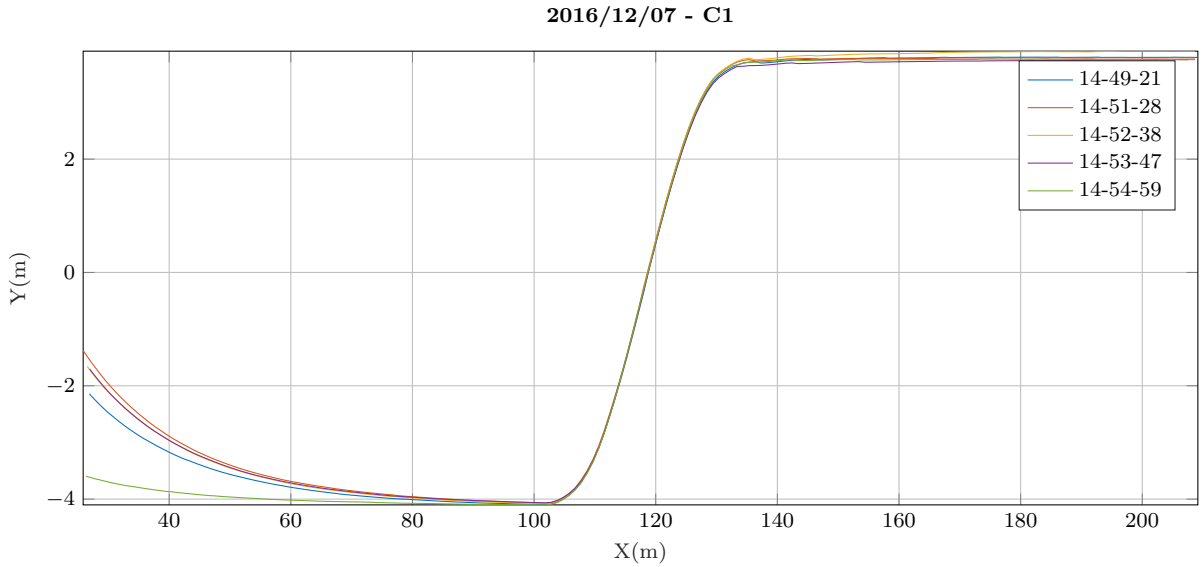


Figure 22: Lane-change maneuver recorded with FASCarII at $v_x = 10m/s$ and $\hat{a}_y = 4m/s^2$.

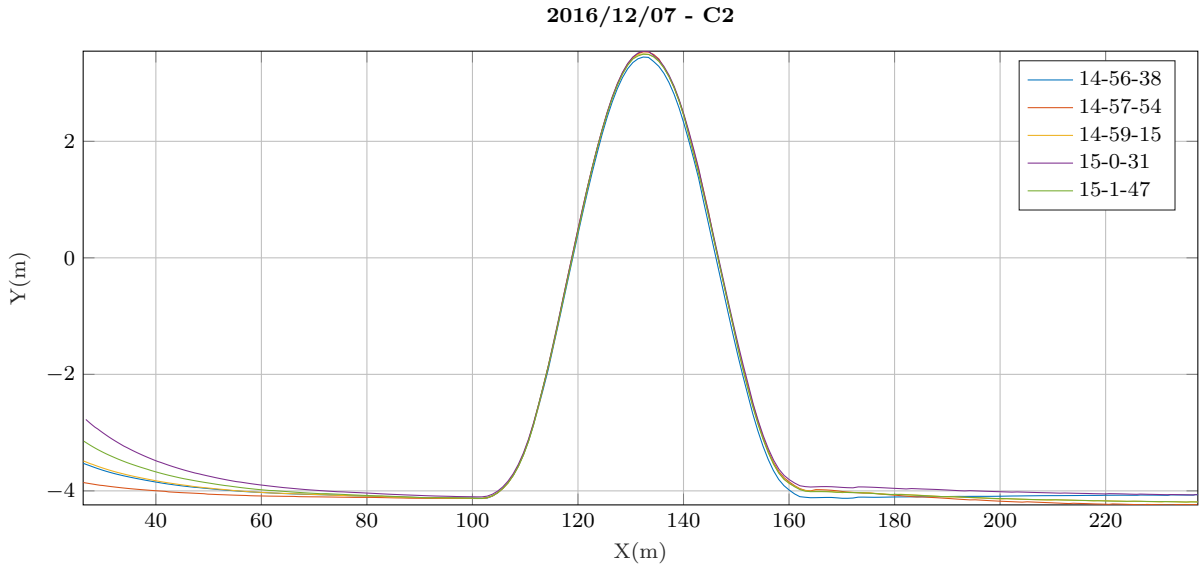


Figure 23: Swerve maneuver recorded with FASCarII at $v_x = 10m/s$ and $\hat{a}_y = 4m/s^2$.

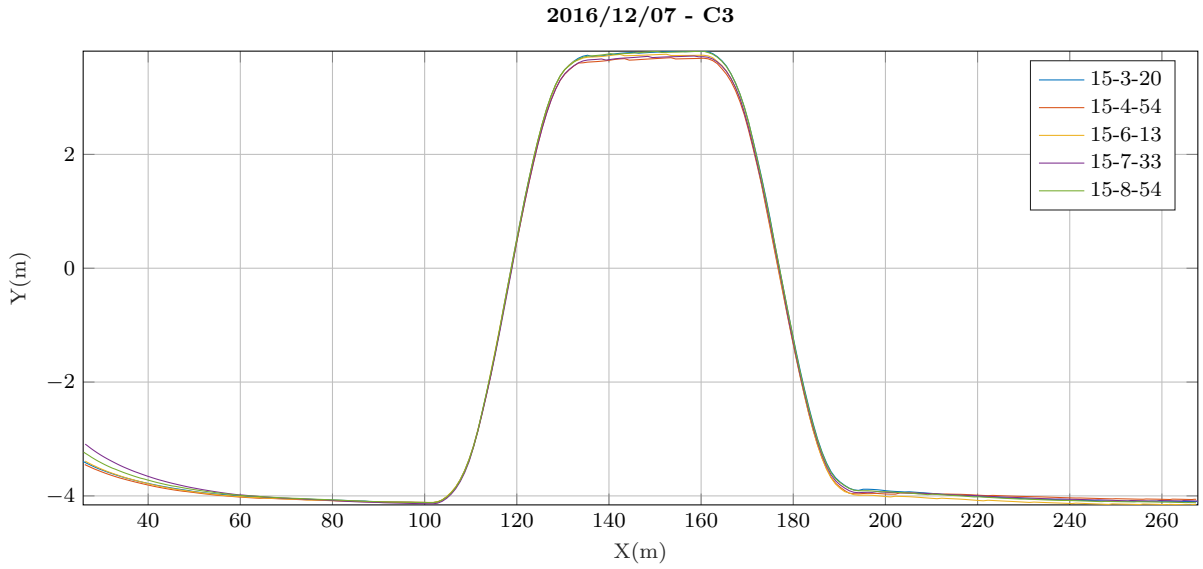


Figure 24: Double-lane-change maneuver recorded with FASCarII at $v_x = 10m/s$ and $\hat{a}_y = 4m/s^2$.

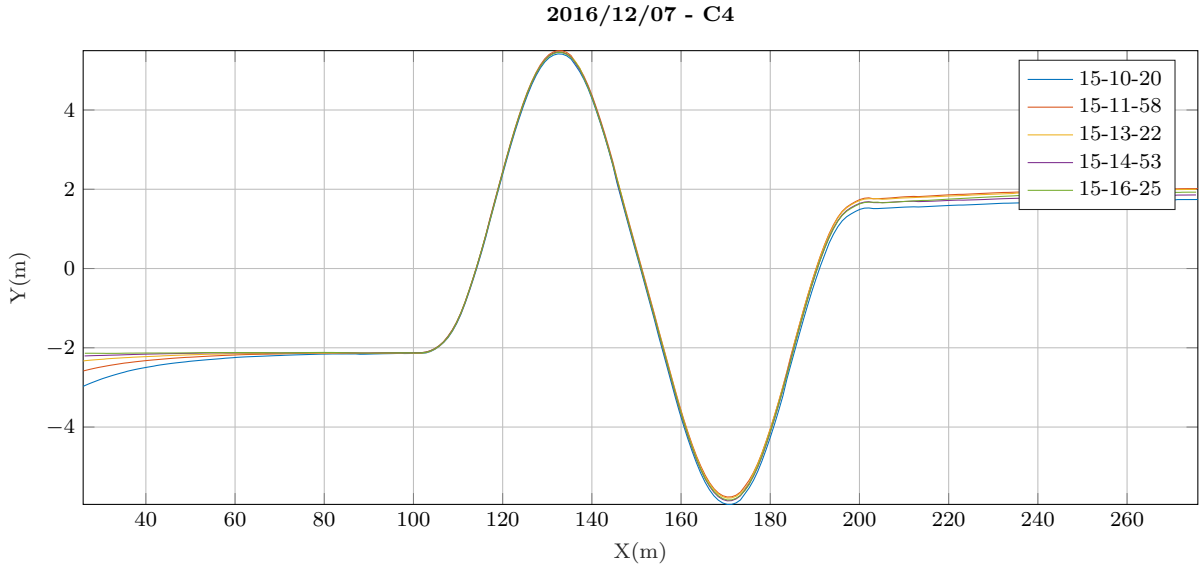


Figure 25: Slalom maneuver recorded with FASCarII at $v_x = 10m/s$ and $\hat{a}_y = 4m/s^2$.

Appendix B: Results DLR Vehicle Trace Conformance for f_B Model

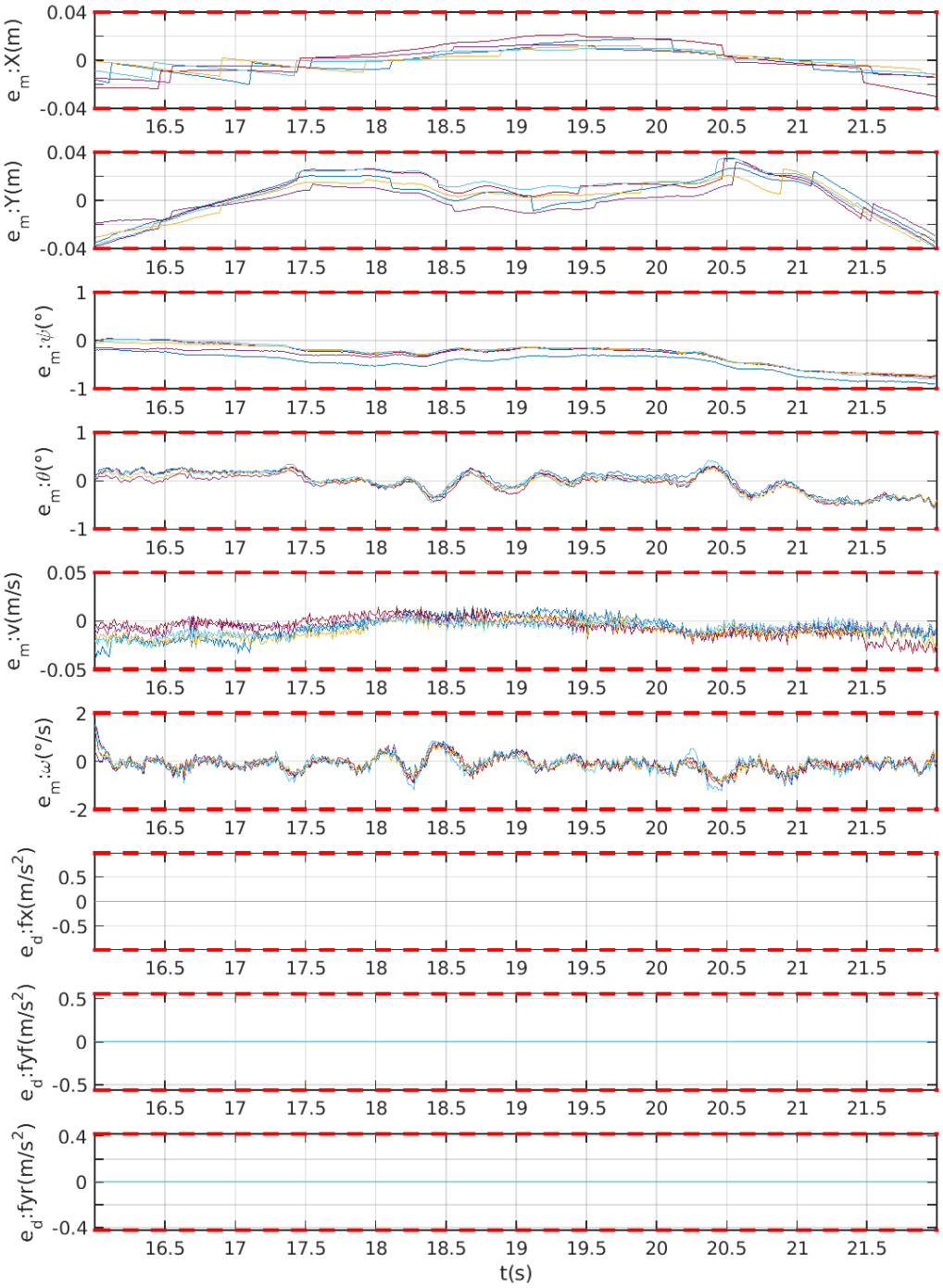


Figure 26: Trace-conformance for f_B model vs FASCar2, DecA1

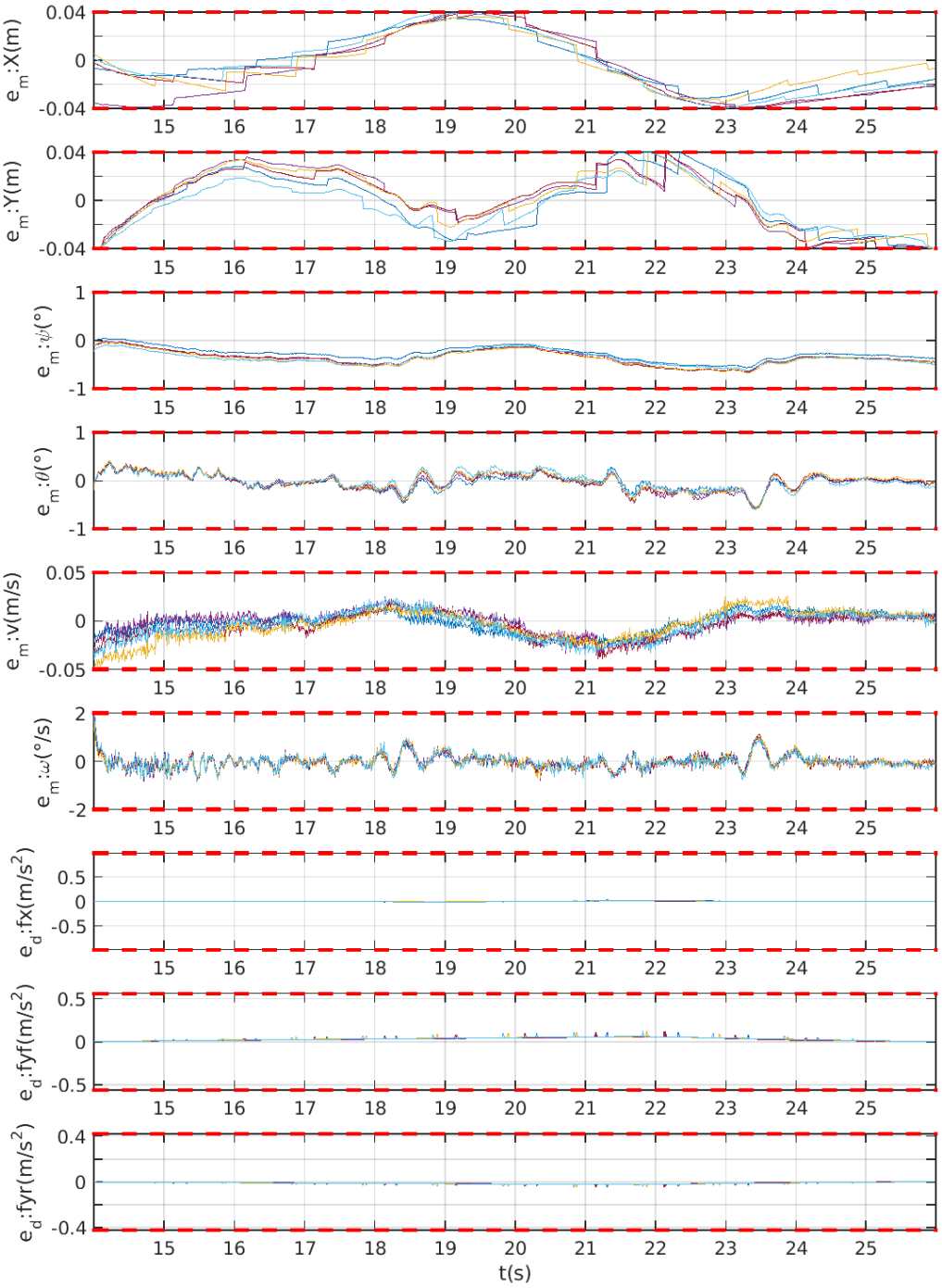


Figure 27: Trace-conformance for f_B model vs FASCar2, DecA2

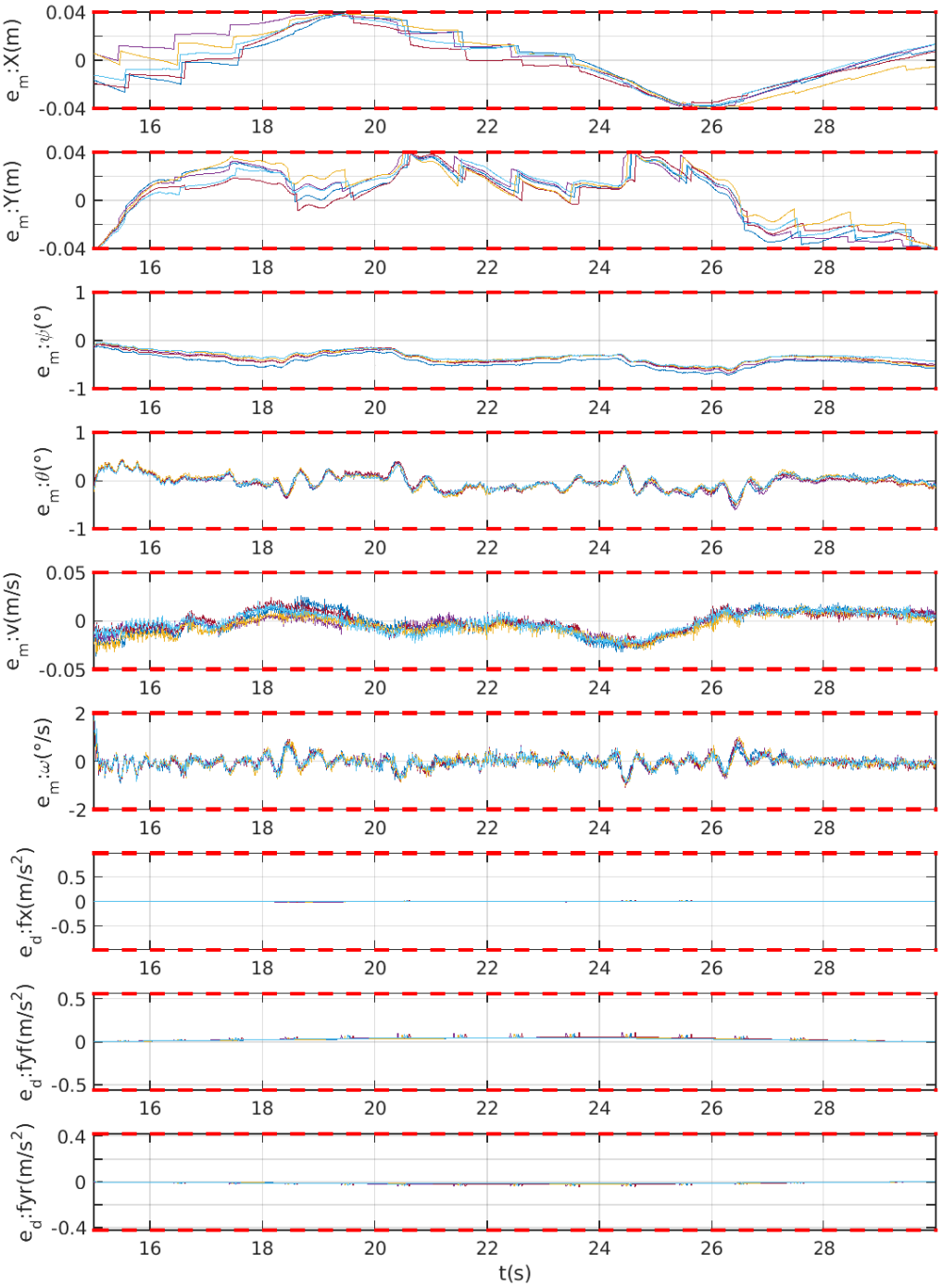


Figure 28: Trace-conformance for f_B model vs FASCar2, DecA3

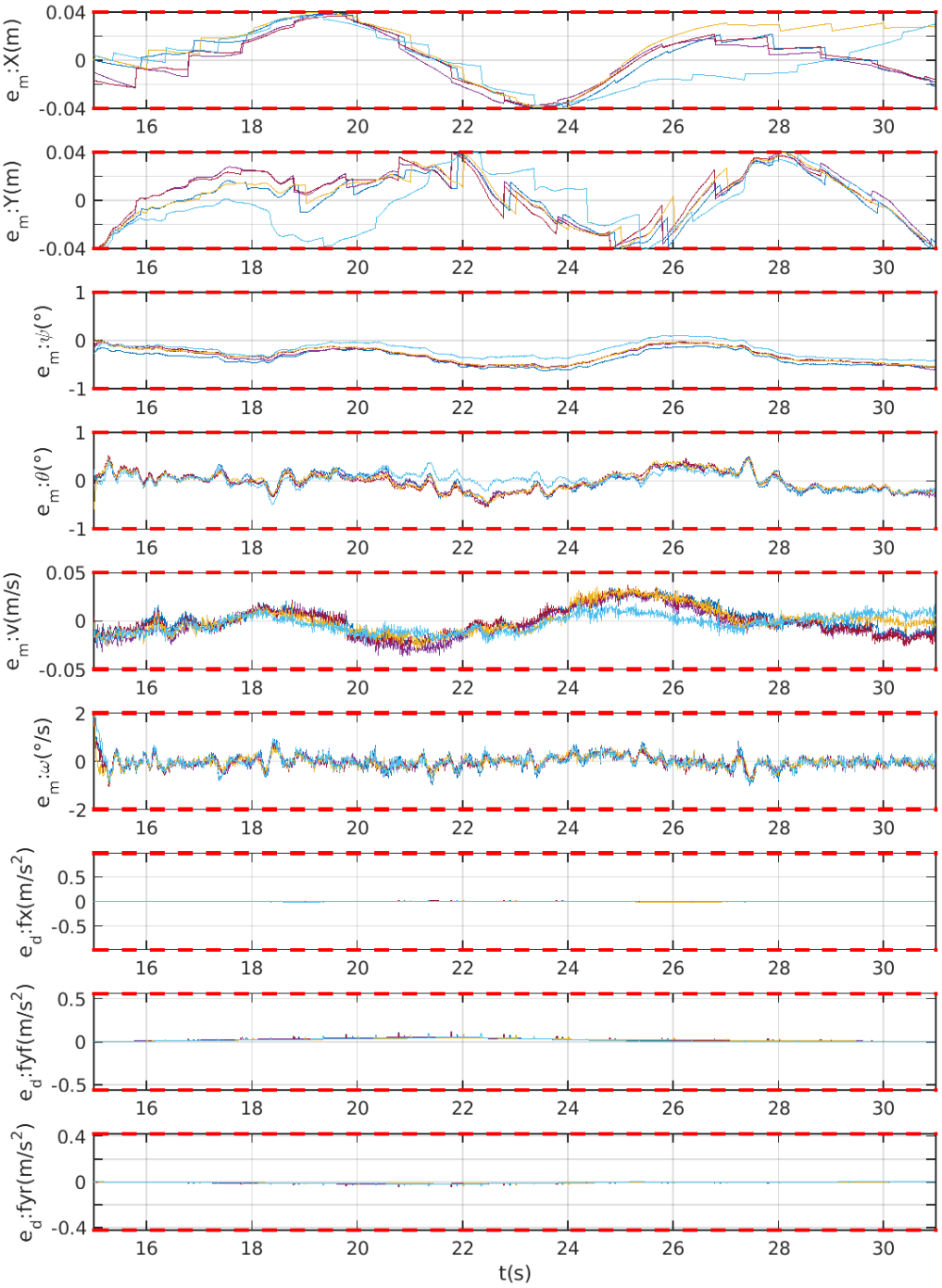


Figure 29: Trace-conformance for f_B model vs FASCar2, DecA4

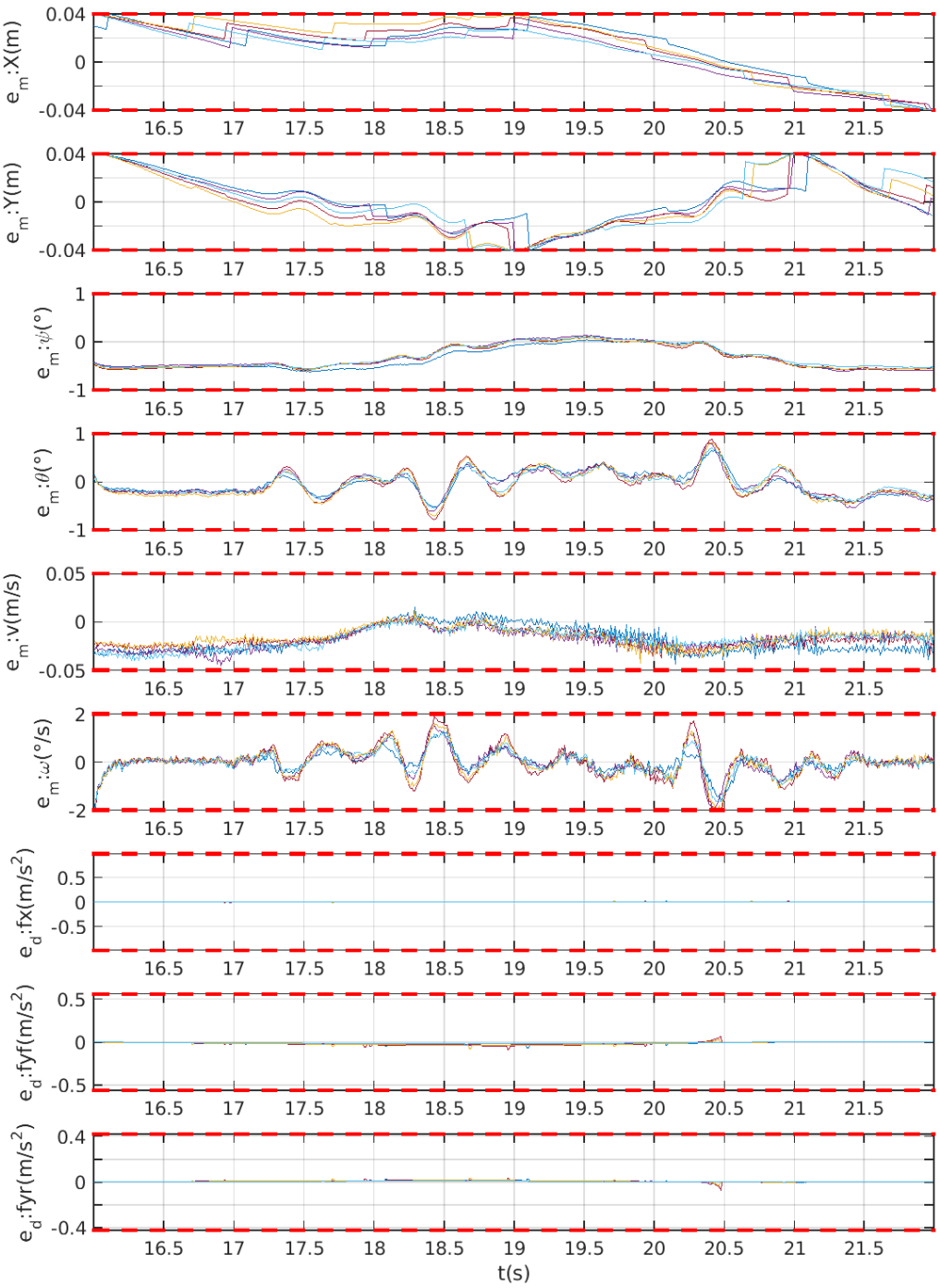


Figure 30: Trace-conformance for f_B model vs FASCar2, DecC1

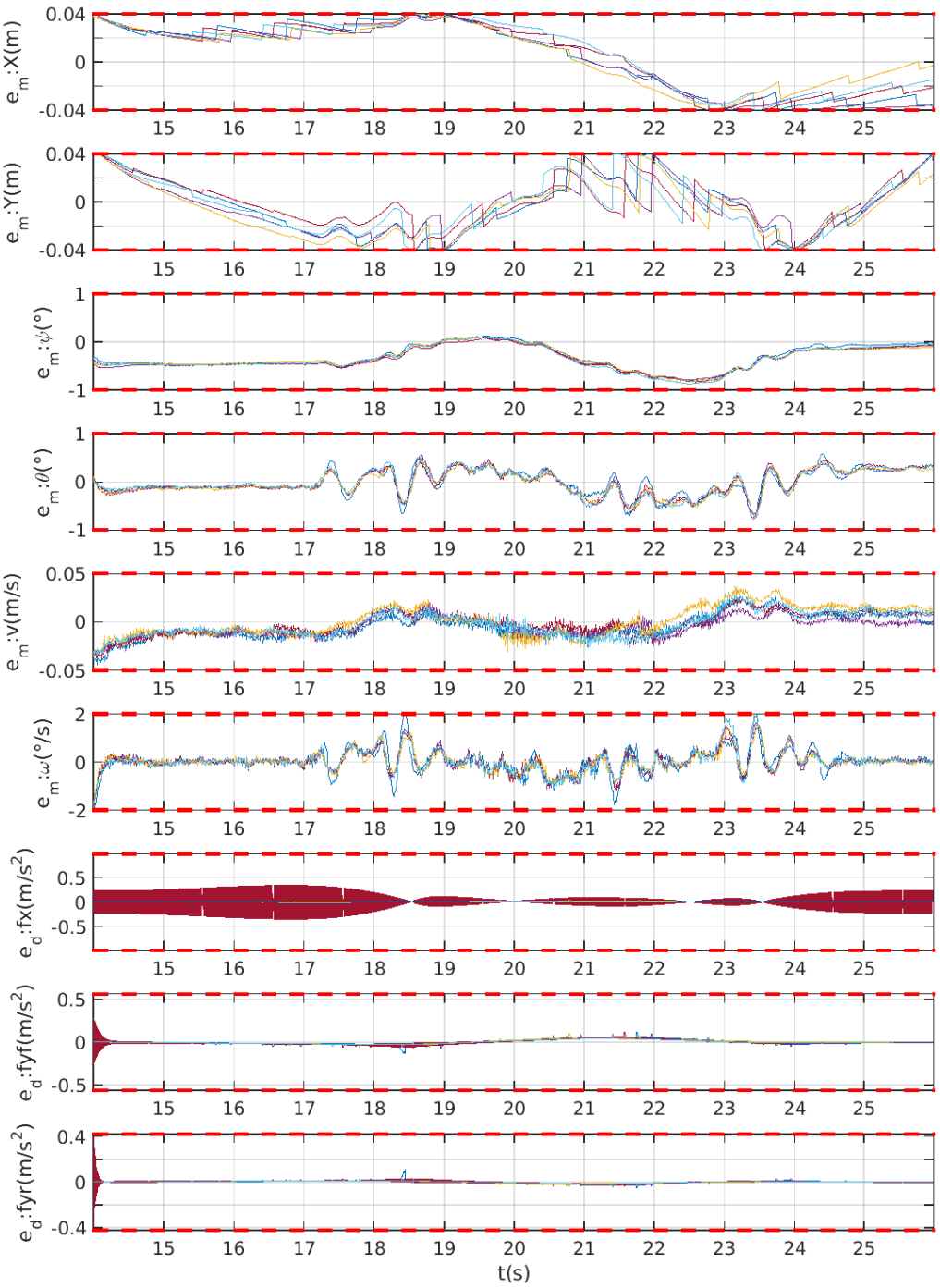


Figure 31: Trace-conformance for f_B model vs FASCar2, DecC2

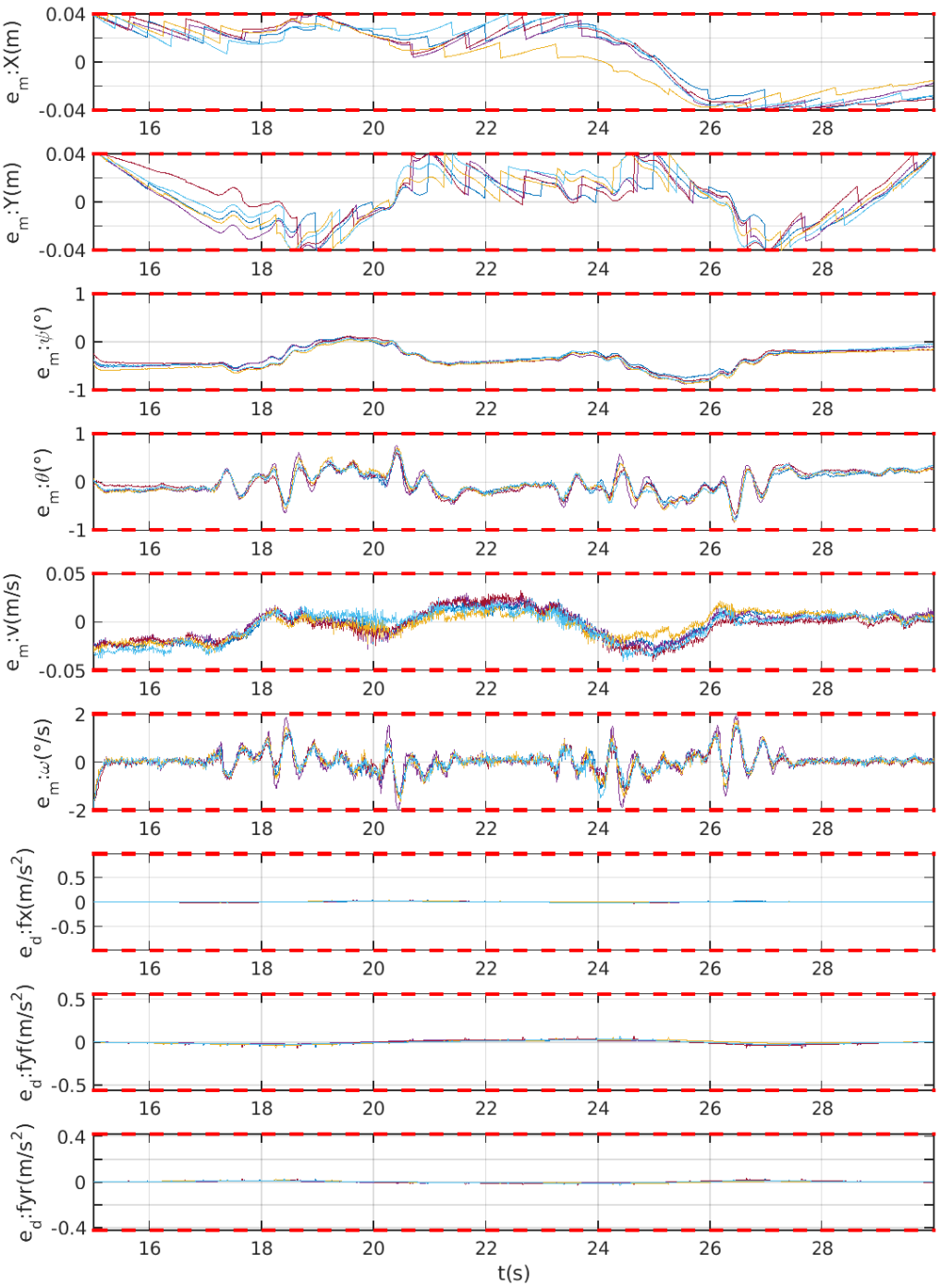


Figure 32: Trace-conformance for f_B model vs FASCar2, DecC3

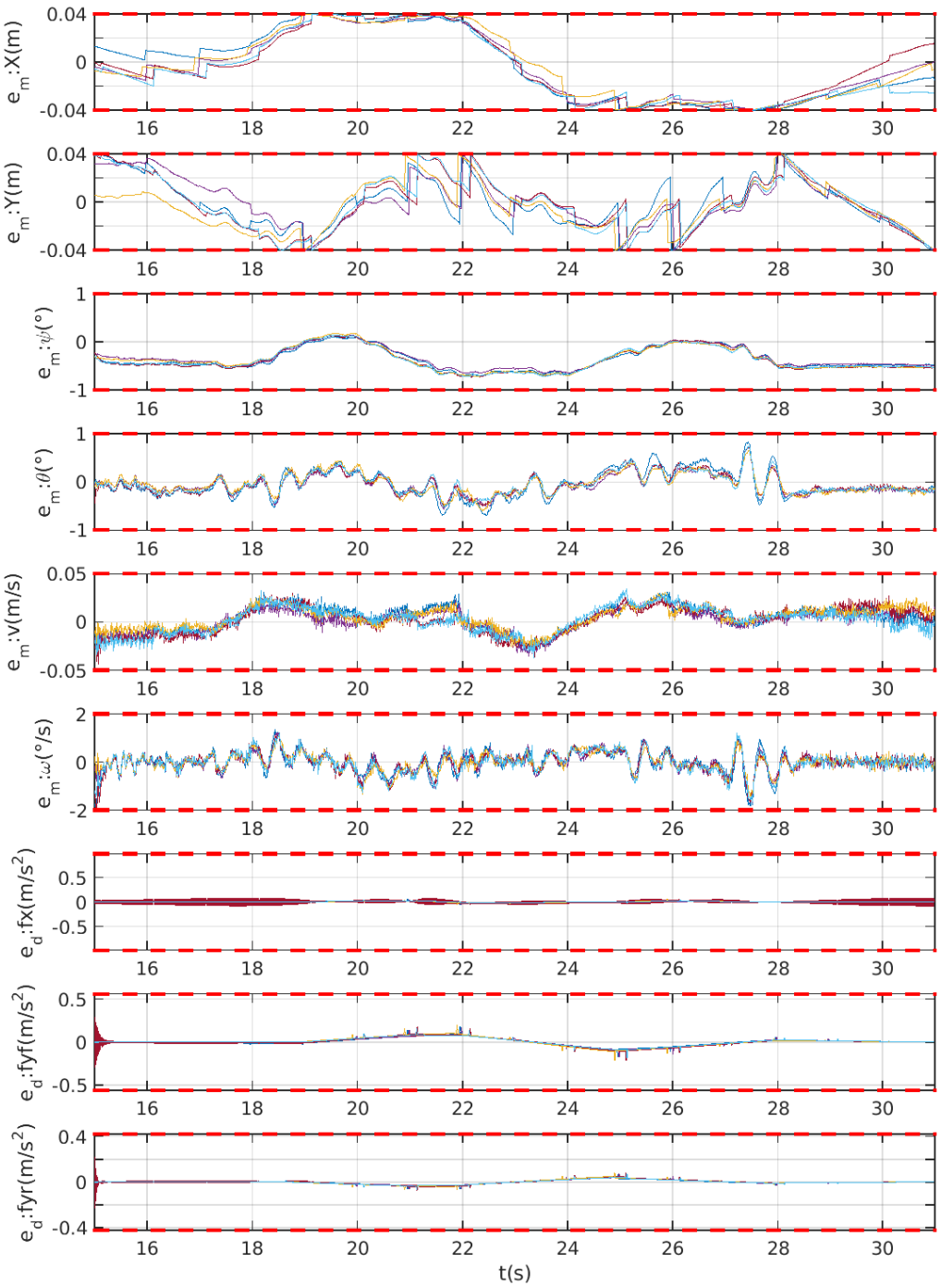


Figure 33: Trace-conformance for f_B model vs FASCar2, DecC4

Appendix C: Results DLR Vehicle Trace Conformance for f_{BX} Model

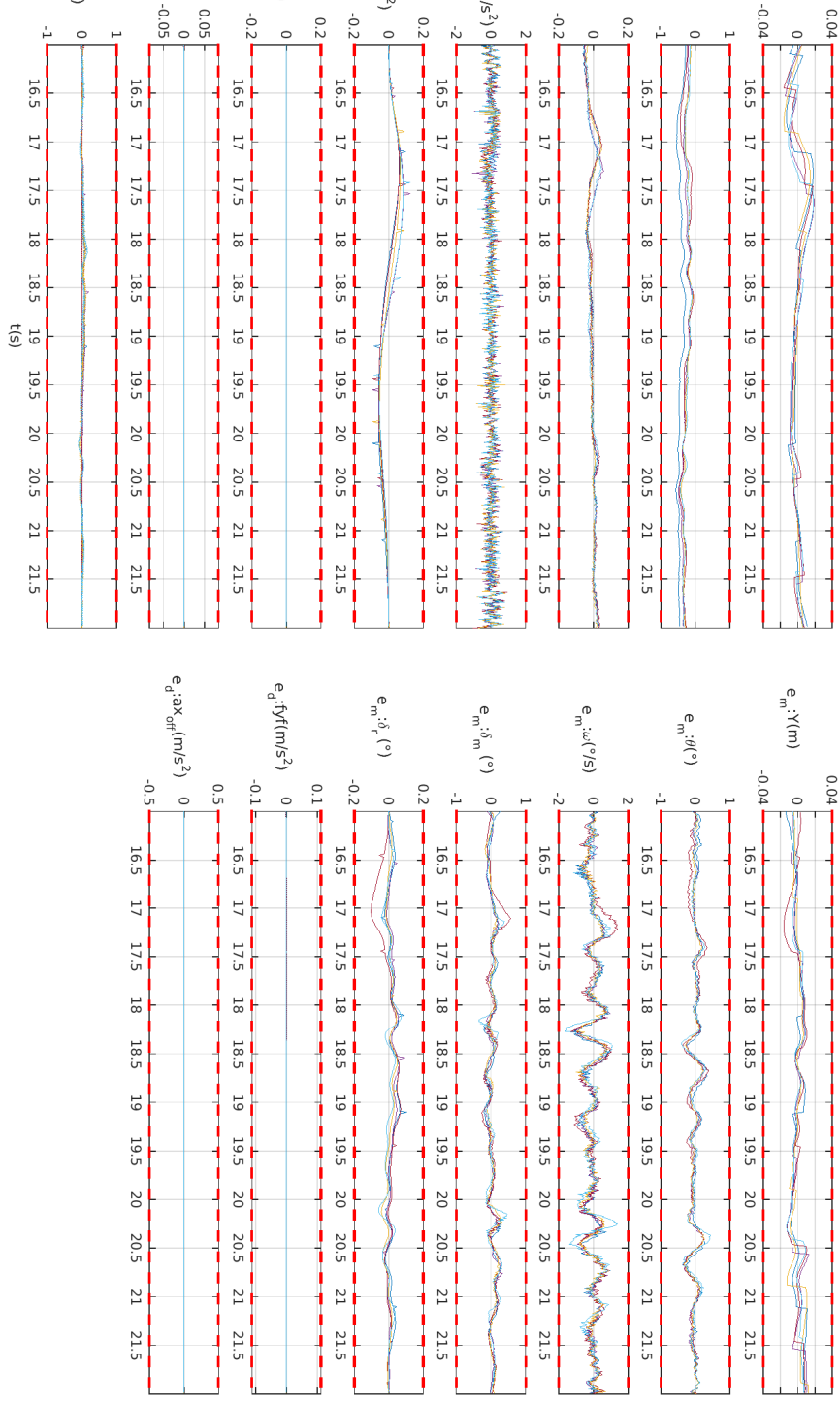


Figure 34: Trace-conformance for f_{BX} model vs FASCar2, DecA1

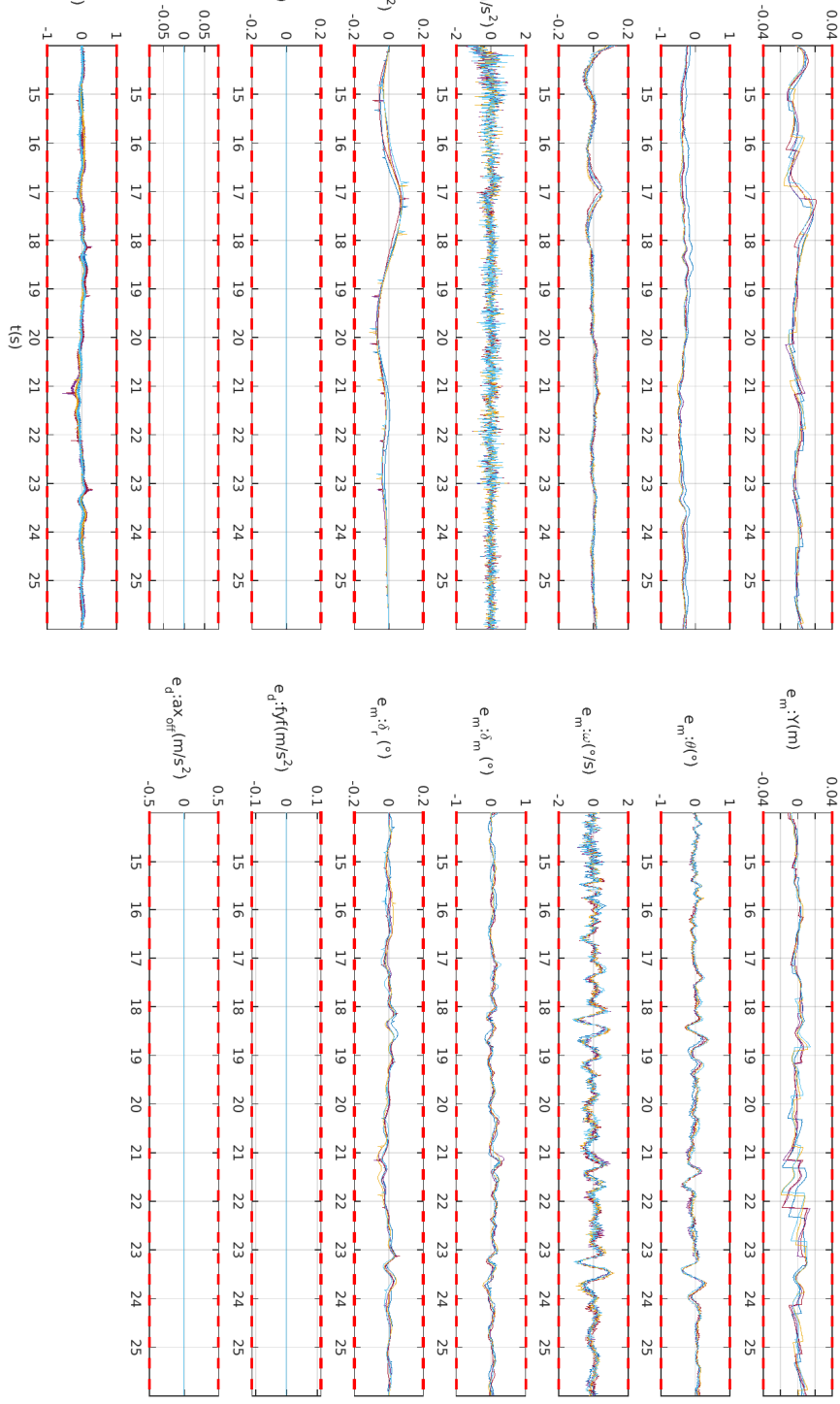


Figure 35: Trace-conformance for f_{BX} model vs FASCar2, DecA2

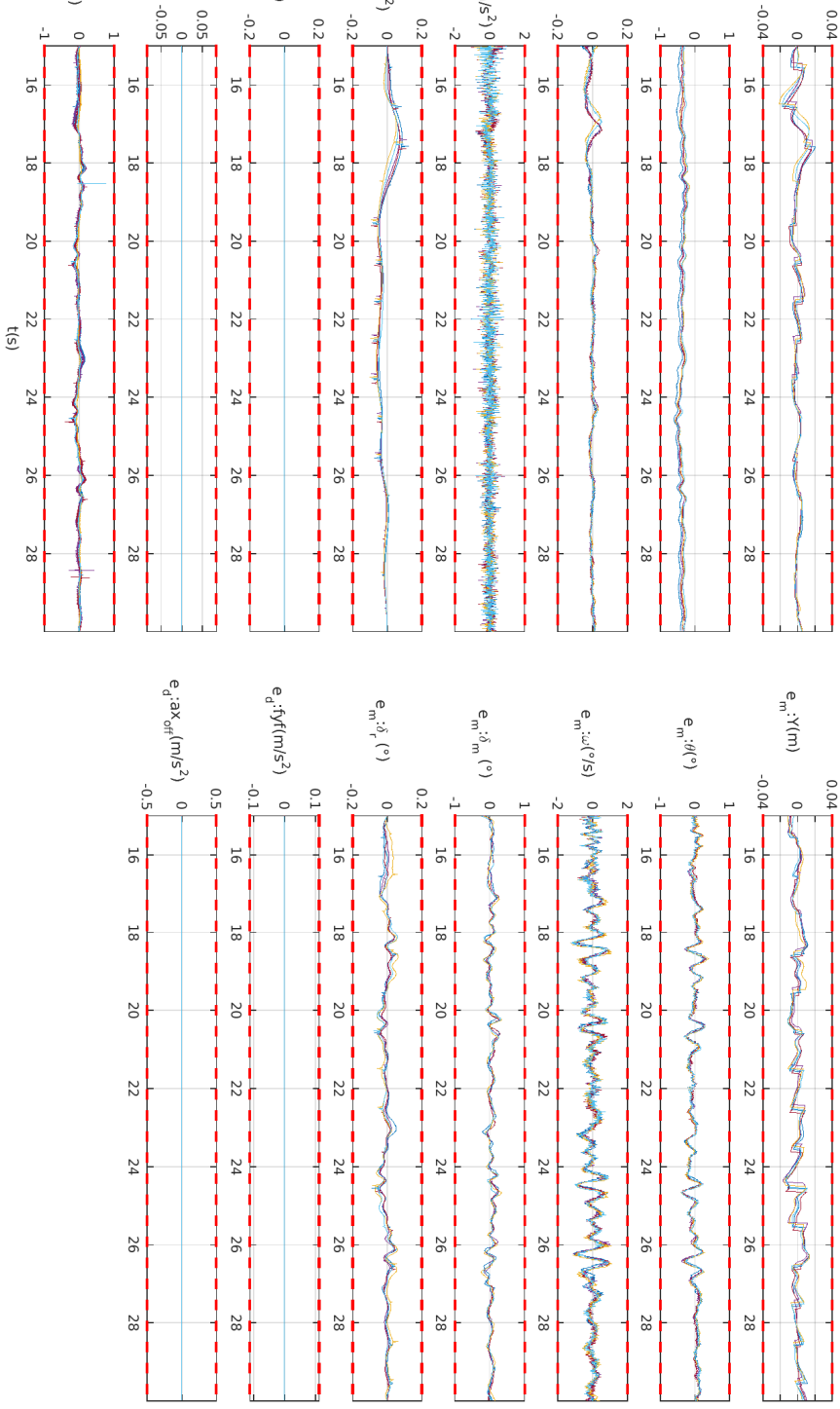


Figure 36: Trace-conformance for f_{BX} model vs FASCar2, DecA3

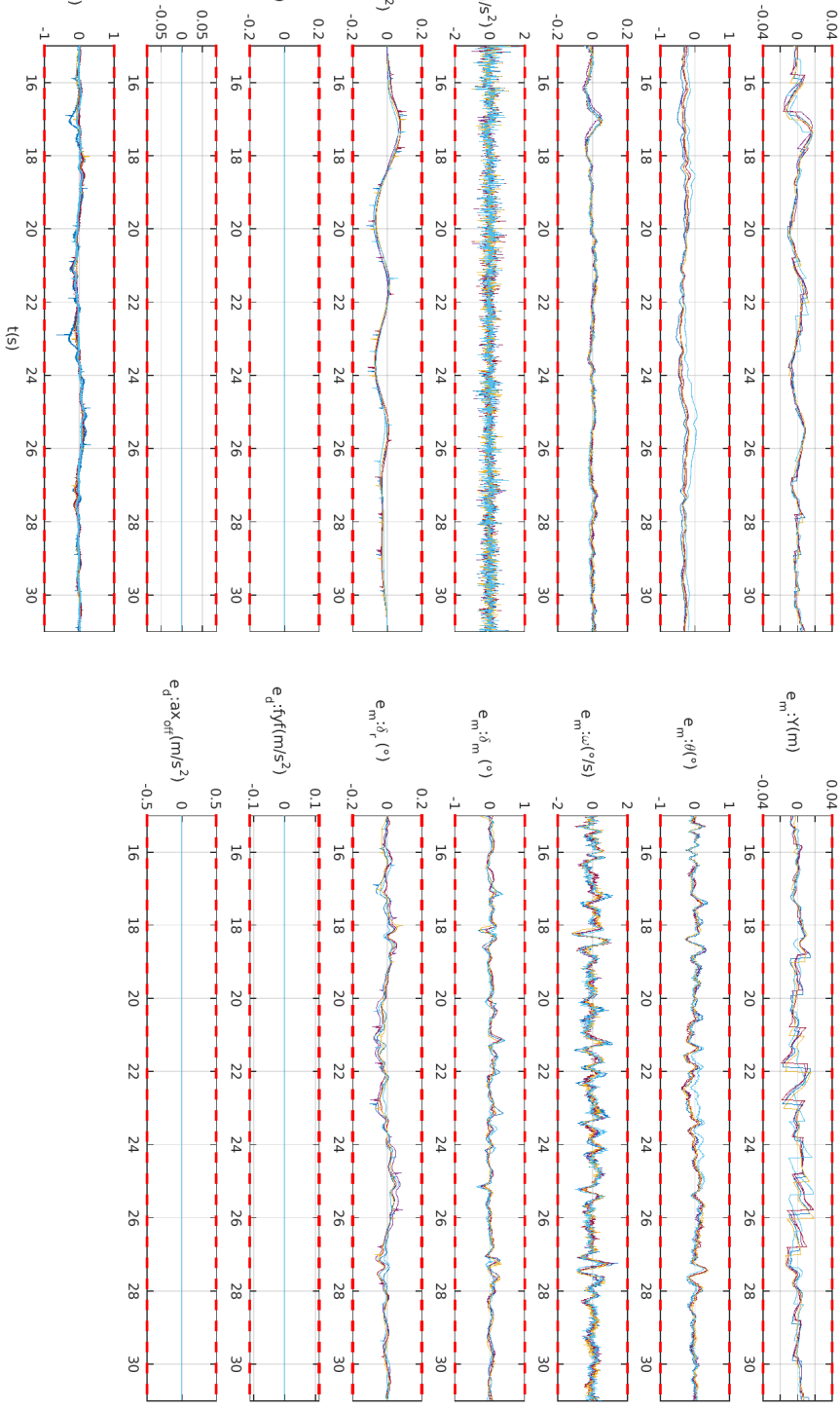


Figure 37: Trace-conformance for f_{BX} model vs FASCar2, DecA4

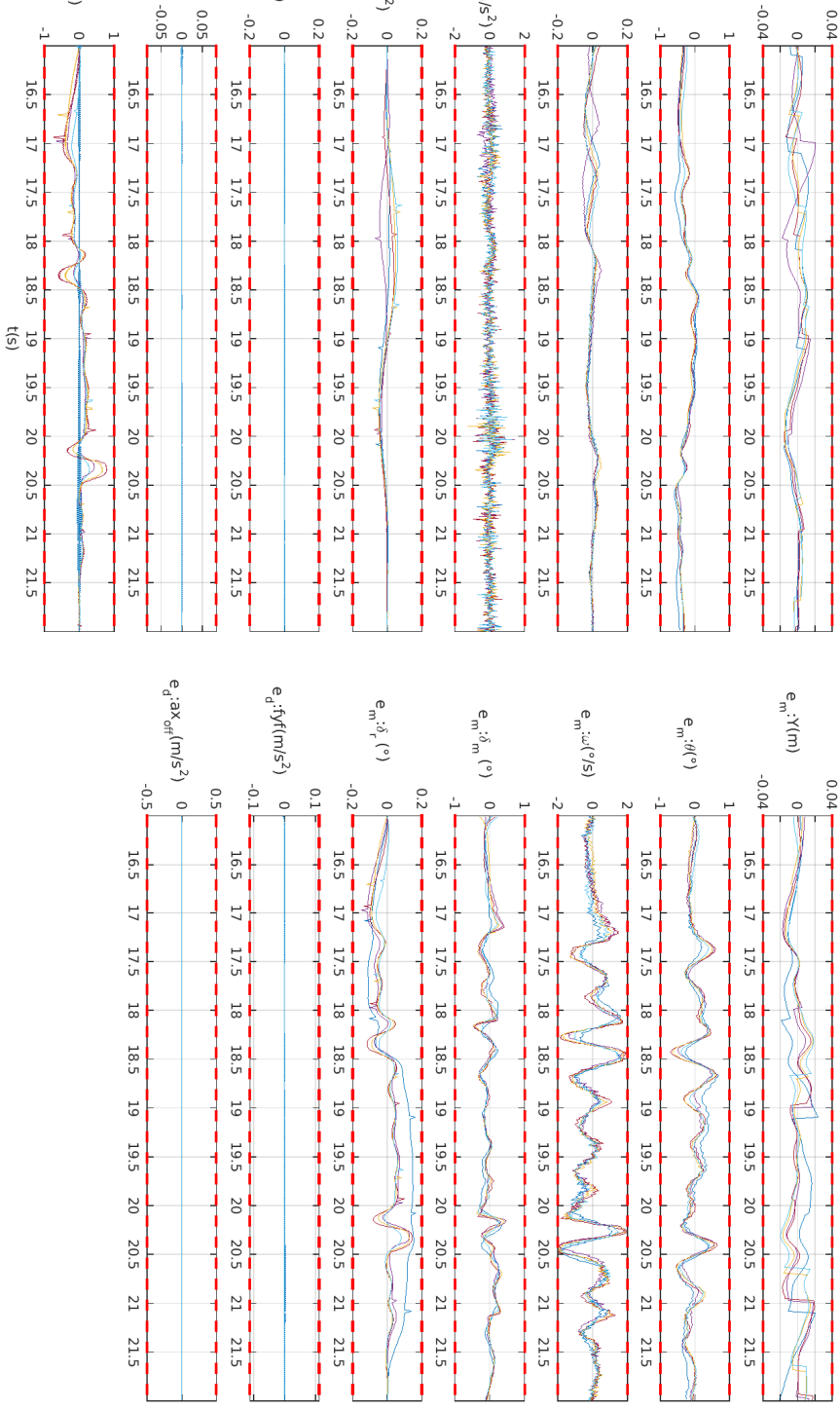


Figure 38: Trace-conformance for f_{BX} model vs FASCar2, DecC1

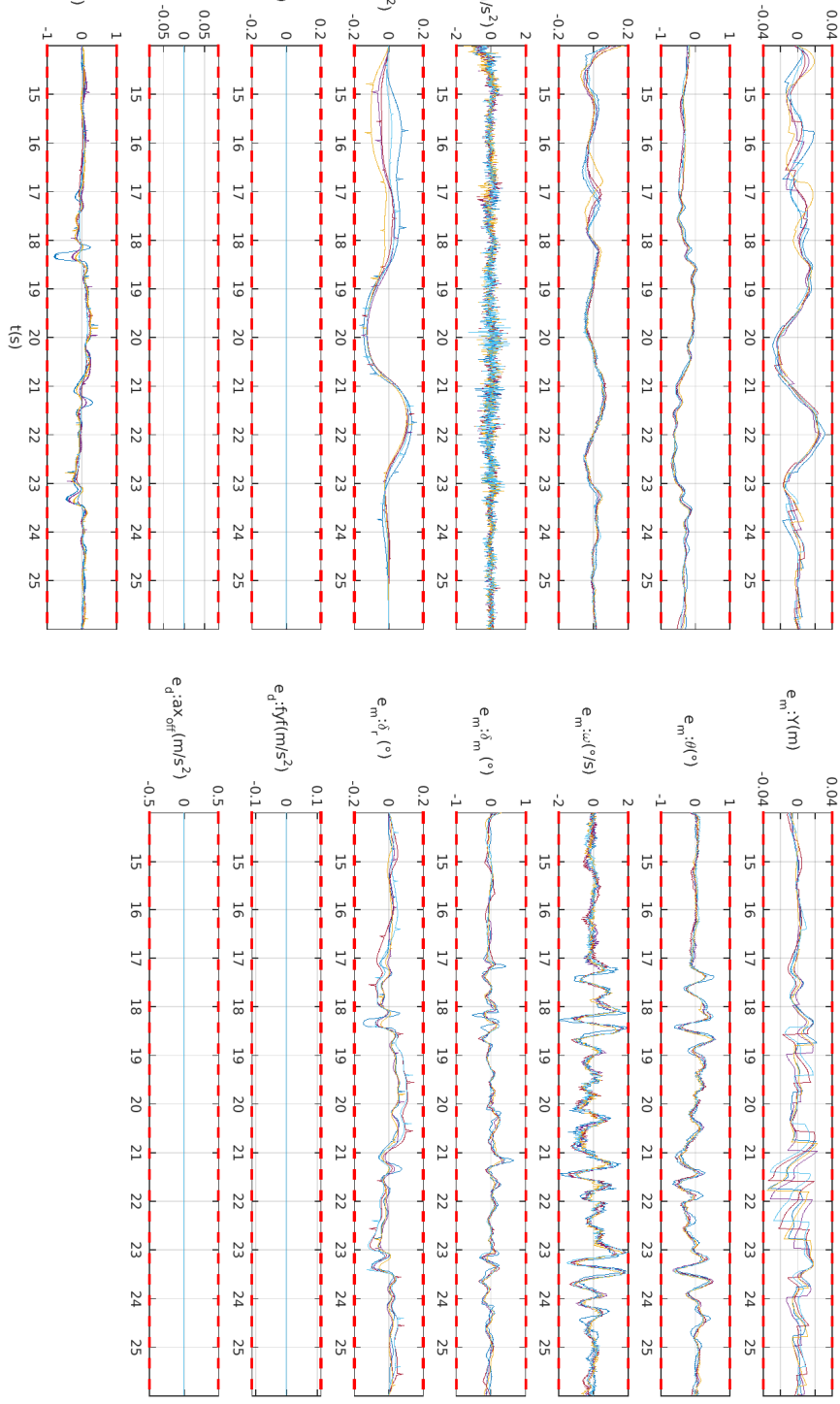


Figure 39: Trace-conformance for f_{BX} model vs FASCar2, DecC2

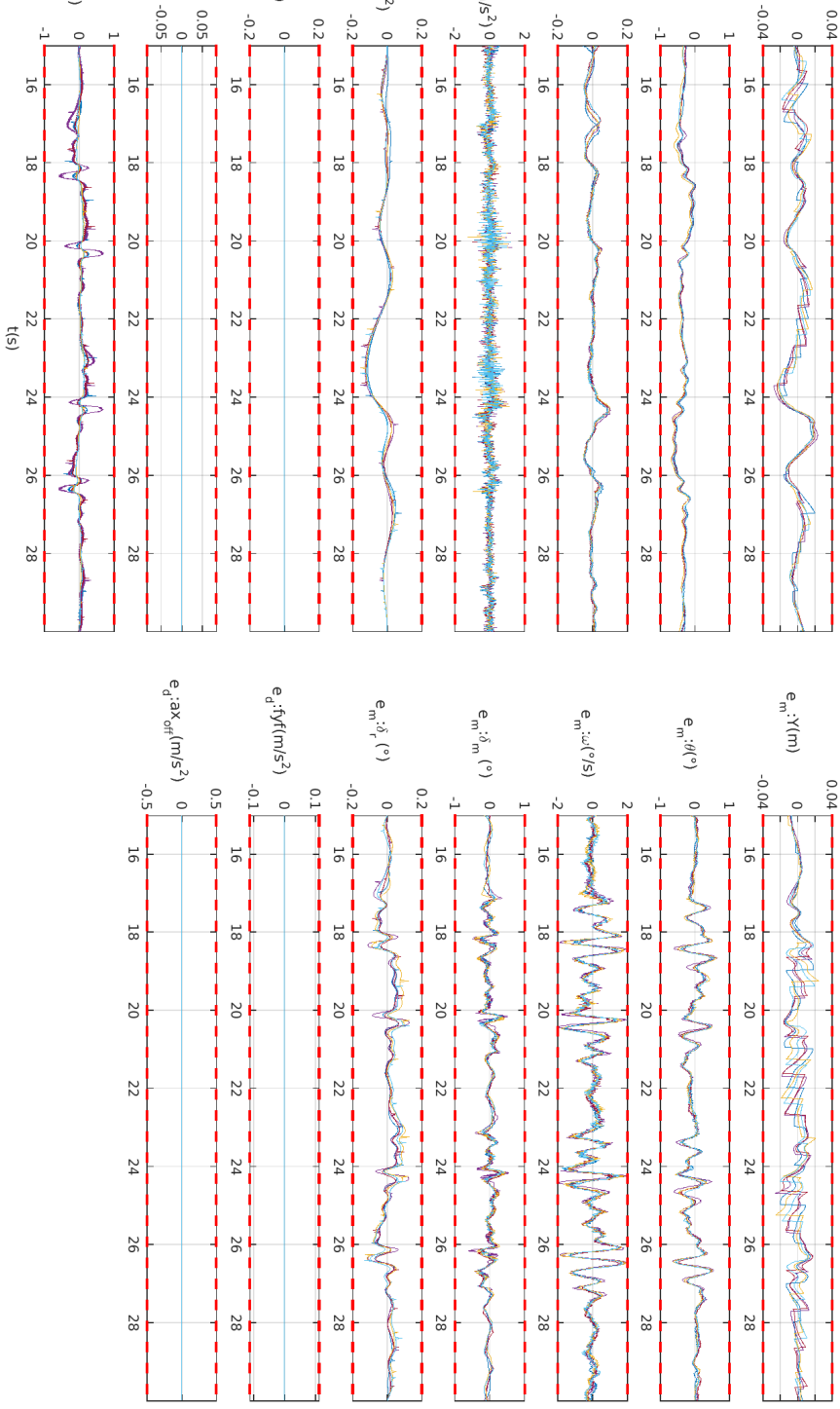


Figure 40: Trace-conformance for f_{BX} model vs FASCar2, DecC3

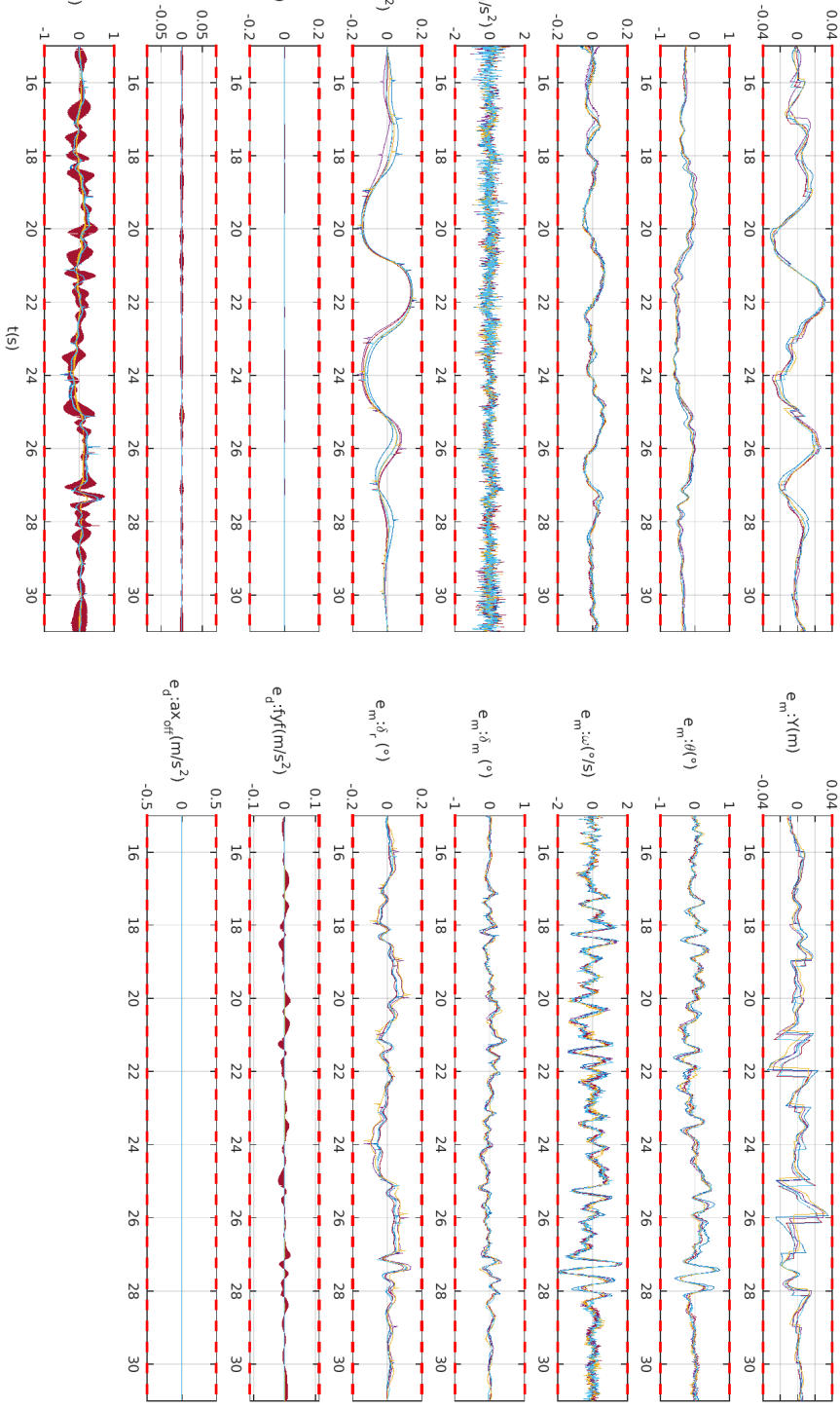


Figure 41: Trace-conformance for f_{BX} model vs FASCar2, DecC4

Appendix D: Tecnalia Vehicle Modeling

Tecnalia Twizy Platform

As real driving platform, Tecnalia maintains an automated Renault Twizy. The steering wheel is controlled by a DC motor, through a gear reduction fixed to the motor axle and the steering bar. The longitudinal and lateral controls are separated in hardware as well as software, so that we can use each system independently. To act on the throttle, a programmable logic controller (PLC) is connected between the embedded PC and the pedal. This sends the target reference (an analogue value) from the control modules in the PC to a trimmer on the throttle that emulates the desired level of pressure applied to the pedal. The computer braking procedure was implemented by adding another DC motor on the brake pedal. The real speed is read directly through the CAN of the vehicle (from the tachometers on the wheels), to close the longitudinal control loop.

A differential GPS-RTK is used for the positioning of the vehicle. This device also integrates an Inertial Measurement Unit (IMU), used to read acceleration and angular speeds.

Figure 42 shows the existing platforms (and their sensors) and the private test track at Tecnalia facilities, used for the plant identification process.

Figure 42: Existing Twizy platform, sensors and private tracks

Validation of Abstract Vehicle Models

Open-loop Model Validation

An abstract open loop model has been implemented to validate the model used in the controller. For this use case, a lane change and a double lane change maneuver have been performed in order to analyze the open-loop responds of the controller.

Based on Figure 43, the controller has two parts; one is formed by a feedback control and the second one is the feed-forward part or model based control. For the experiments, the input for the vehicle is used as the output of the feed-forward.

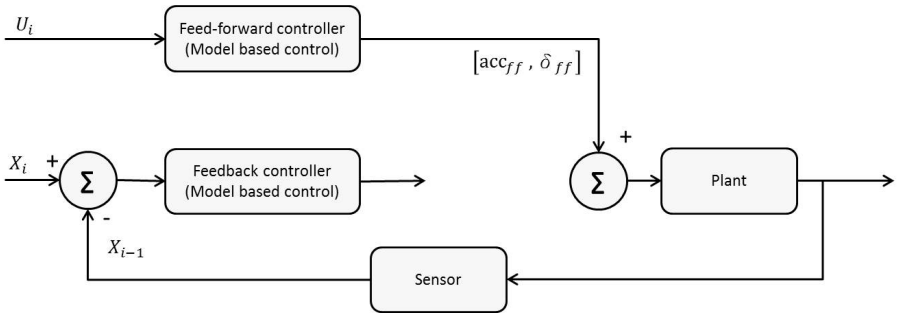
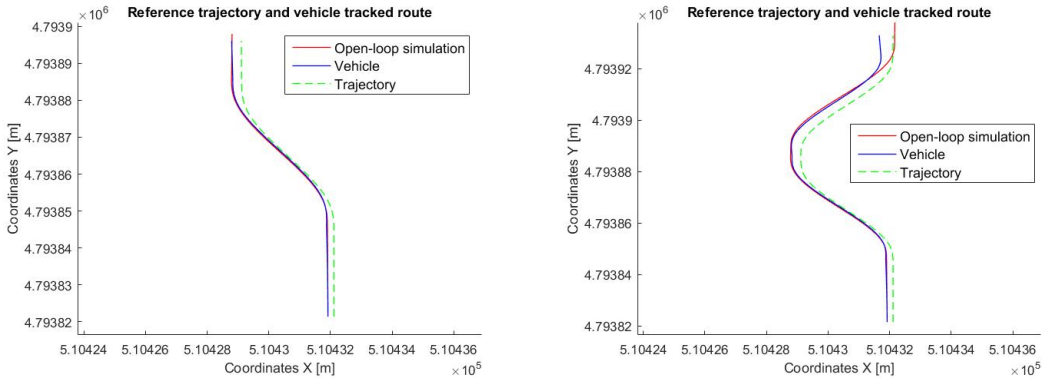


Figure 43: Open-loop diagram of the control system.

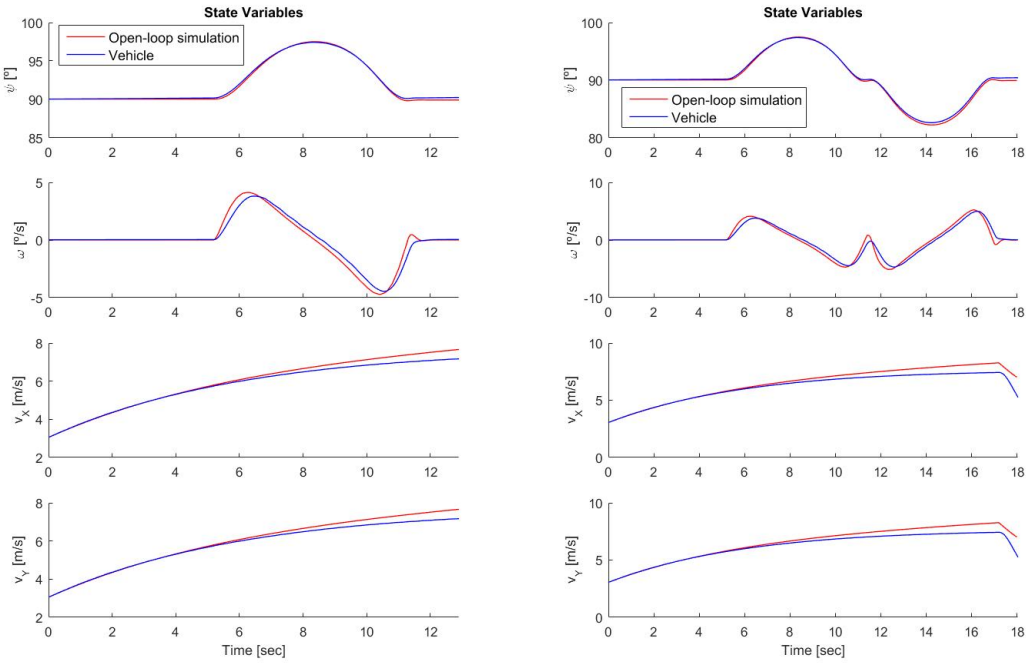
The feed-forward controller uses the trajectory parameters (U) to generate the outputs for the throttle/brake and the wheel angle (proportional to the steering wheel angle). The parameter X_i is the system states on instant i. The plant states are equivalent to the used in section 4 of deliverable D5.1. Figure 44 shows both maneuvers used to validate the controller (see left-hand side for a lane change). The right-hand side of Figure 44 shows a double lane change maneuver. Figures, the green line defines the reference trajectory, the blue line is the position of the DynaCar multi-body model and the red line show the results from the open-loop simulation of the abstract vehicle model. This solution comes from the differential equations (state-pace equation) using the open-loop parameters of the feed-forward controller and the reference trajectory.

Additionally, the position (x-y coordinates) of the vehicle is used to check the yaw angle, longitudinal speed, lateral speed and yaw rate (ω), as part of the state vector. In such a way, the Figure 45 shows those variables of the Dynacar vehicle (blue line) and the ones associated with the abstract open-loop vehicle model (red line) for both scenarios.



(a) (b)

Figure 44: Tracked trajectory (open-loop) for (a) lane change and (b) double lane change



(a) (b)

Figure 45: Open-loop state variables for (a) lane change and (b) double lane change

Closed-loop Model Validation

In the following, we validate the closed-loop behavior of the abstract vehicle model by both feedforward and feedback controllers, as it is shown in Figure 46.

The parameters for the feedback controller are defined by two vector gains. Table 12 shows the values of the longitudinal gains κ_x and the lateral ones κ_y. Both gains are composed for three components. The meaning of each component is related with an integral, proportional and derivative action (in this order).

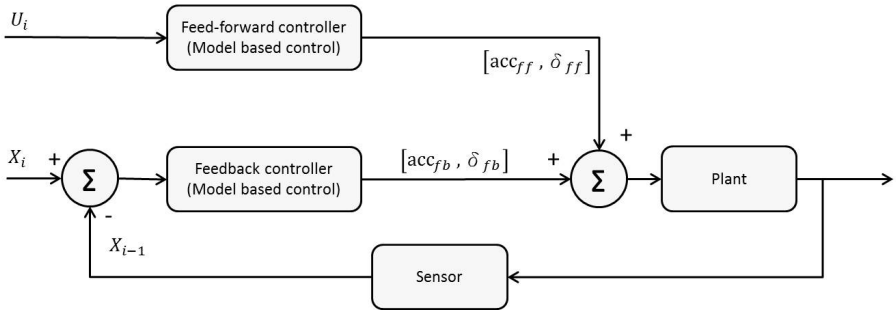


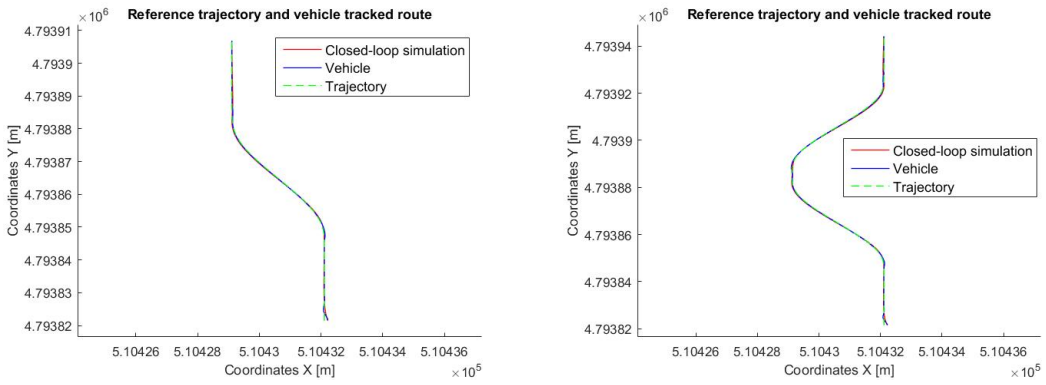
Figure 46: Close-loop diagram of the control system.

$\kappa_x[1]$	$\kappa_x[2]$	$\kappa_x[3]$
0.00	−0.50	9.00
$\kappa_y[1]$	$\kappa_y[2]$	$\kappa_y[3]$
0.00	6.00	3.00

Table 12: Gains of the feedback controller κ_x and κ_y.

The feedback controller is validated by the same procedure used in the feed-foward part (open-loop). This procedure refers to verify that the states generated from the simulation. The modularity of the DyncaCar framework permits an easy integration of the DLR controller (cf. Section 4).

Figure 47 shows the single and double lane change maneuvers used to validate the closed-loop model. In this figure, the green line defines the reference trajectory, the blue line is the position of the Dynacar vehicle and the red line is the result of the closed-loop simulation of the abstract vehicle model (bicycle model).

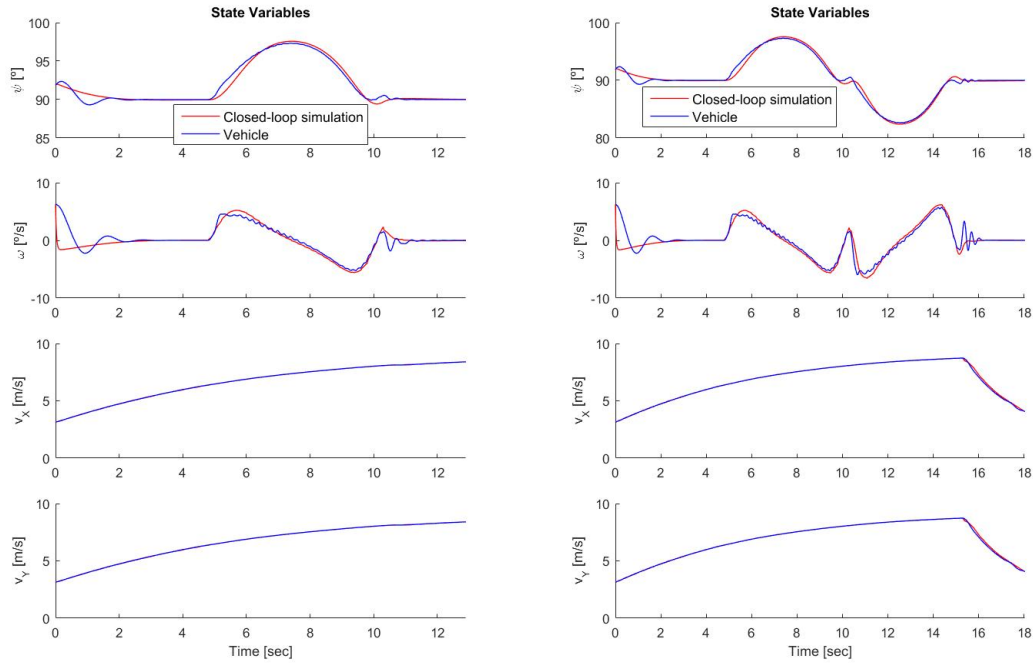


(a) (b)

Figure 47: Tracked trajectory (closed-loop) for (a) lane change and (b) double lane change.

Finally, a validation of the behavior of the yaw angle, longitudinal speed, lateral speed

and yaw rate (ω) was performed, as part of the state vector. Figure 48 shows those variables of the vehicle (blue line) and the ones associated with the closed-loop simulation (red line) for both scenarios.



(a) (b)

Figure 48: Closed-loop state variables for (a) lane change and (b) double lane change.

Dynacar RT Overview

Dynacar (Figure 49) is a simulation tool developed by Tecnalia, which provides a real-time vehicle model covering multiple domains. It focuses on vehicle dynamics, providing a high-fidelity vehicle physics simulation basing on a multibody vehicle model. This is combined with a Pacejka tyre model and sub-models for elements like the engine, transmission, steering system, braking system, aerodynamics, among others. Moreover, it enables to model and integrate components and subsystems of the Electric-Electronic architecture of the vehicle, such es ECUs (electronic control units) and power propulsion elements.

Dynacar allows real-time and accelerated-time simulations. The real-time capability is very valuable, as, combined with its notable modularity and interfacing options, it permits, besides MiL, to execute tests with driver-in-the-loop (DiL) and hardware-in-the-loop (HiL) setups, integrated into Simulink blocks.

Dynacar RT is composed of three main software modules: Real Time (RT) Vehicle

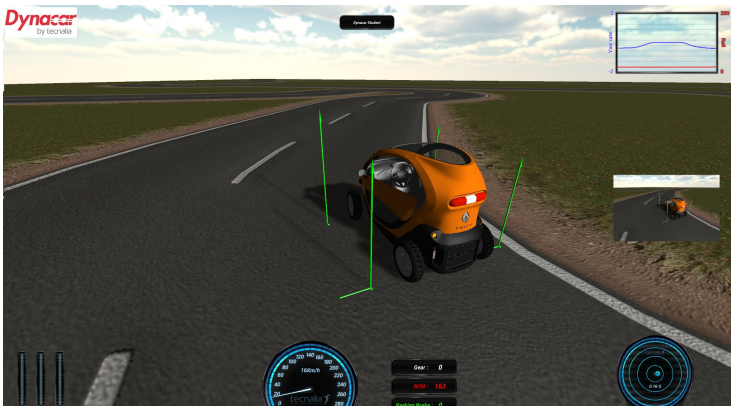


Figure 49: Dynacar by Tecnalia

Dynamics Code that runs on PXI RT Target, Graphic User Interface (GUI) enables project management and vehicle parametrization, and 3D Visual environment for Driving Simulator and test visualization.

Dynacar RT solution is based on: a RT testing platform software (Veristand® real time framework), a Graphic visualization system and vehicle control for real test driving in virtual environment, Dynamic vehicle model running on RT equipment (e.g. PXI hardware), and Test bench hardware depending on the user configuration (ECU, Powertrain, etc.).

Multibody Formulation of Twizy Vehicle in Dynacar

The modeling of chassis vehicle dynamics is based on a semi-recursive multibody formulation with macro-joints model. The key characteristics of the model are as follows:

- Relative coordinates are used to model the vehicle. Mass matrix and force vector of the multibody formulation are recursively obtained [14] deriving equations of motion.
- Each suspension is considered as a macro-joint substituting the suspension links by lookup tables [5], thus leading to a tree-like kinematic structure. The forces due to the spring-damper elements have been introduced through the motion-ratio approach [26].
- Pacejka's 2006 'Magic Formula' semi-empirical approach has been implemented [29], where the tire is characterized by a list of coefficients which can be obtained from experimental tests. This model enables fast and robust tire-road contact force and moment simulation for steady-state and transient tire behavior, using longitudinal, lateral and turn slip, wheel inclination angle and vertical forces as input quantities.
- The resulting formulation is fast and robust, so that different maneuvers can be performed while the execution times are kept within real-time performance, thus allowing carrying

out human-in-the-loop and/or hardware-in-the-loop simulations if desired.

The test vehicle considered in the present work is a Twizy Urban 80, an electric city car. The front and rear suspensions are Pseudo McPherson type, additionally, anti-roll and twist-beam systems are available in the front and rear parts of the vehicle, respectively. Some basic parameters of the car model are listed in Table 13.

Table 13: Basic Parameters of Twizy Vehicle in Dynacar

Wheelbase $[m]$	Front and rear track $[m]$
1.686	1.094
Front knuckle mass $[kg]$	Rear knuckle mass $[kg]$
7.5	19.5
Front wheel mass $[kg]$	Rear wheel mass $[kg]$
8	9.6
$Front/rear$ suspension stiffness $[N/m]$	$Front/rear$ suspension damping $[Ns/m]$
2967.0	1150.9
$Anti-roll$ beam stiffness $[N/m]$	Twist beam stiffness $[N/m]$
9953.1	9970.0
Front tire radius $[m]$	Rear tire radius $[m]$
0.265	0.281

Relative coordinates have been used for the modeling. The three Cartesian coordinates of a chassis point in the front part of the car (x, y, z), along with the three Cardan angles of the chassis with respect to the inertial frame of reference (α, β, γ), are six independent coordinates defining the chassis position. The travel of each suspension is defined by the local (with respect to the chassis reference frame) vertical Cartesian coordinate of the wheel center $Z_i (i = 1 : 4)$. The position of each wheel with respect to the knuckle is defined by an angle around the wheel axis $\varphi_i (i = 1 : 4)$. This makes a total of fourteen independent coordinates, which are grouped into vector Z (Equation 75).

$$Z = [x, y, z, \alpha, \beta, \gamma, \bar{Z}_1, \bar{Z}_2, \bar{Z}_3, \bar{Z}_4, \varphi_1, \varphi_2, \varphi_3, \varphi_4] \tag{75}$$

The steering coordinate is also provided, but it is not included in the list of coordinates, since the steering motion is imposed, therefore, for the front suspensions a different table is generated for every different value of the steering coordinate. Table data are generated, either for the travelling and steering motions, with a resolution of 1 mm in the corresponding input

coordinate (local vertical Cartesian coordinate of the wheel center and steering-rack distance, respectively), the output values being linearly interpolated. Regarding the rear suspensions, corrections are introduced to the knuckle orientation provided by the table to better reproduce the twist beam action: camber and toe corrections are obtained as linear functions of the rolling angle of the vehicle. It must be noted that the resulting model possesses a tree-like topology with no closed loops, as illustrated in Figure 50.

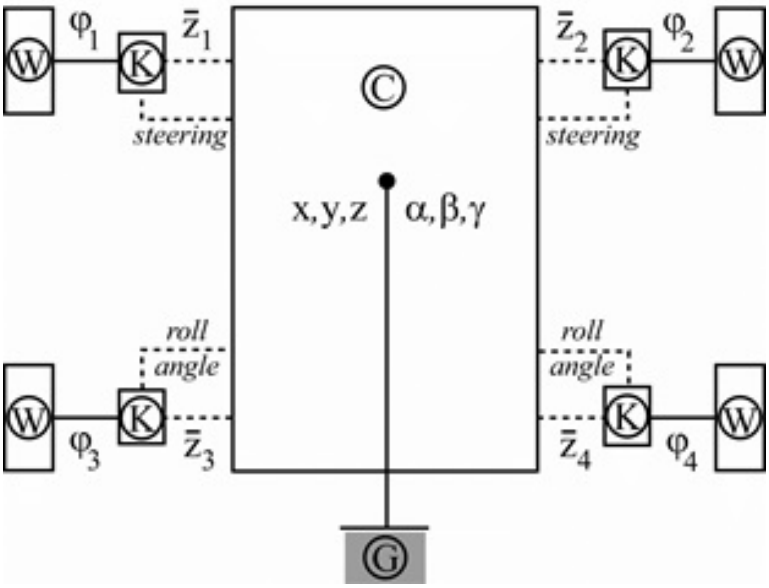
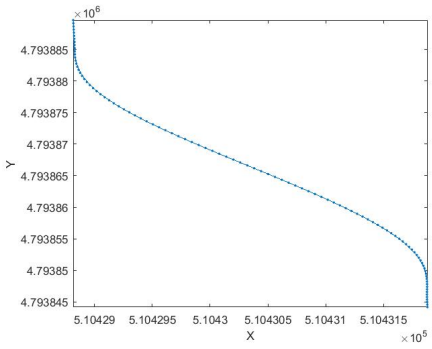


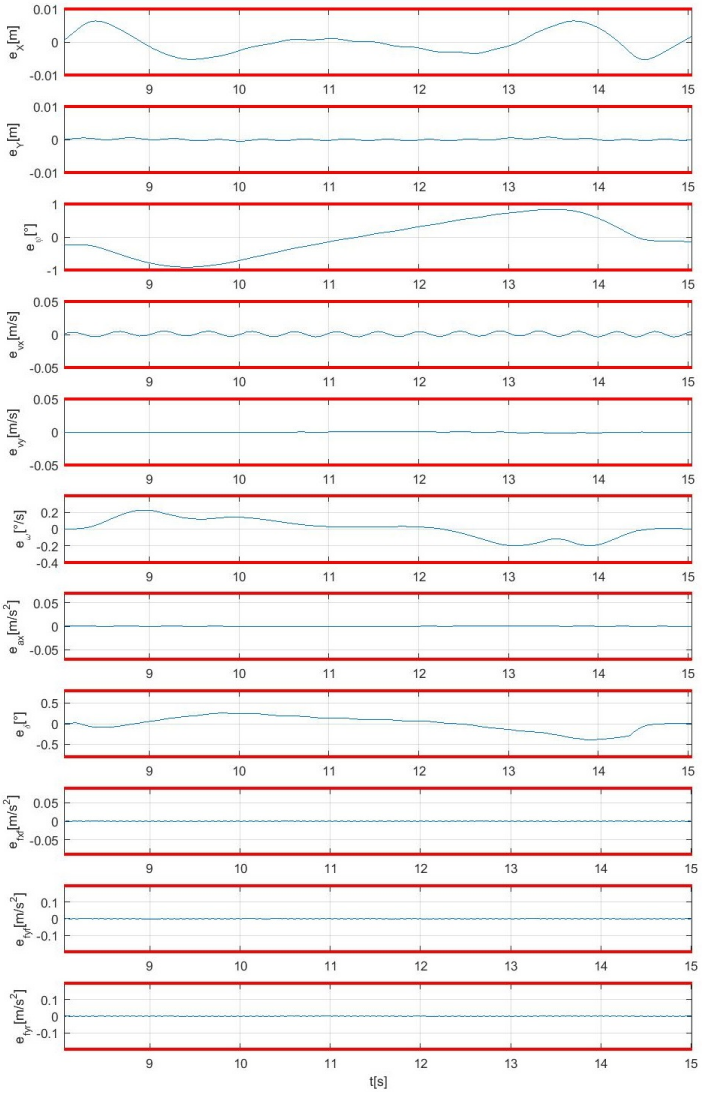
Figure 50: Kinematic structure of the car model. G: ground, C: chassis, K: knuckle, W: wheel and the dotted lines: lookup tables

In summary, at a certain instant of time it is assumed that the independent positions Z and velocities $\dot{Z}$ of the vehicle are known. Then, the tree kinematic structure is defined from the root to the leaves in order to obtain the positions, velocities, rotation matrices and angular velocities of bodies. Likewise, accelerations and angular accelerations $\ddot{Z}$ of bodies are obtained. A maneuver of obstacle avoidance test will consider the car trajectory.

Appendix E: Results Tecnalia Vehicle Trace Conformance

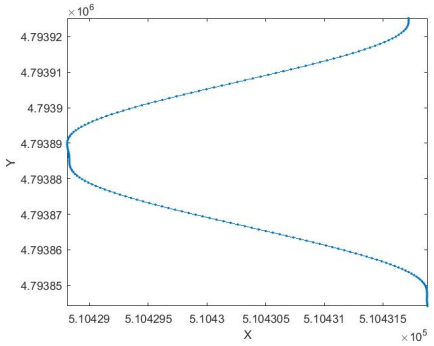


(a)

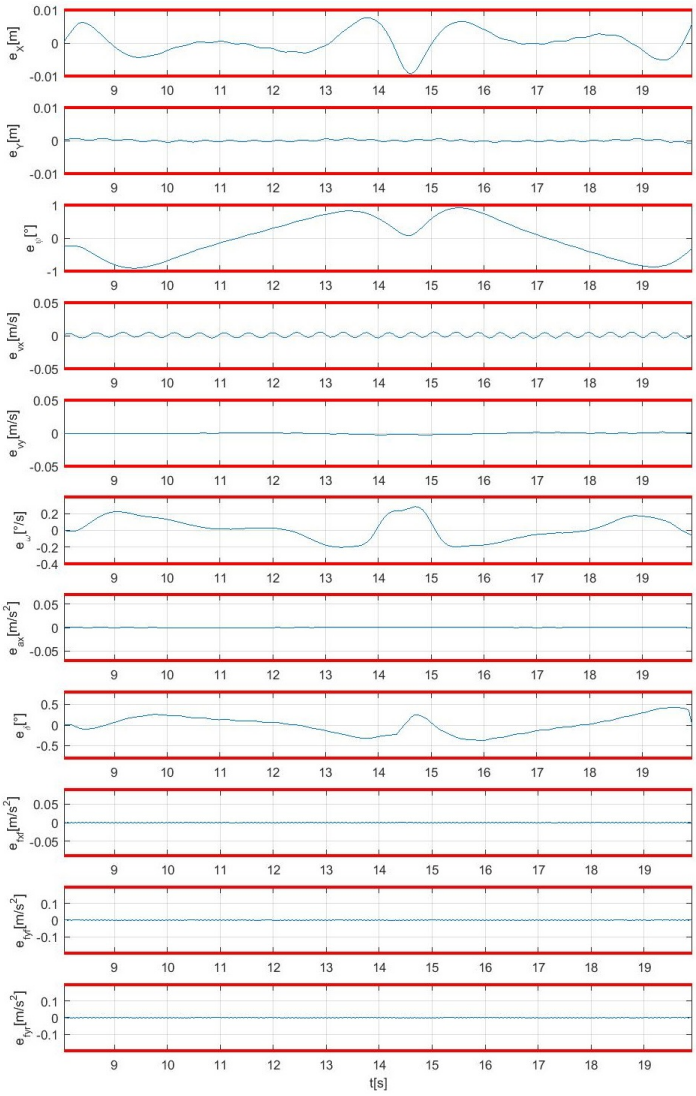


(b)

Figure 51: Conformance test on lane change (open-loop) (a) route and (b) errors and disturbances.

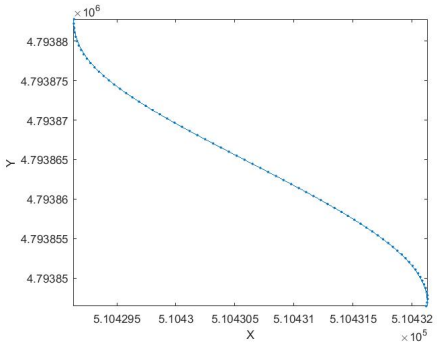


(a)

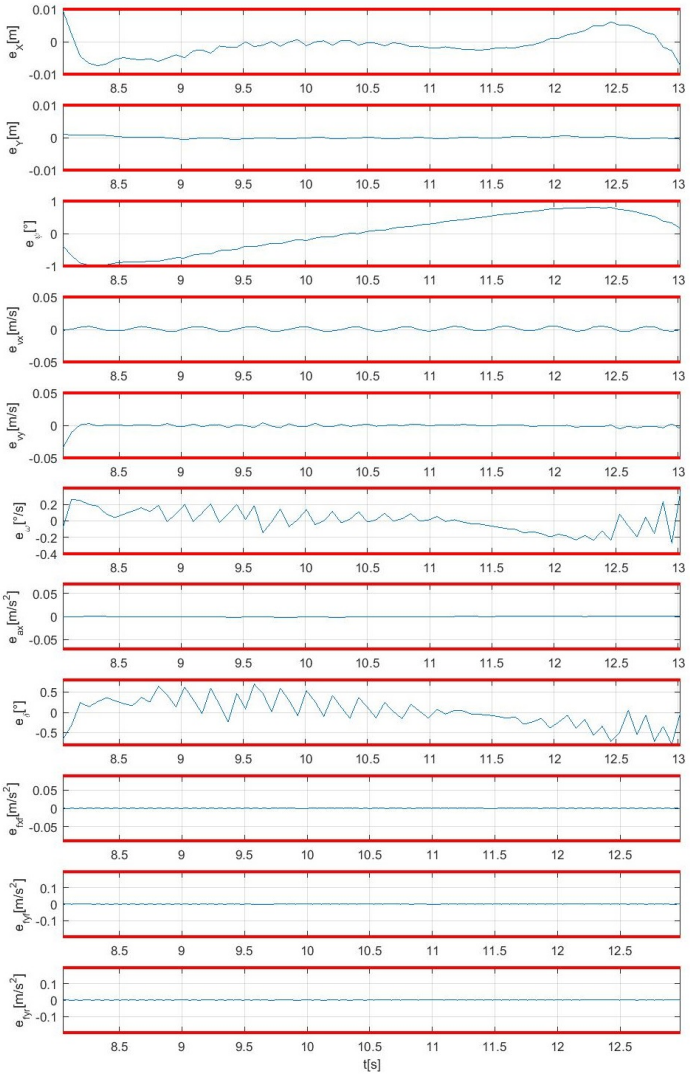


(b)

Figure 52: Conformance test on double lane change (open-loop) (a) route and (b) errors and disturbances.

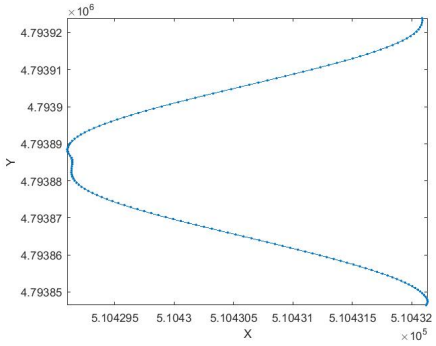


(a)

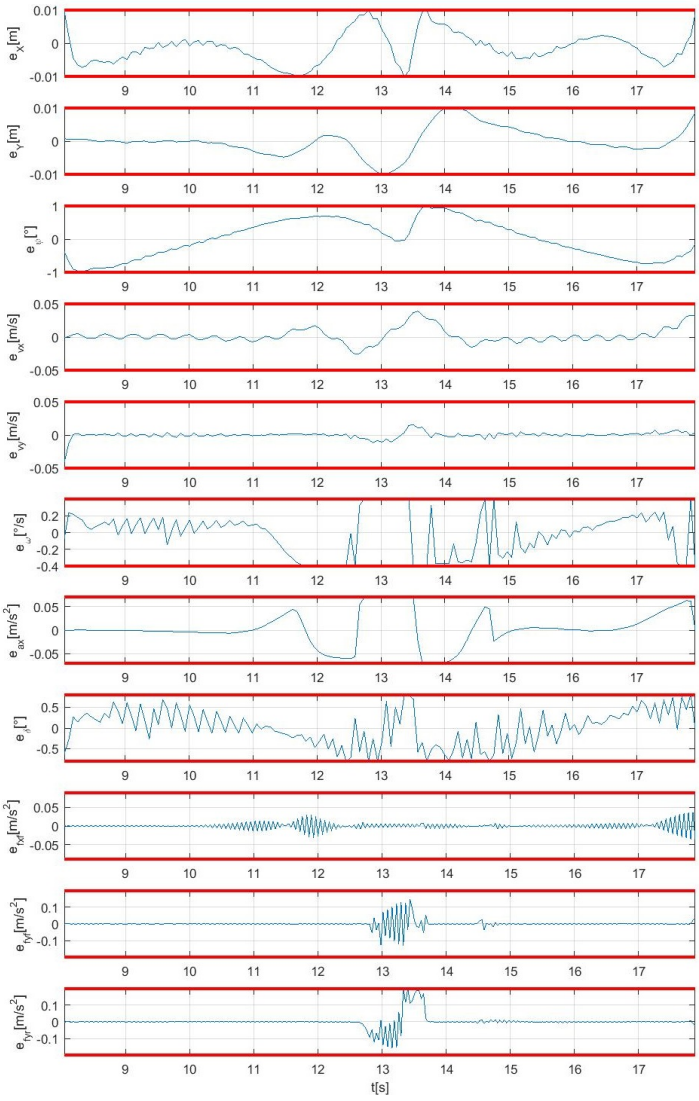


(b)

Figure 53: Conformance test on lane change (closed-loop) (a) route and (b) errors and disturbances.



(a)



(b)

Figure 54: Conformance test on double lane change (closed-loop) (a) route and (b) errors and disturbances.

Appendix F: Derivation of Abstract Wind Turbine Model

The abstract wind turbine model consists of a servo-elastic and an aero-elastic subsystem, which will be described in the following sections.

Servo-elastic Subsystem

In the servo-elastic part, tower fore-aft bending and rotational motion are considered

$$J\dot{\Omega} + M_g/i = M_a(\dot{x}_T, \Omega, \theta, v_0), \tag{76a}$$

$$m_{Te}\ddot{x}_T + c_{Te}\dot{x}_T + k_{Te}x_T = F_a(\dot{x}_T, \Omega, \theta, v_0). \tag{76b}$$

Equation (76a) describes the rotor dynamics with rotor speed Ω, blade pitch angle θ, tower position x_T and rotor effective wind speed v_0. Here, M_a is the aerodynamic torque, M_g is the generator torque, i is the gearbox ratio and J is moment of inertia about the rotor axis

$$J = J_R + J_G/i^2$$

with rotor inertia J_R and generator inertia J_G.

Equation (76b) describes the tower fore-aft dynamics, where F_a is the aerodynamic thrust, m_{Te}, c_{Te} and k_{Te} are the tower equivalent model mass, structural damping and bending stiffness, respectively. They were calculated according to [18] as

$$\begin{aligned} m_{Te} &= 0.25m_T + m_N + m_R, \\ c_{Te} &= 4\pi m_{Te} d_s f_0, \\ k_{Te} &= m_{Te}(2\pi f_0)^2. \end{aligned}$$

with tower mass m_T, nacelle mass m_N, rotor mass m_R, structural damping ratio d_s and natural frequency of the first tower fore-aft bending mode f_0. The parameter values used in the current study are given in Table 15.

Aero-elastic Subsystem

The wind turbine dynamics are highly nonlinear functions of the operating point, which is defined by tip speed ratio λ and blade pitch angle θ. The tip speed ratio is the ratio between the tangential speed of the tip of the blade and the actual velocity of the wind

$$\lambda = \frac{\Omega R_{eff}}{v_{rel}},$$

where Ω is the rotor speed, R_{eff} is the effective rotor radius, and v_{rel} is the relative wind speed. The nonlinearity in the reduced model is contained in the aerodynamic thrust F_a and in the aerodynamic rotor torque M_a

$$F_a = \frac{1}{2}\rho\pi R_{eff}^2 c_T(\lambda,\theta)v_{rel}^2 \tag{77a}$$

$$M_a = \frac{1}{2}\rho\pi R_{eff}^3 \frac{c_P(\lambda,\theta)}{\lambda}v_{rel}^2, \tag{77b}$$

where ρ is the air density and c_P and c_T are the effective power and thrust coefficients, respectively. Again, the parameter values for R_{eff} and ρ are given in Table 15.

The relative wind speed v_{rel} is computed as a superposition of the tower top speed $\dot{x}_T$ and the rotor effective wind speed v_0

$$v_{rel} = (v_0 - \dot{x}_T).$$

In order to calculate aerodynamic thrust and torque from Eq. (77), the so-called aero maps need to be derived. Aero maps are two dimensional look-up tables for the coefficients c_P and c_T. In this study, these lookup tables have been generated using the tool WT_Perf [31]. To find analytical approximations for c_P and c_T, a polynomial fit has been applied to the table data.

$$c_P = p_{21}\lambda^2\theta + p_{20}\lambda^2 + p_{12}\lambda\theta^2 + p_{11}\lambda\theta + p_{10}\lambda + p_{02}\theta^2 + p_{01}\theta + p_{00} \tag{78a}$$

$$c_T = t_{21}\lambda^2\theta + t_{20}\lambda^2 + t_{11}\lambda\theta + t_{10}\lambda + t_{01}\theta + t_{00} \tag{78b}$$

The resulting regression coefficients p_{ij} and t_{ij} are given in Table 14. Please note that θ should be measured in radians.

In [36] the pitch actuator was considered as a third subsystem. In general, blade pitch dynamics have a significant impact on loads and should be included in the model. In this study, however, the reference data is generated by the high fidelity tool FAST (cf. Section 6.2.1). In FAST the blade-pitch angle command from the controller is simply used to orient

c_P	p_{21}	p_{20}	p_{12}	p_{11}	p_{10}	p_{02}	p_{01}	p_{00}
	-0.004	-0.011	-4.262	0.252	0.174	20.833	-2.002	-0.183
c_T	t_{21}	t_{20}		t_{11}	t_{10}		t_{01}	t_{00}
	0.522	-0.006		-7.343	0.147		22.833	-0.005

Table 14: Regression coefficients for the aero maps in Eq. (78) obtained from polynomial fits to WT_perf simulations [31] of the reference turbine [21] .

Parameter	Symbol	Value
Air density	ρ	$1.225\,\frac{\mathrm{kg}}{\mathrm{m}^3}$
Effective rotor radius	R_{eff}	$R\cos\gamma$
Rotor radius	R	63 m
Cone angle	γ	2.5°
Hub height	h_H	90 m
Tower mass	m_T	347460 kg
Nacelle mass	m_N	240000 kg
Rotor mass	m_R	110000 kg
Rotor inertia about rotor axis	J_R	$J_H + 3J'_B$
Hub inertia about rotor axis	J_H	$115926\,\mathrm{kgm}^2$
Blade inertia about rotor axis	J'_B	$J_B + m_B\cos^2\gamma\,(2d_{cm}r_H + r_H^2)$
Blade inertia about root	J_B	$11776047\,\mathrm{kgm}^2$
Blade mass	m_B	17740kg
Center of mass location	d_{cm}	20.475 m
Hub radius	r_H	1.5 m
Generator inertia about rotor axis	J_G	$534.116\,\mathrm{kgm}^2$
Gearbox ratio	i	97
1^{st} tower fore-aft natural frequency	f_0	0.324 Hz
Structural damping ratio	d_s	0.01

Table 15: Parameters of the abstract wind turbine model for the reference turbine [21]

the blade instantaneously with no dynamics. Therefore the pitch actuator subsystem will be omitted from the abstract model as well.

Power Capture and Mechanical Loads

The electrical power P_{el} is calculated by

$$P_{el} = \eta M_g \Omega / i, \tag{79}$$

where η represents the efficiency of the electro-mechanical energy conversion.

Due to the flexible structure, mechanical loads are an important driving factor for the controller design of wind turbines. Concerning fatigue in a wind turbine tower, the tower base

fore-aft bending moment M_{yT} is considered as the most critical load

$$M_{yT} = h_H(c_T\dot{x}_T + k_T x_T). \tag{80}$$

Here, h_H is the hub height and its value is given in Table 15.

References

[1] IEC 61400-1 Wind turbines - Part 1: Design requirements, August 2005.

[2] Safety of machinery – positioning of safeguards with respect to the approach speeds of parts of the human body (iso 13855:2010), May 2010.

[3] Industrial trucks - safety requirements and verification - part 4: Driverless industrial trucks and their systems (iso/dis 3691-4:2006), Sept. 2011.

[4] Robots and robotic devices – safety requirements for personal care robots (iso 13482:2014), Feb. 2014.

[5] M. Acevedo and J. Celigüeta. Real-time dynamic simulation of passenger cars. *In Proceedings of the 27th ISATA on Mechatronics & Supercomputing Applications in the Transportation Industries*, 559-566, Aachen, Germany, 1994.

[6] M. Althoff. An introduction to CORA 2015. In *Proc. of the Workshop on Applied Verification for Continuous and Hybrid Systems*, pages 120–151, 2015.

[7] M. Althoff, D. Hess, and F. Gambert. Road occupancy prediction of traffic participants. In *16th Intern. IEEE Conf. on Intelligent Transportation Systems*, pages 99–105, 2013.

[8] R. Alur, R. Grosu, I. Lee, and O. Sokolsky. Compositional refinement for hierarchical hybrid systems. In *International Workshop on Hybrid Systems: Computation and Control*, pages 33–48. Springer, 2001.

[9] R. Alur, R. Grosu, I. Lee, and O. Sokolsky. Compositional modeling and refinement for hierarchical hybrid systems. *The Journal of Logic and Algebraic Programming*, 68(1-2):105–128, 2006.

[10] D. Araiza-Illan, D. Western, A. Pipe, and K. Eder. Coverage-driven verification: An approach to verify code for robots that directly interact with humans. In *Proc. of the 11th International Hardware and Software: Verification and Testing*, pages 69–84, 2015.

[11] D. Araiza-Illan, D. Western, A. Pipe, and K. Eder. Systematic and realistic testing in simulation of control code for robots in collaborative human-robot interactions. In *Towards Autonomous Robotic Systems: 17th Annual Conference*, pages 20–32, 2016.

[12] S. Bouraine, T. Fraichard, and H. Salhi. Relaxing the inevitable collision state concept to address provably safe mobile robot navigation with limited field-of-views in unknown dynamic environments. In *Proc. of IROS 2011*, pages 2985–2991.

[13] T. S. Chow. Testing software design modeled by finite-state machines. *IEEE transactions on software engineering*, (3):178–187, 1978.

[14] J. Cuadrado, D. Dopico, M. Gonzalez, and M. Naya. A combined penalty and recursive real-time formulation for multibody dynamics. *Journal of Mechanical Design*, 2004.

[15] T. Dang. Model-based testing of hybrid systems. In J. Zander, I. Schieferdecker, and P. J. Mosterman, editors, *Model-Based Testing for Embedded Systems*, chapter 14, pages 383–424. CRC Press, Inc., 2011.

[16] T. Dang and T. Nahhal. Coverage-guided test generation for continuous and hybrid systems. *Formal Methods in System Design*, 34(2):183–213, 2009.

[17] D. Fox, W. Burgard, and S. Thrun. The dynamic window approach to collision avoidance. *IEEE Robot. Autom. Mag.*, 4(1):23–33, 1997.

[18] R. Gasch and J. Twele, editors. *Windkraftanlagen - Grundlagen, Entwurf, Planung und Betrieb*. Springer Vieweg, 8th edition, 2013.

[19] T. A. Henzinger, M. Minea, and V. Prabhu. Assume-guarantee reasoning for hierarchical hybrid systems. In *International Workshop on Hybrid Systems: Computation and Control*, pages 275–290. Springer, 2001.

[20] J. M. Jonkman and M. L. Buhl. FAST users's guide. Technical Report NREL/EL-500-38230, National Renewable Energy Laboratory, 2005.

[21] J. M. Jonkman, S. Butterfield, W. Musial, and G. Scott. Definition of a 5-MW reference wind turbine for offshore system development. Technical Report NREL/TP-500-38060, National Renewable Energy Laboratory, 2009.

[22] N. D. Kelley and B. J. Jonkman. Overview of the TurbSim stochastic inflow turbulence simulator. Technical Report NREL/TP-500-41137, National Renewable Energy Laboratory, 2007.

[23] T. Kruse, A. K. Pandey, R. Alami, and A. Kirsch. Human-aware robot navigation: A survey. *Robotics and Autonomous Systems*, 61(12):1726 – 1743, 2013.

[24] W. S. Levine. *The Control Systems Handbook: Control System Advanced Methods*. CRC press, 2010.

[25] N. A. Lynch, R. Segala, and F. W. Vaandrager. Hybrid I/O automata revisited. In *Hybrid Systems: Computation and Control, 4th International Workshop, HSCC 2001, Rome, Italy, March 28-30, 2001, Proceedings*, pages 403–417, 2001.

[26] W. Milliken and D. Milliken. Race car vehicle dynamics. Technical report, SAE International, 1994.

[27] J. Minguez, F. Lamiraux, and J.-P. Laumond. Motion planning and obstacle avoidance. In B. Siciliano and O. Khatib, editors, *Springer Handbook of Robotics*, pages 1177–1202. Springer, 2016.

[28] S. Mitsch, K. Ghorbal, and A. Platzer. On provably safe obstacle avoidance for autonomous robotic ground vehicles. In *Proc. of Robotics: Science and Systems*, June 2013.

[29] H. Pacejka. *Tyre and vehicle dynamics*. Butterworth-Heinemann, 2006.

[30] S. Pellegrini, A. Ess, K. Schindler, and L. van Gool. You'll never walk alone: modeling social behavior for multi-target tracking. In *Proc. of ICCV*, pages 261–268, 2009.

[31] A. D. Platt and M. L. Buhl. WT_perf users guide. Technical Report NREL/TP-XXXXX, National Renewable Energy Laboratory, 2012.

[32] R. Rajamani. *Vehicle dynamics and control*. Springer Science & Business Media, 2011.

[33] M. Rhudy and Y. Gu. Understanding nonlinear kalman filters part i: Selection of ekf or ukf. *Interactive Robotics Letters, West Virginia University*, 2013.

[34] H. Roehm, J. Oehlerking, M. Woehrle, and M. Althoff. Reachset conformance testing of hybrid automata. In *Proceedings of the 19th International Conference on Hybrid Systems: Computation and Control, HSCC 2016, Vienna, Austria, April 12-14, 2016*, pages 277–286, 2016.

[35] H. Roehm, J. Oehlerking, M. Woehrle, and M. Althoff. Reachset conformance testing of hybrid automata. In A. Abate and G. E. Fainekos, editors, *Proceedings of the 19th International Conference on Hybrid Systems: Computation and Control, HSCC 2016, Vienna, Austria, April 12-14, 2016*, pages 277–286. ACM, 2016.

[36] S. Schuler, F. D. Adegas, and A. Anta. Benchmark problem: hybrid modelling of a wind turbine. In *Proc. of the Workshop on Applied Verification for Continuous and Hybrid Systems (ARCH)*, 2016.

[37] P. Tabuada. *Verification and Control of Hybrid Systems - A Symbolic Approach*. Springer, 2009.

[38] G. J. Tretmans. *A formal approach to conformance testing*. PhD thesis, Universiteit Twente, 1992.

[39] M. Van Osch. Hybrid input-output conformance and test generation. In *Formal Approaches to Software Testing and Runtime Verification*, pages 70–84. Springer, 2006. appended file is a more detailed technical report.

[40] M. P. W. J. van Osch. *Automated model-based testing of hybrid systems*. PhD thesis, Eindhoven University of Technology, 2009.