MISSIONE 4
ISTRUZIONE
RICERCA

FORESEEN

Deliverable D4.2 - Report on the use of Abstract Interpretation for Robustness assessment

Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA
UNIVERSITÀ DI PISA

MISSIONE 4
ISTRUZIONE
RICERCA

FORESEEN

FORmal m**E**thod**S** for attack d**E**t**E**ction in autonomous drivi**N**g systems

PRIN 2022 PNRR

Project number: P2022WYAEW
CUP: I53D23006130001

Deliverable D4.2: Report on the use of Abstract Interpretation for Robustness assessment

Project Start Date: 30/11/2023 **Duration**: 24 months

Coordinator: *University of Pisa*

Deliverable No	D4.2
WP No:	WP3
WP Leader:	RU-MOL
Tasks:	T3.4 - Leader RU-PI
Due date:	M: 15-20
Delivery date:	July 31, 2025
Authors:	RU-MI, RU-MOL, RU-PA, RU-PI

Dissemination Level:

PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Contents

List of Acronyms

CACC	Cooperative Adaptive Cruise Control
CPS	Cyber Physical System
INTO-CPS	Integrated Toolchain for model-based design of Cyber Physical Systems
FP	False Positives
FN	False Negatives
FDR	False Discovery Rate
FNR	False Negative Rate

1 Introduction

This deliverable reports the results of Task 3.4 of WP3 that investigates the robustness of the proposed model checking analysis. Such analysis is based on the model of the system built starting by abstract traces. A set of properties satisfied by traces without attacks is collected. Given a trace, the violation of a property from the given set, is interpreted as a possible attack in the system.

As a first contribution, we provide a simplified linear interpretation of the physical system, on top of which we create a tool for abstract interpretation and execution of the system, using convex polyhedra. We use such interpretation to create a region in state-space of the correct execution traces. For instance, we would like to show that a trace under attack will not belong to such a region in state-space. We also provide a MATLAB implementation of the tool.

As a second contribution, we show examples of validation of robustness of the model checking analysis in case of no-attack/attack using the developed tool. This is done by comparing the results of our linear interpreter with an abstract trace inferred from a real simulation traced using the approach developed in the project (described in Deliverable D4.1).

2 Abstract interpretation of the platoon

In this section we provide a model for abstract interpretation of the system [Cous92][Yam19][Ranz20].

Simple model

Firstly, we start by simplifying the physical model of our system by linearizing it. For simplicity's sake and without any loss of generality we reduced the number of cars in the platoon to just 3, one leader and two followers. The resulting system can be seen as an iterative method

$$x[k+1] = Cx[k] + b,$$

where $x[k]$ is the vector representing the state of the system at time kT. The state vector is made of the following values $x = [a_0, a_1, \varepsilon_1, d_1, \varepsilon_2, d_2]$, which are, respectively:

- The imposed acceleration on the leader
- The acceleration of the first follower car
- The speed difference between the first car and the leader, defined as $\varepsilon_1 = v_0 - v_1$
- The distance between the first car and the leader, defined as $d_1 = p_0 - p_1$

- The speed difference between the second car and the first, defined as $\varepsilon_2 = v_1 - v_2$
- The distance between the second car and the first, defined as $d_2 = p_1 - p_2$

Regarding the quantization constant T, the system works with both $T = 0.05$ and $T = 0.01$, we will consider the former value.

Let us assume the leader moves forward with constant acceleration a_0 and the followers decide their acceleration using the CACC control law:

$$u_i = \alpha_1 a_{i-1} + \alpha_2 a_0 - \alpha_3 \varepsilon_i - \alpha_4 \sum_{j=1}^{i} \varepsilon_j - \alpha_5 d_i$$

The summation found in forth addendum represents the speed difference between the leader the i-th car, by writing the formula this way the save a state variable in the vector x for each follower car we add. The vector of parameters is defined as $\alpha = [0.5 \quad 0.5 \quad -0.3 \quad -0.1 \quad -0.04]$.

Let us also assume the dynamics of the follower cars can be described as

$$a_0[k+1] = \gamma a_o[k], \qquad a_i[k+1] = u_i[k]$$

That is the acceleration of the leader is called by a certain factor γ and the acceleration of the follower cars are set by the CACC control law.

$$v_i[k+1] = a_i[k]T + v_i[k]$$

$$p_i[k+1] = \frac{1}{2} u_i[k]T^2 + v_i[k]T + p_i[k]$$

That is the velocity and position of the cars are approximated by the uniform rectilinear motion.

Then, substituting, obtaining the linear system described by the matrix and vector

$$C = \begin{bmatrix} \gamma & 0 & 0 & 0 & 0 & 0 \\ 1.0000 & \psi & 0.4000 & 0.0400 & 0 & 0 \\ -0.0050 & -0.0010 & 0.9960 & -0.0004 & 0 & 0 \\ -0.0000 & -0.0000 & 0.0100 & 1.0000 & 0 & 0 \\ 0.0050 & -0.0040 & 0.0030 & 0.0004 & 0.9960 & -0.0004 \\ 0.0000 & -0.0000 & 0.0000 & 0.0000 & 0.0100 & 1.0000 \end{bmatrix}, b = \begin{bmatrix} 0 \\ -0.5600 \\ 0.0056 \\ 0.0000 \\ 0 \\ 0 \end{bmatrix}$$

Assuming $\gamma \in (0,1)$, the matrix has spectral radius $\rho(C) < 1$ thus we know that the iterative method defined in this way will converge. We modified the control law of the first follower to $\widetilde{u_1} = u_1 + \psi a_1$ with $\psi = 0.01$, this way we ensure that $\|C\| \neq 0$ while $\rho(C) < 1$ remains true; otherwise, we would've had a singular matrix.

With $\gamma = 1$ the spectral radius is $\rho(C) = 1$, thus there will be no convergence. This is obvious as $\forall k \; x_0[k] = a_0[k] = a_0[1]$.

Polyhedron

The idea is to define a convex polyhedron of the possible states at the state k defined as

$$P_k = \{x : A[k]x \leq d[k]\}$$

Where the matrix A and the vector d have as many rows as the number of inequality constraints imposed on the polyhedron.

Let us consider a single iteration of the method $y = Cx + b$, it follows that $x = C^{-1}y - C^{-1}b$, thus a polyhedron defined as $Ax \leq d$ can be rewritten as $AC^{-1}y \leq d + C^{-1}b$, giving us the update rule:

$$A[k+1] = A[k]C^{-1}$$
$$d[k+1] = d[k] + C^{-1}b$$

Now with this method we can give an abstract interpretation of a set of infinite possible evolutions of the system over time.

Unfortunately, such a method is numerically unstable, that is there’s the numerical error of performing the computation on a computer is so large that the results are meaningless. The culprits are the product between matrices and the computation of the inverse. In the next section we introduce stabler method.

Polyhedron represented with its vertices

Let us consider the base case with $\psi = 0$ and let us represent the convex polyhedron as the convex hull of a set of vertices, that is:

$$P_k = \text{conv}(\{V_1[k], \dots, V_m[k]\})$$

A limitation of this representation is that we can only represent bounded polyhedra, it is possible to represent unbounded polyhedra by adding a cone in the definition; however, for simplicity’s sake we will consider only the case of bounded polyhedra. When $\gamma \in (0,1)$, since the iterative method is convergent, it is impossible for a bounded polyhedron to become unbounded, if not for some numerical errors that might emerge; on the other hand, when $\gamma = 1$, we will limit ourselves to say that the resulting polyhedron is unbounded in the event on the vertices becoming infinity. In fact, with the former condition, $\gamma \in (0,1)$, we have that method will always converge to $x = (I - C)^{-1}b = [0, 0, 0, 14, 0, 14]$; whereas, with the latter, the matrix $(I - C)^{-1}$ is singular thus no unique solution exists.

We can apply our affine transformation to each vertex, obtaining the polyhedron at the next step through the iterative method defined as:

$$P_{k+1} = \text{conv}(\{CV_1[k], \dots, CV_m[k]\}) \oplus b$$

The resulting MATLAB program is

```
VERTICES = [
    0 0 10 3 0 5;
    0.2 0 10 6 0 5;
    0.1 0 -4 6 0 5;
    0 0 15 6 0 5;
    0 0 -4 3 4 8;
    0 3 12 5.5 -2 10;
    0 -2 10 3 4 5;
]';

STEPS = 3000;
t = 0;

for i = 1:STEPS
    VERTICES = C*VERTICES + b;

    t = t + T;
end
```

Let us consider the set of vertices described by the matrix below, where each column represents a point. We iterate the method for 30 seconds, that is for 3,000 iterations. In Figure 1 we show the projection over the $d_1 \times d_2$ plane of the convex hulls given by the vertices at beginning and after the process.

$$V_0 = \begin{bmatrix} 0 & 0 & 0.1 & 0 & 0 & 0 & 0 & 0.5 & -0.24 & 0.11 \\ 0 & 0 & 0 & 0 & 0 & 3 & -2 & -2 & -2 & 2 \\ 3 & 6 & 6 & 6 & 3 & 5.5 & 3 & 2 & 9 & 0 \\ 10 & 10 & 4 & 15 & 4 & 12 & 10 & 10 & 10 & 20 \\ 0 & 0 & 0 & 0 & 4 & -2 & 4 & 4 & 4 & 0 \\ 5 & 5 & 5 & 5 & 8 & 10 & 5 & 7.5 & 2 & 14 \end{bmatrix}$$

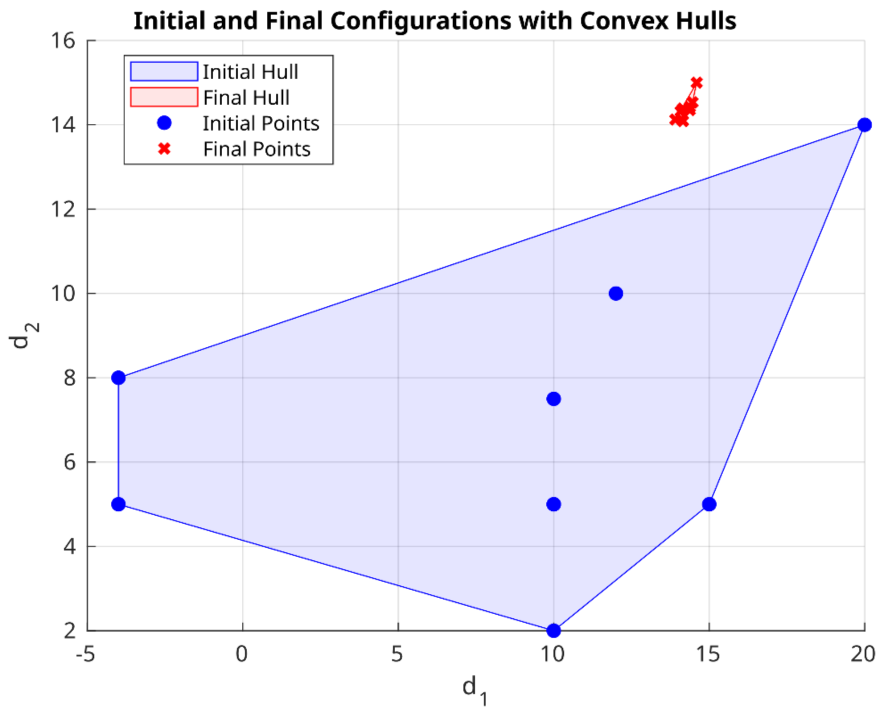


Figure 1 Evolution of a set of points after 30 seconds

With these starting points the method seems to converge in the neighborhood of (14,14), even with the use of $\gamma = 1$. Below we report the matrix of vertices after the 3,000 iterations:

$$V_{3000} = \begin{bmatrix} 0 & 0.20 & 0.10 & 0 & 0 & 0 & 0 & 0.50 & -0.24 & 0.11 \\ -0.00 & 0.19 & 0.10 & -0.01 & 0.00 & -0.01 & -0.00 & 0.50 & -0.26 & 0.11 \\ -0.03 & -0.06 & -0.02 & -0.08 & 0.02 & -0.06 & -0.03 & -0.01 & -0.10 & -0.02 \\ 14.15 & 14.38 & 14.13 & 14.46 & 13.91 & 14.37 & 14.15 & 14.08 & 14.60 & 14.10 \\ -0.01 & -0.05 & -0.01 & -0.07 & -0.02 & -0.04 & -0.06 & -0.05 & -0.14 & -0.02 \\ 14.09 & 14.42 & 14.11 & 14.53 & 14.12 & 14.35 & 14.39 & 14.32 & 15.00 & 14.13 \end{bmatrix}$$

The figures have been truncated to the second decimal place for brevity's sake.

Again, each column of the matrix represents a vertex of the polyhedron after the transformation. We can see how all points are converging to the desired distance of 14 meters. Interestingly, convergence seems to be a little slower for the vertex with negative leader acceleration, in which the vehicles keep a greater gap of fifteen meters.

3 Abstract interpreter

We begin by considering the intervals as defined in the deliverable D4.1 on the driving mode 1. We have the following intervals:

Label	*Acceleration*	*Speed*	*Distance*
VERY LOW	-0.5000 -0.0222	0 0.1672	4 12
LOW	-0.0222 0.0033	0.1672 5.7513	12 13.5
MEDIUM	0.0033 1.4962	5.7513 16.0666	13.5 14.5
HIGH	1.4962 3.4146	16.0666 16.1035	14.5 16.0
VERY HIGH	3.4146 4.0000	16.1035 16.5000	16 20

The lower bounds of VERY LOW and the upper bounds of VERY HIGH are set to the lowest and highest value found in the simulations' traces, respectively. In principle they should be set to minus and plus infinity, but such a value would generate an unbounded polyhedra, that is untreatable with the method presented.

Let us assume the acceleration of the leader zero, i.e. $a_0 = 0$. Let us consider all the possible combinations of labels for the values of the tuple $x = [a_0, a_1, \varepsilon_1, d_1, \varepsilon_2, d_2]$, keep in mind that $\varepsilon_i = v_{i-1} - v_i$; we have to vary a grand total of six variables over five possible values, giving us $5^6 = 15'625$ combinations.

Since we are considering closed intervals for each of the state variables of x – even the acceleration can be considered as $a_0 \in [0,0]$, – the resulting polyhedron is a hyperrectangle described by $2^6 = 64$ vertices, where six is the dimensionality of the state vector. All the vertices represent all the possible combinations of the lower and upper bounds of the various intervals. Since the acceleration of the leader remains constant, we can avoid saving the permutations on a_0, thus the polyhedron can be described by $2^5 = 32$ vertices.

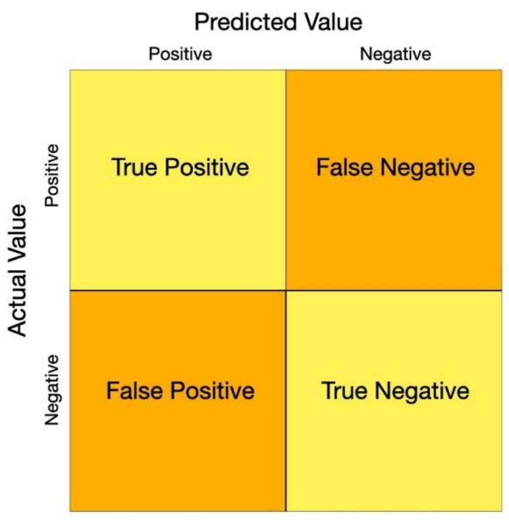
4 Analysis of the robustness of the model checking analysis based on abstract traces

Let us consider a state x_0 at a certain time t_0 and the state x_1 at time t_1 from a certain simulation trace. The abstract interpreted will associate to these two states two sets of labels, that can be considered as two hyperrectangles (polyhedra) such that $x_0 \in P_0, x_1 \in P_1$.

The idea is to feed the polyhedron P_0 to the iterative method for enough steps to reach t_1, the resulting polyhedron, let us call it R_1, represents the set of possible evolutions.

Ideally $R_1 \subseteq P_1$ as the iterative method can represent more complex state-spaces that simple hyperrectangles and one would expect the entirety of the feasible points predicted by the iterative method to be within the feasible points given by the label-based abstract domains. We want to gauge the quality of the step that maps the points in P_0 to P_1 by using the abstract interpretation.

Let us define some metrics to evaluate these two methods against each other. The **intersection** defined as $Z_1 = R_1 \cap P_1$, gives an indication of robustness as it represents correct information learnt by the analysis, that is it represents the *true positives*. The **difference** $P_1 - Z_1$ represents the state-space of possible states that were not learnt by the model checking analysis, that is the *false negatives* (FN). The **difference** $R_1 - Z_1$ represents states learnt by the model checking which are not correct, that is the *false positives* (FP). We also define the **union set** as $U_1 = R_1 \cup P_1$.



For instance, in a spam filter for e-mails, the *false negatives* are spam e-mails that were classified as not spam; whereas the *false positives* are normal e-mails that were classified as spam.

We consider the hypervolumes of these metrics. We also consider the **false discovery rate** (FDR), that is the number of false positives divided by the cardinality of R_1 ; and the **false negative rate** (FNR), that is the number of false negatives divided by the cardinality of the set P_1.

We also observe how the iterative method tends to quickly *flatten out* – so to speak – certain dimensions when considering the space-state in its entirety, so we will consider three projections on hyperspaces of lower dimensionality:

- A **5D** projection in which we drop the acceleration of the leader, after all this is a constat as it is an input of the problem, thus certainly resulting in a polyhedron with volume zero in the 6D space
- A **3D** projection in which we remove the inter-vehicular distances and we keep only $a_1, \varepsilon_1, \varepsilon_2$
- A **2D** projection in which we keep only the two inter-vehicular distances d_1, d_2

For instance, let us consider the following state variables from a simulation trace:

$$x_0 = [2.4, 2.3565, 0.7953, 7.8037, 0.6073, 6.6309]$$

$$x_1 = [2.8, 2.9476, 0.7978, 8.6090, 0.6265, 7.2450]$$

These two states appeared at time 5 and 6 seconds respectively, thus resulting in a time interval of $t_1 - t_0 = 1$ second.

We run the procedure, obtaining the following figures:

Volume	***5D***	***3D***	***2D***
$\|P_0\|$	2.613E+04	4.083E+02	6.400E+01
$\|R_1\|$	1.419E-05	3.058E+01	4.064E+02
$\|P_1\|$	2.613E+04	4.083E+02	6.400E+01
$\|Z_1\|$	1.130E-07	1.421E+01	6.207E+01
$\|U_1\|$	8.361E+04	5.757E+02	4.104E+02
$\|R_1 - Z_1\|$ (False Positives)	1.408E-05	1.637E+01	3.443E+02
$\|P_1 - Z_1\|$ (False Negatives)	2.613E+04	3.941E+02	1.927E+00
$\|R_1 - Z_1\|/\|R_1\|$ (False Discovery Rate)	99.204%	53.523%	84.727%
$\|P_1 - Z_1\|/\|P_1\|$ (False Negative Rate)	100.000%	96.519%	3.011%

The E denotes the scientific notation, for instance 1.4E5 is 1.4×10^5. The two bars |P| denote the volume of a polyhedron.

When considering the most complete projection in the five-dimensional hyperspace, quite unfortunately, the rate of false discovery and false negatives is almost 100%. If we drop the inter-vehicular distances, thus considering the 3D projection on ($a_1, \varepsilon_1, \varepsilon_2$), the false negative ratio falls to 96% and the false discovery rate to 53%. On the other hand, when considering only the projection on the dimensions of the inter-vehicular distances, the false discovery rate remains quite terrible, increasing to 84%; but the false negative rate decreases to an impressive 3%.

In Figure 2 we plot the projection of the three polyhedra onto the d_1, d_2 plane. This projection is related to the platoon safety property that uses this metric, that is the distance between a vehicle and the one in front. We see how the iterative method gives us a greater area than what we got with our abstract interpreter. When considering this projection, we see that are certain points in P_0 that might cause a collision, i.e. $d_i < 0$, this might be due to certain points in P_0 that under normal conditions should not appear in a nominal – that is without attacks – platoon trace. We also see that it is almost true that, for this projection, P_1 is almost contained in R_1.

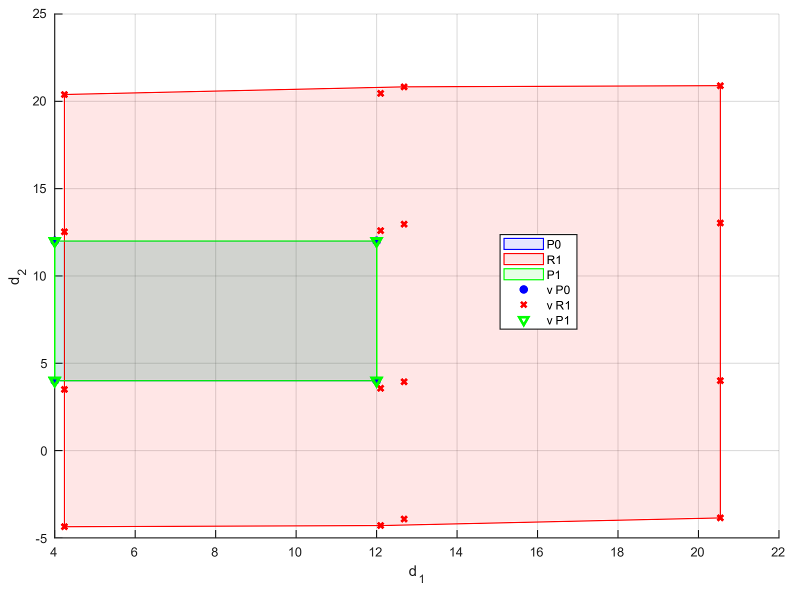


Figure 2 2D projection over (d1, d2) P0 and P1 are same polyhedron

In Figure 3, when projecting over the other three axis $a_1, \varepsilon_1, \varepsilon_2$ we obtain a smaller volume. In this projection, the P_1 is clearly not contained in R_1 and the section that does not overlap is greater, thus producing a greater number of false negatives, as already described in the previous table.

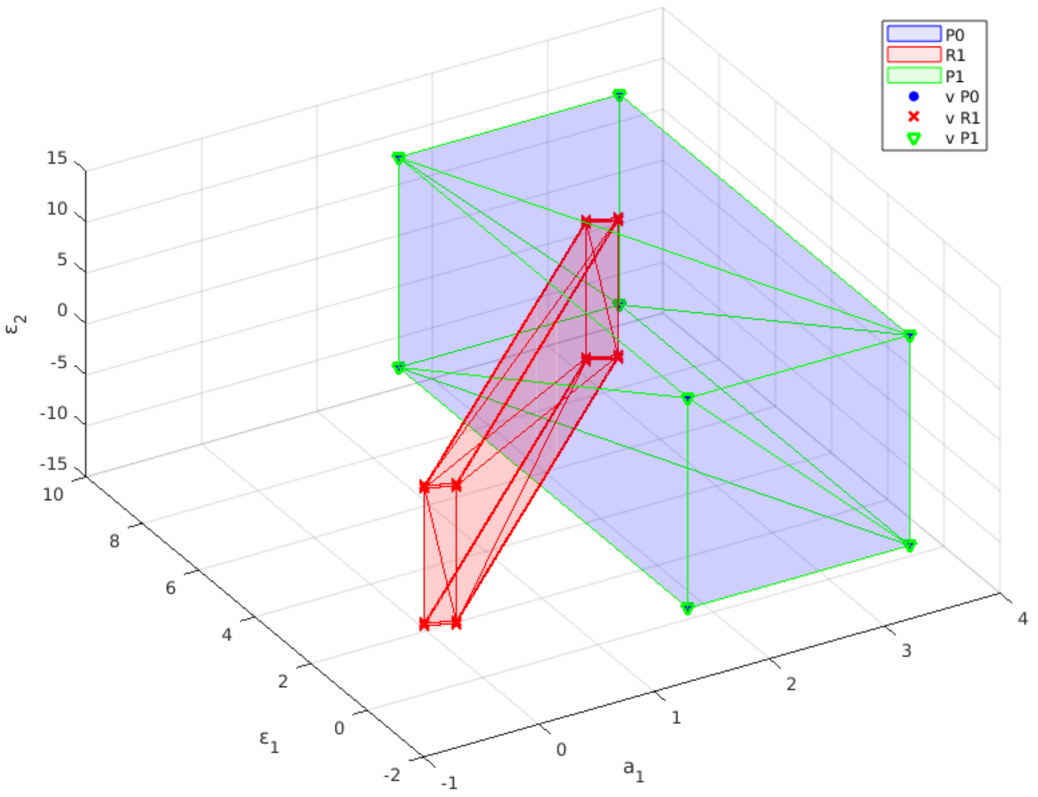


Figure 3 3D projection over $a_1, \varepsilon_1, \varepsilon_2$ P0 and P1 are same polyhedron

In general, one can analyze projection over different axis, in one example we considered the projection over the inter-vehicular distances and in the other one, we considered the projection over the relative speeds and the acceleration of the first car.

One glaring issue of the 5D projection is that the $|Z_1| \simeq 0$, that is the volume is very small compared to the other two polyhedra, possibly making the figures meaningless.

Let us now consider another step from the same simulation trace:

$$x_0 = [0, 0.0168, 0.0223, 13.8765, 0.0377, 13.7347]$$

$$x_1 = [0, -0.0110, 0.0043, 14.2764, 0.0101, 14.1403]$$

These two states appeared at time 5 and 6 seconds respectively, thus resulting in a time interval of $t_1 - t_0 = 6$ seconds. These states come from a part of the simulation without attacks where the steady state was about to be reached.

These two data-points are interesting because they map to two different classes and thus two different hyperrectangles when fed to the abstract interpreter.

We run the procedure, obtaining the following figures:

Volume	***5D***	***3D***	***2D***
$\lvert P_0 \rvert$	2.57E-01	2.57E-01	1
$\lvert R_1 \rvert$	2.12E-12	1.42E-04	2.07E+00
$\lvert P_1 \rvert$	8.01E-03	8.01E-03	1
$\lvert Z_1 \rvert$	1.05E-12	1.34E-04	8.18E-01
$\lvert U_1 \rvert$	1.10E-02	8.11E-03	2.33E+00
$\lvert R_1 - Z_1 \rvert$ (False Positives)	1.07E-12	8.77E-06	1.26E+00
$\lvert P_1 - Z_1 \rvert$ (False Negatives)	8.01E-03	7.87E-03	1.82E-01
$\lvert R_1 - Z_1 \rvert / \lvert R_1 \rvert$ (False Discovery Rate)	50.38%	6.15%	60.54%
$\lvert P_1 - Z_1 \rvert / \lvert P_1 \rvert$ (False Negative Rate)	100.00%	98.33%	18.19%

When considering the hyperspace in its entirety we still get a terrible 100% of false negative rate, but, on the bright side, the false discovery rate is a more reasonable 50%. When considering the projections on the 3D space – i.e. we consider the 3D projection, – we obtain further improvement in the false discovery rate, decreasing to almost 6% but the false negative rate remains at an unremarkable 98%. If the study the 2D projection on the inter-vehicular distances, we see that the false discovery rate increases again to 60%; but the false negative rate drops to a quite positive 18%, again reaffirming the 2D projection to be the best one among these under study.

These results confirm that the 2D projection on the inter-vehicular distances is the one that works better as it reduces the rate of false negatives greatly; however, the false discovery rate remains quite high in any case.

Figure 4 shows the polyhedra projected over the 2D space defined by the two inter-vehicular distances (d_1, d_2). Unlike the previous example, there's no possibility of collisions. Also, the figures are near the convergence point $(14,14)$ for both the iterative method and the abstract interpreter. The two areas overlap greatly, as we already saw from the metrics in the table.

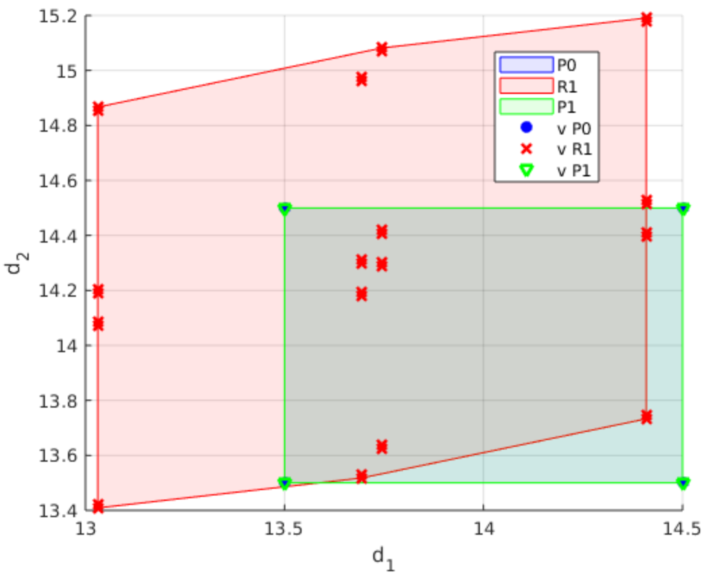


Figure 4 2D projection over (d1, d2) P0 and P1 are same polyhedron

Figure 5 shows the projection on the space defined by $a_1, \varepsilon_1, \varepsilon_2$. Again, both the iterative method and the abstract interpreter are converging near the convergence point (0,0,0). In this case there's a change of class, and thus of hyperrectangle, considered by the abstract interpreter.

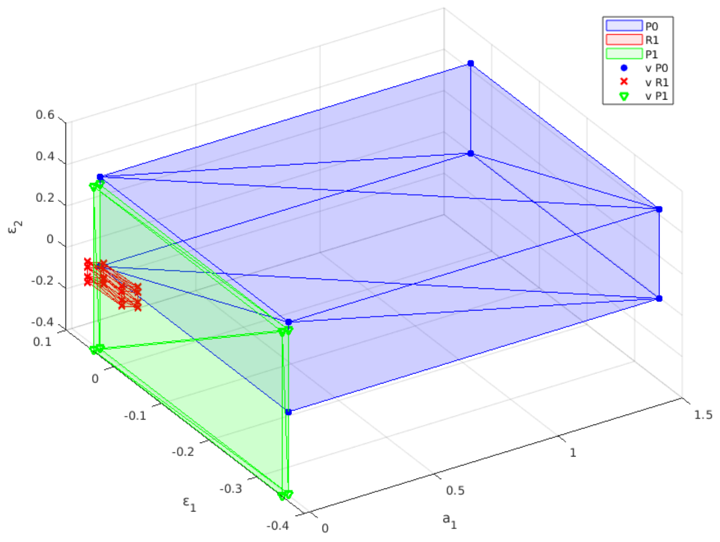


Figure 5 3D projection over $a_1, \varepsilon_1, \varepsilon_2$

Again, in the 5D projection we have $|Z_1| \simeq 0$, that is the volume is very small compared to the other two polyhedra.

5 Analysis of a trace under attack

We now want to apply our method of abstract interpretation for the detection of an attack during the execution of the system by analyzing a trace of simulation. Firstly, we take two data-points representing the state of the platoon, then – as we did in the previous section –we use the abstract interpreter to map these two points to two abstract hyperrectangles P_0, P_1, then we apply on the first hyperrectangle the iterative method to obtain a certain convex polyhedron R_1.

Ideally, the method should be able to detect a possible attack as the space R_1 describing the feasible states should not overlap at all with the possible states found in the trace, that is $P_1 \cap R_1 = \emptyset$.

We consider a simulation trace of an attack that begins from $t_A = 60$, that is in the first sixty seconds the platoon behaves normally, then an attack takes place. The attack is performed on the first car, and it is one of the attacks on the actuators described in the previous deliverables. In particular, the acceleration command of the first car is modified as follows:

$$\tilde{u}_1 = u_1 + 0.08$$

Where the u is the command imposed by the CACC control law on the car. It is important to note that this is the u found on the MATLAB and INTO-CPS implementation of the platoon, previously introduced and defined and defined in the previous deliverables, and not the u of the iterative method found in the previous sections of this one.

Let us now consider another step from the same simulation trace:

$$x_0 = [0, 0.008, 0.003, 14.000, 0.001, 13.994\,]$$

$$x_1 = [0, -0.010, -0.115, 12.856, 0.001, 14.283]$$

These two states appeared at 59 and 69 seconds respectively, thus resulting in a time interval of $t_1 - t_0 = 10$ seconds. That is one second before the attack was about to start and nine seconds after the attack began.

As before, we ran the method and obtained the following figures:

Volume	***5D***	***3D***	***2D***
$\lvert P_0 \rvert$	4.69E-01	4.69E-01	1
$\lvert R_1 \rvert$	1.55E-13	4.71E-05	1.56E+00
$\lvert P_1 \rvert$	1.20E-02	8.01E-03	2.50E+00

$\|Z_1\|$	0	4.71E-05	1.34E-02
$\|U_1\|$	1.66E-02	8.01E-03	3.56E+00
$\|R_1 - Z_1\|$	1.55E-13	1.49E-13	1.54E+00
$\|P_1 - Z_1\|$	1.20E-02	7.96E-03	2.44E+00

Interestingly, the volume of the intersection is empty, that is $Z_1 = P_1 \cap R_1 = \emptyset$, or at least almost empty in the various projections. Thus, we showed that none of the possible correct evolutions from P_0 can lead to any state found in P_1, thus suggesting that something isn't quite right in the behavior of the platoon. Indeed, an attack is taking place.

In Figure 6 we show the projection of the found polyhedra on the d_1, d_2 plane. We can see how the rectangle containing the trace under attack, represented by the green area "P1" in the plot, contains values of the distance between the leader and the first car between 12 and 13.5 meters – i.e. $d_1 \in [12, 13.5]$ – suggesting car 1 is traveling way to close. This confirms again the importance of the distance metric as a metric to gauge the correct operations of the platoon. The small overlapping section Z_1 is 0.3% of the union U_1 of the two sets.

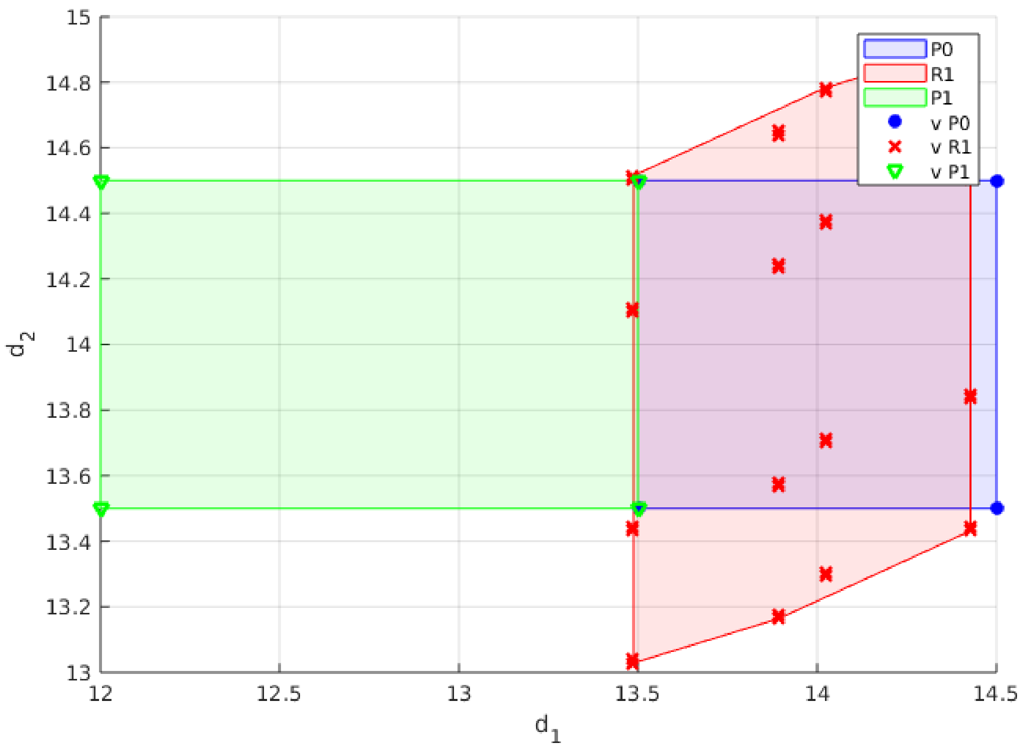


Figure 6 2D projection over (d1, d2)

Analogously, when considering the 3D projection, show in Figure 7, there's little overlap between the two sets, in particular the volume of Z_1 is only the 0.6% of the volume of the union U_1.

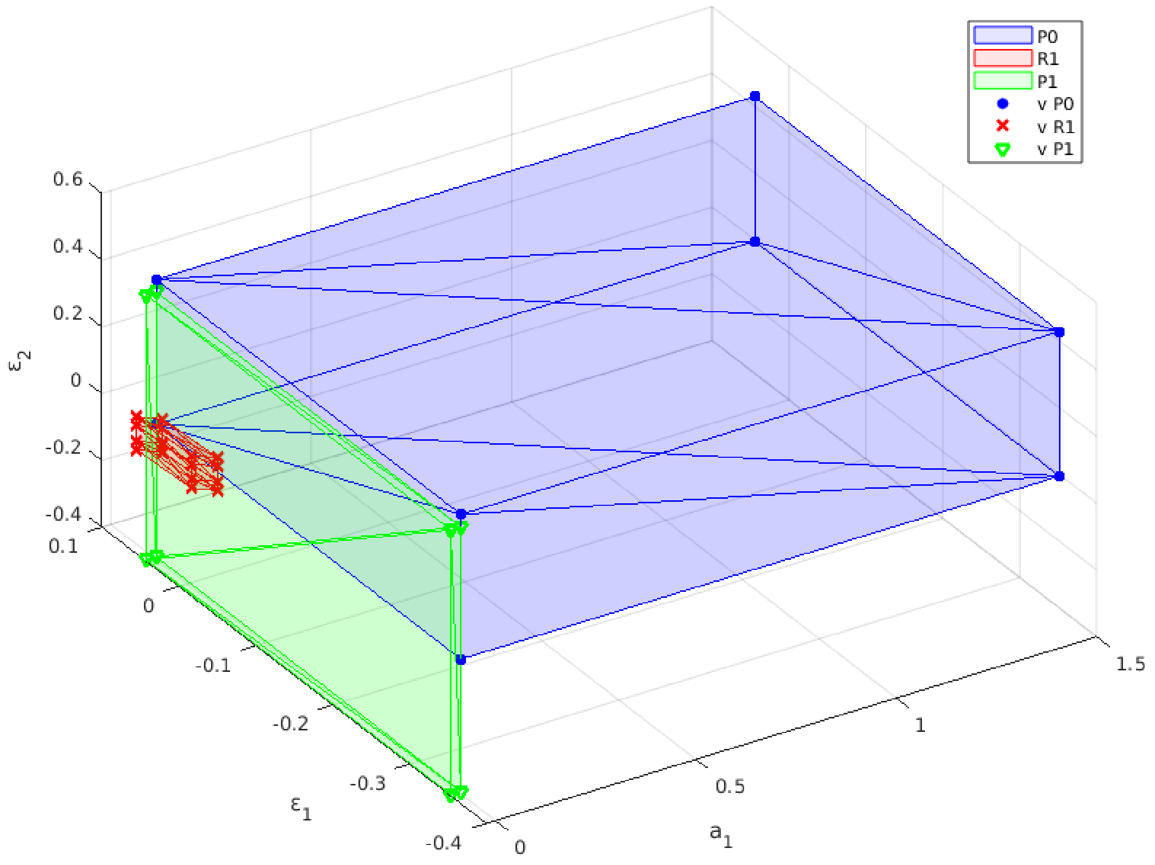


Figure 7 Projection over a1, eps1, eps2

The MATLAB programs are available at the following git repository hosted on the GitHub website:

https://github.com/ForeseenPRIN/formal_method_analysis

6 Conclusions

This deliverable has presented the development and application of an abstract interpretation framework to assess the robustness of a model checking analysis for attack detection in autonomous vehicle platoons. The work focused on creating a simplified, linearized model of a three-vehicle platoon using convex polyhedra to represent sets of possible system states to then compare the results with the abstract traces obtained through an abstract interpreter.

The core methodology involved propagating a set of initial states (a polyhedron) through the linearized system dynamics to compute the reachable set R_1 after a given time window. This predicted set was then compared against the abstract state P_1 derived from a simulation trace using an interval-based abstraction. The comparison, quantified through metrics like False Discovery Rate (FDR) and False Negative Rate (FNR) across different dimensional projections, served as the basis for evaluating the robustness of the trace-based analysis.

The key findings of our investigation are as follows:

1. **Effectiveness in Attack Detection:** The primary success of this approach is its demonstrated ability to detect attacks. In the analyzed scenario, the abstract interpreter correctly identified an anomalous condition: the reachable set R_1 from a pre-attack state had a null intersection with the abstract state P_1 observed during the attack ($Z_1 \simeq \emptyset$). This clear discrepancy signals a violation of the system's expected behavior, successfully flagging the presence of the actuator attack.

2. **Projection-Dependent Performance:** The robustness metrics are highly dependent on the chosen state-space projection. The analysis revealed that:

   - **High-Dimensional Projections (5D/3D):** These projections, which include acceleration and relative velocity variables, resulted in prohibitively high false negative rates (often >96%). This indicates that the current interval-based abstraction for these variables is too coarse, leading the model checker to potentially miss many deviations that the precise linear model would catch.

   - **2D Distance Projection:** In contrast, the projection onto the inter-vehicular distances (d_1, d_2) proved to be the most reliable metric for this specific safety property. It consistently yielded low false negative rates (as low as 3% and 18% in nominal cases), meaning the abstract interpreter and the linear model largely agreed on the feasible distance states. This confirms that inter-vehicle distance is a critical and robust indicator of platoon health.

3. **Limitations and Numerical Challenges:** The vertex-based method for polyhedron propagation, while more stable than a matrix-inversion approach, still faces challenges. The volumes of polyhedra in high-dimensional spaces can become vanishingly small or numerically unstable, making metrics like FDR and FNR in full 5D space less meaningful.

In conclusion, this work successfully developed a formal framework for robustness assessment and demonstrated its practical value in verifying the capabilities in directly detecting attacks. It provides not just a verification method but also actionable insights for refining the analysis and strengthening the security of autonomous driving systems.

Bibliography

[Cous92] P. Cousot, R. Cousot, Abstract interpretation frameworks, J. Logic Comput. 2 (1992) 511–547.

[Yam19] Yamaguchi, T., et al. "Application of Abstract Interpretation to the Automotive Electronic Control System." In Enea, C., Piskac, R.(eds) Verification, Model Checking, and Abstract Interpretation. VMCAI 2019. LNCS, 11388. 2019

[Ranz20] Ranzato, F., Zanella, M. "Abstract interpretation of decision tree ensemble classifiers," Proc. Of the AAAI Conf. on Artificial Intelligence, 34(4), 2020.

MATLAB, https://www.mathworks.com/products/matlab.html