ACC Case Study

No Author Given

No Institute Given

1 Evaluation

1.1 Safety verification for NNACC

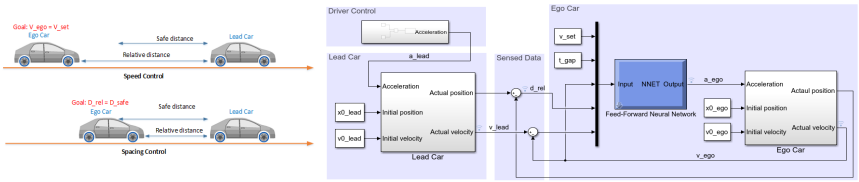


Fig. 1: Neural network adaptive cruise control system.

NNACC description. In this section, we evaluate the extension of the star-based reachability algorithm on safety verification of the NNACC system as depicted in Figure 1. The NNACC system consists of two cars in which the ego car equipped with adaptive cruise control (ACC) has a radar sensor to measures the distance to the lead car in the same lane, D_{rel}, as well as the relative velocity of the lead car, V_{rel}. The system operates in two modes including speed control and spacing control. In speed control mode, the ego car travels at a driver-set speed $V_{set} = 30$ while in spacing control mode, the ego car maintains a safe distance from the lead car, D_{safe}. If $D_{rel} \geq D_{safe}$, then speed control mode is active. Otherwise, spacing control mode is active [**?**]. Neural network adaptive cruise controllers with different sizes are trained to replace the existing MPC controller for the cars with linear motion dynamics. The control period is selected as 0.1 seconds. We analyze the closed-loop control system with the neural network controllers for the cars with nonlinear dynamics to justify two aspects: 1) the closed-loop system is safe (in a bounded time interval) and 2) the neural network controllers adapt well with the below nonlinear car models.

$$\dot{x}_{lead}(t) = v_{lead}(t),\ \dot{v}_{lead}(t) = \gamma_{lead},\ \dot{\gamma}_{lead}(t) = -2\gamma_{lead}(t) + 2a_{lead} - \mu v_{lead}^2(t),$$
$$\dot{x}_{ego}(t) = v_{ego}(t),\ \dot{v}_{ego}(t) = \gamma_{ego},\ \dot{\gamma}_{ego}(t) = -2\gamma_{ego}(t) + 2a_{ego} - \mu v_{ego}^2(t),$$

where $x_{lead}(x_{ego})$, $v_{lead}(v_{ego})$ and $\gamma_{lead}(\gamma_{ego})$ are the position, velocity and actual acceleration of the lead (ego) car respectively, $a_{lead}(a_{ego})$ is the acceleration

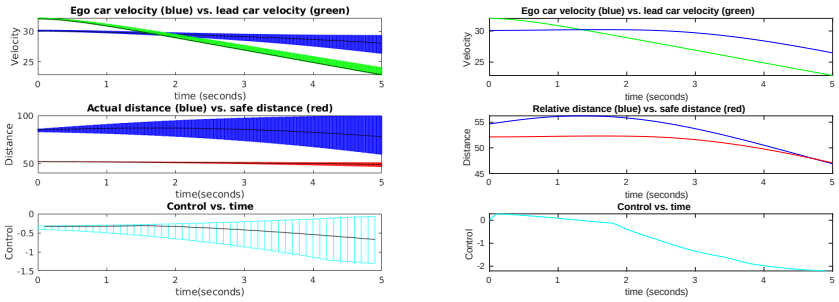
control input applied to the lead (ego) car, and $\mu = 0.0001$ is the friction parameter.

Safety-related scenario. The safety verification scenario of interest is that when the ego is in the speed control mode and the two cars are running with a safe distance between them, the lead car driver suddenly de-accelerate with $a_{lead} = -2$ to reduce the speed. We expect that the neural network controllers will also de-accelerate the ego car to remain a safe distance between two cars. Formally, the safety specification of the system is $D_{rel} = x_{lead} - x_{ego} \geq D_{safe} = D_{default} + T_{gap} \times v_{ego}$, where $T_{gap} = 1.4$ seconds and $D_{default} = 10$. We want to check if there is a collision in the next 5 seconds after the lead car de-accelerate. The initial conditions of the system are: $x_{lead}(0) \in [90, 110]$, $v_{lead}(0) \in [32, 32.2]$, $\gamma_{lead}(0) = \gamma_{ego}(0) = 0$, $v_{ego}(0) \in [30, 30.2]$, $x_{ego} \in [10, 11]$. Note that the initial distance between two cars and their velocities are chosen such that if the controllers do not adapt well with this situation, the ego car can collide with the lead car in 5 seconds. We partition the initial position of the lead car into ten smaller ranges and investigate the safety of the system corresponding to these ranges. The partition aims at reducing the accumulation of the over-approximation error in reachable set computation due to large input set.

$\mathbf{x_{lead}(0)}$	**Controller 1 (3x20)**		**Controller 2 (5x20)**		**Controller 3 (7x20)**		**Controller 4(10x20)**	
	Result	VT (sec)	Result	VT (sec)	Result	VT (sec)	Result	VT (sec)
[108, 110]	safe	211.84	safe	292.67	safe	398.41	safe	1762
[106, 108]	safe	210.16	safe	288.83	safe	393.35	safe	2270.3
[104, 106]	safe	211.54	safe	302.31	safe	412.81	safe	2674.5
[102, 104]	safe	215.21	safe	292.94	safe	446.47	safe	2863.8
[100, 102]	safe	222.87	safe	294.81	safe	440.94	safe	2606
[98, 100]	safe	233.02	safe	302.74	safe	491.29	uncertain	2855
[96, 98]	safe	237.12	safe	289.87	safe	515.43	uncertain	3249.9
[94, 96]	safe	238.46	safe	301.99	uncertain	571.75	uncertain	3851.5
[92, 94]	safe	259.46	safe	325.51	uncertain	598.22	uncertain	3220.2
[90, 92]	uncertain	265.29	safe	359.92	uncertain	558.65	uncertain	2336.9

Table 1: Verification results for NNACC system with different neural network controllers in which VT is the verification time and controller $k \times n$ means the controller has k hidden layer and n neuron per layer. The experiment is done on a desktop with following configuration: Intel Core i7-6700 CPU @ 3.4GHz $\times$ 8 Processor, 62.8 GiB Memory, 64-bit Ubuntu 16.04.3 LTS OS.

Verification results and controllers performance. The verification results are presented in Table 1 which shows that different controllers lead to different verification results. The second controller is the safest controller since it guarantees the safety of the NNACC system for the whole range of the lead car's initial position. The safety of the system can be intuitively observed via



(a) The reachable sets of the NNACC system with the second controller (5x20) and $x_{lead}(0) \in [94, 96]$.

(b) A falsification trace of the NNACC system with the first controller (3x20) and $x_{lead}(0) \in [65, 70]$.

reachable set visualization as depicted in Figure 2a showing that the relative distance reachable set does not intersect with the safe distance reachable set. In addition, we can also see that the second controller performs well in this scenario since it de-accelerates the ego car to keep the system safe. Interestingly, the controllers with large number of neurons, e.g., the third and the fourth controllers, are not necessary be the good candidates for keeping the system safe. In many cases, the verification results for these controllers are uncertain which imply that these controllers may or may not control the system safely. In these cases, the relative distance reachable set intersects with the safe distance reachable set. However, we do not know this intersection is due to the over-approximation error of the related reachable sets or the relative distance is actually smaller than the required safe distance. Since the safety of the system may be violated in these cases, we further randomly generate simulation traces of the system to find counter example inputs that make the system unsafe. If counter example inputs are found, we can conclude that the system is actually unsafe. Otherwise, we can conclude nothing about the safety of the system. The falsification results for the NNACC system using 1000 random simulations are presented in Table 2. Interestingly, we cannot find counter examples for the system whenever $x_{lead} \in [70, 110]$. However, when $x_{lead} \in [65, 70]$, we can find counter examples of the system for all controllers. Figure 2b describes a counter example to prove that the NNACC system is unsafe as $x_{lead} \in [65, 70]$ in which the relative distance between two cars is smaller than the required safe distance. Note that in this case, even the controller de-accelerates the ego car. It still can not guarantee the safety for the system.

Timing performance and scalability. As depicted in Table 1, the verification time depends on the size of the controller. A controller with large number of neurons causes a large verification time. Importantly, the experiment results show that our approach is promisingly scalable for a neural network control system with a large controller. Our approach can prove the safety of the NNACC system with the fourth controller having totally 200 neurons in some cases with

$\mathbf{x_{lead}(0)}$	**Controller 1 (3 x 20)**		**Controller 2 (5 x 20)**		**Controller 3 (7 x 20)**		**Controller 4 (10 x 20)**	
	N_c	FT (sec)	N_c	FT (sec)	N_c	FT (sec)	N_c	FT (sec)
[65, 70]	100	43.63	303	45.25	486	46.91	134	49.13
[70, 110]	0	44.14	0	45.35	0	46.82	0	49.07

Table 2: Falsification results for NNACC system with different neural network controllers using 1000 random simulations in which N_c is the number of counter examples and FT is the falsification time.

reasonable verification times (less than 1 hour). On a brief comparison with existing approaches, our approach is potentially faster and more scalable than the Verisig [**?**] and SMC-based approaches [**?**]. *Because the tools for these approaches are not available for comparison, we can only present some intuitive comparison as follows.* Verisig takes averagely 1690 seconds (on their personal computer) to verify a single safety property (corresponding to a single input set) in 30 time steps of the quadrotor system with 12 state variables and a neural network controller having 40 neurons while our approach spends averagely 275.68 seconds to verify a single safety property of the NNACC system with 6 state variables and a neural network controller having 100 neurons in 50 time steps. The SMC-based approach can falsify safety property of neural network control system with fairly large number of neurons in the controller (22 to 182 neurons). However, in the case that there is no counter example exist, the SMC-based approach usually reaches timeout (= 1 hour). Its experimental results show that only three controllers (in 17 controllers) with 22, 32 and 82 neurons are successfully verified. An important factor making our approach potentially faster and more scalable than the Verisig and SMC-based approaches is, our approach can efficiently compute the exact reachable set of DNNs on multi-core platforms. Therefore, our verification time for neural network control system can be reduced significantly by exploiting the power of parallel computing as shown in the ACAS Xu case study. We note that the new abstraction-based approach proposed recently in [**?**] is promisingly the fastest and the most scalable approach for safety verification of NNCS since it can compute the reachable set of NNCS with neural network controller of 500 neurons in 50 time steps with just 1081 seconds.