Appendix
Reachability Analysis of Neural Network Control Systems

Chi Zhang,[1] **Wenjie Ruan,** [1] **Peipei Xu** [2]

[1] University of Exeter
[2] University of Liverpool
{cz338;w.ruan}@exeter.ac.uk, peipei.xu@liverpool.ac.uk

The appendix is organised as follows:

- In Appendix A, we discuss a detailed proof of Theorem 1, which is about the Lipschitz continuity of NNCS.
- In Appendix B, we provide the pseudo-code for the Lipschitz optimisation. With optimisation, we can estimate the output range.
- In Appendix C, we prove the convergence of the Lipschitz optimisation.
- In Appendix D, we provide a case study of adative cruise control (ACC). We verify whether the NNCS is safe to maintain the relative distance of two cars. It illustrates how DeepNNC works when verifying complex systems.
- In Appendix E, we provide the benchmarks of 6 odes, which is used for the comparison experiments.
- In Appendix F, we provide the technical details of the airplane case study. The complex system contains 12 states and multiple equations, using a six-dimensional NNC to control the movement, gesture, and location of the air plane.

A. Lipschitz Continuity of NNCSs

In this section, we provide a detailed proof of Theorem 1 about the Lipschitz continuity of NNCS. We recall Theorem 1

Theorem 1 (Lipschitz Continuity of NNCS)**.** *Given an initial state X_0 and a neural network controlled system $\phi(x,t)$ with controller $\sigma(x)$ and plant $f(x,u)$, if f and σ are Lipschitz continuous, then system $\phi(x,t)$ is Lipschitz continuous in initial states $\mathcal{X}_0$. A real constant $K \geq 0$ exists for any time point $t > 0$ and for all $x_0, y_0 \in \mathcal{X}_0$:*

$$|\phi(x_0,t) - \phi(y_0,t)| \leq K\,|x_0 - y_0| \tag{1}$$

The smallest K is the best Lipschitz constant for the system, denoted K_{best}.

The essential idea is to open the control loop and further demonstrate that the system $\phi(x_0,t)$ is Lipschitz continuous on x_0. To unroll the NNCS and build an equivalent open-loop system, we duplicate the controller block and the plant block at the beginning of each control step. The information of the states will be passed from the current plant $x(t)$ with $t \in [(j-1)\delta, j\delta]$, to the next controller $u(j\delta)$, and further to the plant of the next control step $x(t)$ with $t \in [(j)\delta, (j+1)\delta]$. The information stream passes only once, which avoids loop construction and builds an equivalent simple, straightforward control system, as shown in Figure 1. Note that data transformation between different control steps occurs only at the beginning of the control steps. The plant behaviour of different time steps are independent of each other, and the state information within one control time interval is generated from the identical plant block. To determine the Lipschitz continuity of the system, we give the following definition.

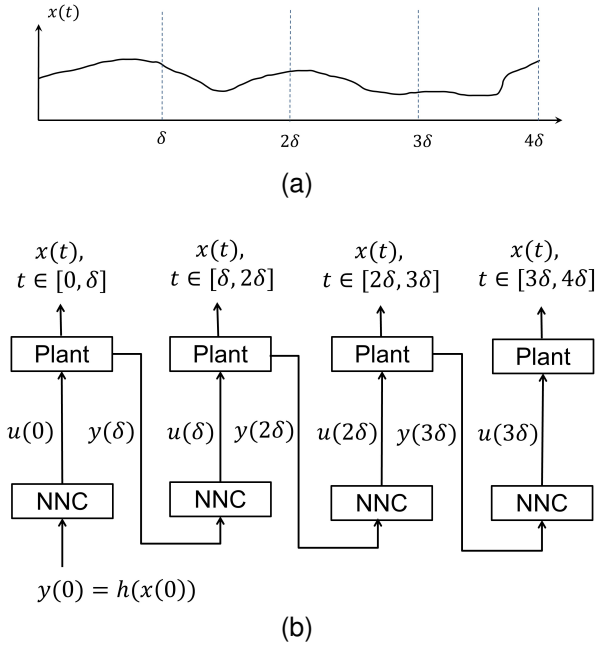

Figure 1: (a) System output of four control time steps $[0, 4\delta]$. (b) Corresponding equivalent open-loop system, which is constructed by duplication of the plant and NNC block. The *i-th* plant block gives the output of the system within the *i-th* control step.

Before we formally prove the Lipschitz continuity of NNCS, we introduce the Grönwall's Inequality (Bellman

1943).

Theorem 2 (Grönwall's Inequality (Bellman 1943))**.** Suppose that the functions $g(t), f(t)$ are continuous in the range $t \in [t_0, a]$, for $a > t_0 > 0, k(t) \geq 0$, and g obeys the inequality

$$g(t) \leq G(t) \equiv c + \int_{t_0}^{t} k(\tau)s(\tau)d\tau \tag{2}$$

Then for all $t \in [t_0, a]$, holds the inequality

$$g(t) \leq ce^{\int_{t_0}^{t} k(\tau)d\tau} \tag{3}$$

Proof. Differentiation of G in (2) gives the equation

$$\dot{G}(t) = k(t)g(t) \leq k(t)G(t) \tag{4}$$

Therefore, $\dot{G}(t) - k(t)G(t) \leq 0$. We multiply it by the positive factor $e^{\int_{t_0}^{t} k(\tau)d\tau}$ and get

$$e^{\int_{t_0}^{t} k(\tau)d\tau}(\dot{G}(t) - k(t)G(t)) = \frac{d}{dt}(G(t)e^{\int_{t_0}^{t} k(\tau)d\tau}) \leq 0 \tag{5}$$

Integration of the inequality gives

$$G(t)e^{\int_{t_0}^{t} k(\tau)d\tau} \leq G(0) \tag{6}$$

since $g \leq G$ and $G(0) = c$, we obtain

$$g(t) \leq G(t) \leq ce^{\int_{t_0}^{t} k(\tau)d\tau} \tag{7}$$

which demonstrates the Grönwall's Inequality. □

Now, we formally prove Lipschitz continuity of NNCS.

Proof. We first demonstrate the Lipschitz continuity in the time interval $t \in [0, \delta]$, and use the equivalent open-loop structure to extend the continuity to the whole time horizon. As demonstrated in Definition 3, we have $\phi(x_0, t) = x(t)$ and $\phi(y_0, t) = y(t)$, which can be generated by integration.

$$\begin{aligned} \phi(x_0, t) &= x_0 + \int_{t_0}^{t} f(x(\tau), u_x(\tau)))\, d\tau \\ \phi(y_0, t) &= y_0 + \int_{t_0}^{t} f(y(\tau), u_y(\tau)))\, d\tau \end{aligned} \tag{8}$$

Thus, the difference of the NNCS output holds the inequality

$$\begin{aligned} &|\phi(x_0, t) - \phi(y_0, t)| \leq |x_0 - y_0| \\ &+ \int_{t_0}^{t} |\, f(x(\tau), u_x(\tau)) \;-\; f(y(\tau), u_y(\tau))|\, d\tau \end{aligned} \tag{9}$$

We add the term $f(x(\tau), u_y(\tau)) \;-\; f(x(\tau), u_y(\tau))$ to the right side and further induce the formula

$$\begin{aligned} &|\phi(x_0, t) - \phi(y_0, t)| \leq |x_0 - y_0| \\ &+ \int_{t_0}^{t} |\, f(x(\tau), u_x(\tau)) \;-\; f(x(\tau), u_y(\tau))|\, d\tau \\ &+ \int_{t_0}^{t} |f(x(\tau), u_y(\tau)) \;- f(y(\tau), u_y(\tau))|\; d\tau \end{aligned} \tag{10}$$

As previously defined, the plant $\dot{u} = f(x, u)$ is Lipschitz continuous on x and u so that the system carries an unique solution (Meiss 2007). We designate the Lipschitz constant as L_u and L_x respectively and further transform Equation (10) to Equation (11) according to the Lipschtiz continuity.

$$\begin{aligned} &|\phi(x_0, t) - \phi(y_0, t)| \leq |x_0 - y_0| \\ &+ L_u \int_{t_0}^{t} |\, u_x(\tau) - u_y(\tau)\,|\; d\tau \\ &+ L_x \int_{t_0}^{t} |\, x(\tau) \;-\; y(\tau)\,|\; d\tau \end{aligned} \tag{11}$$

Researches (Ruan, Huang, and Kwiatkowska 2018) have demonstrated that the regular neural network layers are Lipschitz continuous, including Sigmoid, ReLU and Tanh. As stated in the NNCS definition, the control inputs u are decided by the states at the beginning of the control time step and fixed within δ, so that the integration containing control input u can be simplified and described as

$$\begin{aligned} L_u \int_{t_0}^{t} |u_x(\tau) - u_y(\tau)|\, d\tau &= L_u(t - t_0)\, |u(x_0) - u(y_0)| \\ &\leq L_u L_n (t - t_0)\, |x_0 - y_0| \end{aligned} \tag{12}$$

with L_n as the Lipschitz constant of the neural network controller. Substitution of Equation (12) to inequality (11) gives Equation (13).

$$\begin{aligned} |\phi(x_0, t) - \phi(y_0, t)| \leq \{L_u L_n (t - t_0) + 1\} \cdot |x_0 - y_0| \\ + L_x \int_{t_0}^{t} |\, x(\tau) - y(\tau)\,|\; d\tau \end{aligned} \tag{13}$$

Applying Grönwall's Inequality to (13) with

$$c = \{L_u L_n (t - t_0) + 1\} \cdot |x_0 - y_0|$$

and $k(\tau) = L_u$ gives

$$\begin{aligned} &|\phi(x_0, t) - \phi(y_0, t)| \\ &\leq \{L_u L_n (t - t_0) + 1\} \cdot |x_0 - y_0| \cdot e^{L_x (t - t_0)} \end{aligned} \tag{14}$$

Hence, we have demonstrated the Lipschitz continuity of the closed-loop system on their initial states with in the time interval $[0, \delta]$. We define the Lipschitz constant as L_δ and simplify Equation (14) to

$$|\phi(x_0, t) - \phi(y_0, t)| \leq L_{step}\, |x_0 - y_0| \tag{15}$$

with $L_{step} = \{L_u L_n (t - t_0) + 1\} \cdot e^{L_x (t - t_0)}$. In the second control time step $t \in [\delta, 2\delta]$, we have the analogous proving procedure, except for the replacement of x_0, y_0 and t_0 by x_δ, y_δ and δ on the right side of Equation (14). According to Equation (15), $|x_\delta - y_\delta| \leq L_\delta\, |x_0 - y_0|$, with $L_\delta = (L_u L_n \delta + 1) \cdot e^{L_x \delta}$. Thus, for the second time step, the inequality holds.

$$\begin{aligned} &\|\phi(x_0, t) - \phi(y_0, t)| \\ &\leq L_\delta \{L_u L_n (t - \delta) + 1\} \cdot |x_0 - y_0| \cdot e^{L_x (t - \delta)} \end{aligned} \tag{16}$$

We expand the proving process to the whole time range based on Figure 1 and develop the inequality for the entire time range,

$$\begin{aligned}&|\phi(x_0,t)-\phi(y_0,t)|\\ \leq& L_\delta^j\{L_uL_n(t-j\delta)+1\}\cdot|x_0-y_0|\cdot e^{L_x(t-j\delta)}\end{aligned} \tag{17}$$

with $t\in[j\delta,(j+1)\delta]$, $j=0,1,2,3....$ □

So far, we have demonstrated the Lipschitz continuity of NNCS in the initial input range, allowing the use of Lipschitz optimization approaches in NNCS. Given the Lipschitz continuity of plant $f(x,u)$ and the controller $u(x)$, we prove the Lipschitz continuity of the entire system ϕ. To guarantee the existence of a unique solution in ODE form, we require the function of the plant f to be Lipschitz continuous in x and u (Meiss 2007). The authors in (Szegedy et al. 2013; Ruan, Huang, and Kwiatkowska 2018) demonstrated that deep neural networks with convolutional, max-pooling layer and fully-connected layers with ReLU, Sigmoid activation function, Hyperbolic Tangent, and Softmax activation functions are Lipschitz continuous. Thus, most of the practical neural network control systems are Lipschitz continuous.

B. Algorithm

In the optimization process, the input range, the function of NNCS system $\phi(x)$, maximum iteration number k_{max} are given as inputs. We define $\Delta=\hat{\phi}_{min}-R_{min}$ and give $\Delta=10\epsilon$ as the initial value to start the iteration. Δ decreases with the number of iterations. Optimisation is terminated when iteration is greater than k_{max} or Δ is smaller than the allowable error ϵ. The R_{min} and $\hat{\phi}_{min}$ of the last iteration are returned as output. We use the character value R_{min} for the estimation of the output range.

Algorithm 1: Lipschitz Optimization

Input: Input range $[p,q]$ of data x, function $\phi(x)$, ϵ, maximal iteration number k_{max},
Output: Upper bound ϕ_{min} and lower bound R_{min}
Initialize $a_0=p,a_1=q,k=0,j=0,\Delta=10\epsilon$.
while $k\leq k_{max}$ and $\Delta\geq\epsilon$ **do**
  Add a_{new} to $[a_j,a_{j+1}]$ subinterval
  Update $\{a_0,a_1,...\}$
  Calculate $\phi_{new}=\phi(a_{new})$, Update $\{\phi_0,\phi_1...\}$
  Calculate $\hat{\phi}_{min}=min\{\phi_0,\phi_1,...\}$
  Update $\{R_0,R_1...\}$
  $R_{min}=min\{R_0,R_1,...\}$, $j=\text{argmin}\{R_0,R_1,...\}$
  $\Delta=\hat{\phi}_{min}-R_{min}$
end while
Return $\hat{\phi}_{min}$ and R_{min}

C. Convergence Analysis

In the one-dimensional problem, the optimisation will converge under two conditions: 1. $\lim_{k\to\infty}\hat{\phi}_{min}-R_{min}^k=0$; 2. $\lim_{k\to\infty}R_{min}^k=\phi_{min}$. In the k-th iteration, we divide the sub-interval $D_j=[a_j,a_{j+1}]$ into $D_{new}^1=[a_j,x_j^R]$ and $D_{new}^2=[x_j^R,a_{j+1}]$. The new character value

$$\begin{aligned}R_{new}^1-R_j&=l_j\frac{a_{j+1}-x_j^R}{2}-\frac{\phi_{j+1}-\phi(x_j^R)}{2}>0\\ R_{new}^2-R_j&=l_j\frac{x_j^R-a_j}{2}-\frac{\phi(x_j^R)-\phi_j}{2}>0\end{aligned} \tag{18}$$

Since $R_{min}^k=\min\{R_0,...R_n\}\setminus\{R_j\}\}\cup\{R_{new}^1,R_{new}^2\}$, we confirm that R_{min}^k increases strictly monotonically and is bounded. Convergence conditions are met.

In the multidimensional problem, the problem is transformed to a one-dimensional problem with nested optimisation. We introduce Theorem 3 for the inductive step.

Theorem 3. *In optimisation, if* $\forall x\in\mathbb{R}^d$, $\lim_{k\to\infty}R_{min}^k=\inf_{x\in[a,b]^d}\phi(x)$ *and* $\lim_{i\to\infty}(\hat{\phi}_{min}^k-R_{min}^k)=0$ *are satisfied, then* $\forall x\in\mathbb{R}^{d+1}$, $\lim_{k\to\infty}R_{min}^k=\inf_{x\in[a,b]^{d+1}}\phi(x)$ *and* $\lim_{k\to\infty}(\hat{\phi}_{min}^k-R_{min}^k)=0$ *hold.*

Proof. By the nested optimisation scheme, we have

$$\min_{\mathbf{x}\in[a_i,b_i]^{d+1}}\phi(\mathbf{x})=\min_{x\in[a,b]}W(x);\quad W(x)=\min_{\mathbf{y}\in[a_i,b_i]^d}\phi(x,\mathbf{y}) \tag{19}$$

Since $\min_{\mathbf{y}\in[a_i,b_i]^d}\phi(x,\mathbf{y})$ is bounded by an interval error $\epsilon_{\mathbf{y}}$, assuming that $W^*(x)$ is the accurate global minimum, then we have $W^*(x)-\epsilon_{\mathbf{y}}\leq W(x)\leq W^*(x)+\epsilon_{\mathbf{y}}$. So the $d+1$-dimensional problem is reduced to the one-dimensional problem $\min_{x\in[a,b]}W(x)$. The difference from the real one-dimensional case is that the evaluation of $W(x)$ is not accurate, but bounded by $|W(x)-W^*(x)|\leq\epsilon_{\mathbf{y}},\forall x\in[a,b]$, where $W^*(x)$ is the accurate function evaluation.

Assume that the minimal value obtained from our method is $W_{min}^*=\min_{x\in[a,b]}W^*(x)$ under accurate function evaluation, with corresponding R_{min}^{k*} and $\widehat{W}_{min}^{k*}$. For the inaccurate evaluation case, we have $W_{min}=\min_{x\in[a,b]}W(x)$, R_{min}^k and $\widehat{W}_{min}^k$. The termination criteria for both cases are $|\widehat{W}_{min}^k-R_{min}^k|\leq\epsilon_x$ and $|\widehat{W}_{min}^{k*}-R_{min}^{k*}|\leq\epsilon_x$, and w^* represents the ideal global minimum. Then we have $w^*-\epsilon_x\leq R_{min}^k$. Assuming that R_{min}^{k*} is the character value of $[x_j,x_{j+1}]$ and we have

$$w^*-\epsilon_x\leq R_{min}^{k*}=\frac{W^*(x_j)+W^*(x_{j+1})}{2}-l\frac{(x_{j+1}-x_j)}{2} \tag{20}$$

Since $|W(x_i)-W^*(x_i)|\leq\epsilon_{\mathbf{y}}$, and $\forall i=j,j+1$

$$w^*-\epsilon_x\leq\frac{W(x_j)+W(x_{j+1})}{2}+\epsilon_y-\frac{l(x_{j+1}-x_j)}{2} \tag{21}$$

Based on the definition of R_{min}^k, we have $w^*-R_{min}^k\leq\epsilon_{\mathbf{y}}+\epsilon_x$. In analogy, we can get $\widehat{W}_{min}^k-w^*\leq\epsilon_{\mathbf{y}}+\epsilon_x$. Thus, the accurate global minimum is bounded. Theorem 3 is proved. □

D. Case Study: Adaptive Cruise Control

We consider an ACC model, which was originally introduced in (Tran et al. 2020), consisting of two vehicles and an NNC. The structure of the system is shown in Figure 2. The

ego car is equipped with a sensor that measures the relative distance and the velocity. The NNC controls the movement of the ego car by influencing its acceleration. The goal is to maintain a safe distance between the two cars, with the ego car travelling at a driver set speed V_{set}.

There are six states $\{x_1, x_2, x_3, x_4, x_5, x_6\}$ involved, with $\{x_1, x_2, x_3\}$ the position, velocity, and acceleration of the leading car and $\{x_4, x_5, x_6\}$ of the ego car. The controller is a neural network with three hidden layers containing 20 neurons each. The NNC has five inputs, with two fixed parameters $V_{set} = 30$ and a time gap between vehicles $T_{gap} = 1.4$, and three dynamic variables. The three updated variables $x_5, x_1 - x_4, x_2 - x_5$ are the velocity of the ego car, the relative position, and the relative velocity, all measured from the ego car. The controller takes the five inputs and gives an output u, which feeds back to the ego car and further updates the car dynamic. The dynamic of the two vehicles is described as ODEs,

$$\dot{x_1} = x_2, \quad \dot{x_2} = x_3, \quad \dot{x_3} = 2a - 2x_3 - \mu(x_2)^2 \tag{22}$$

$$\dot{x_4} = x_5, \quad \dot{x_5} = x_6, \quad \dot{x_6} = 2u - 2x_6 - \mu(x_5)^2 \tag{23}$$

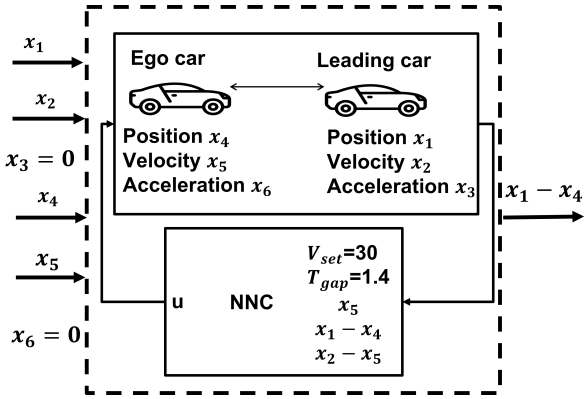

Figure 2: A NNC controlled ACC model. The NNC takes 5 inputs from the sensor and generates a control input u to regulate the ego car. The acceleration of the ego car and leading car are 0 at the beginning. We check if the relative distance $x_1 - x_4$ is safe in a time horizon, when the initial position and velocity x_1, x_2, x_3, x_4 stay in a predefined range.

The entire closed-loop system is presented inside the dashed line box in Figure 2. In our approach, we can neglect the detailed structure inside the dashed box and treat it as a black-box system. Since the initial states of x_3 and x_6 are fixed with $x_3^0 = 0$ and $x_6^0 = 0$, the whole system is regarded as a function $\phi_t(x_1^0, x_2^0, x_4^0, x_5^0) = x_4^t - x_1^t$. The reachable set estimation of relative distance at time t is treated as a four-dimensional optimization problem.

$$\begin{aligned} min \quad & \phi_t(x_1^0, x_2^0, x_4^0, x_5^0) \\ subject\ to \quad & 91 \geq x_1 \geq 90, 32.05 \geq x_2 \geq 32, \\ & 11 \geq x_4 \geq 10, 30.05 \geq x_5 \geq 30. \end{aligned} \tag{24}$$

We use Lipschitz optimization to solve problem (24), acquiring ϕ_{tmin}. The maximal value ϕ_{tmax} is estimated in the same way. The reachable set calculated for the relative distance at time t is $[\phi_{tmin}, \phi_{tmax}]$.

We want to analyze safety, that is, the relative distance, in the circumstance where the leading car suddenly slows down with $a = -2$. The friction coefficient is $\mu = 0.0001$. Initial sets are $x_1 \in [90, 91], x_2 \in [32, 32.05], x_3 = 0, x_4 \in [10, 11], x_5 \in [30, 30.05], x_6 = 0$. Since the ego car is in a cruise model, we want its speed to change around the driver set velocity $V_{set} = 30m/s$.

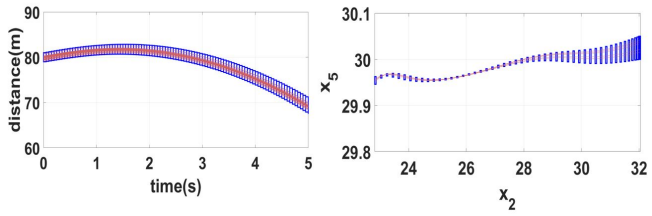

Figure 3: (a) The relative distance between two cars. The blue reachable sets cover the red random simulated trajectories. The relative distance is greater than $60m$ within $5s$. The ACC model is safe (b) The velocity x_5 of the ego car remains around $30m/s$ with the changes of the leading car velocity x_2

The reachable sets of relative distance within $5s$ are shown in Figure 3 (a). The blue trajectory represents the estimated reachable sets, and the red is the random simulated results. The estimated results cover all simulated trajectories with a tight band, and the relative distance is above $60m$ within $5s$. We can confirm that the ACC is safe in 5 seconds, without the danger of collision. We also analyze the leading car velocity x_2 and the ego car velocity x_5 in Figure 3 (b). We affirm that the ego car travels around the speed of 30m/s despite the leading car's speed variance.

E. Comparison with Baseline Methods

The benchmarks include dynamic plants bearing 2 to 4 state variables, which can be described with ordinary differential equations. The controller is a fully connected neural network with k layer and j hidden neurons. Various activation functions are considered, including ReLU, Sigmoid, and Tanh combinations of ReLU and Tanh, since DeepNNC only specifies the Lipschitz continuity, and the regular neural network structures fulfil the requirement. Given an input range of initial states, we estimate the reachable set at $t = 6s$ with a control step $\delta = 0.2s$. For the hyperparameter in our method, we set the $\epsilon = 0.005$ and maximal trial number $k_{max} = 10^4$.

F. Case study: airplane

We analyze a complex control system of a flying airplane, which is a benchmark in ARCH-COMP21 (Johnson et al. 2021). The flying airplane is shown in Figure 4, which contains 12 states $[x, y, z, u, v, w, \alpha, \beta, \gamma, r, p, q]$. (x, y, z) indicates the position, and (α, β, γ) are the Euler angles that describe the gesture. $[u, v, w]$ are the components of velocity

Table 1: Benchmark Setting: B1-B6 are the benchmarks index. N denotes the dimension of NNC input. NNC denotes the size of the NN by layer number $\times$ neuron number. ODE is the ordinary differential equation for plant dynamics. The allowable perturbed ranges of initial states are in the last column.

	N	NNC	ODE	Initial states
B1	2	2 $\times$20	$\dot{x_1} = x_2$ $\dot{x_2} = ux_2{}^2 - x_1$	$x_1 \in [0.8, 0.9]$ $x_2 \in [0.5, 0.6]$
B2	2	2 $\times$20	$\dot{x_1} = x_2 - x_1{}^3$ $\dot{x_2} = u$	$x_1 \in [0.7, 0.9]$, $x_2 \in [0.7, 0.9]$
B3	2	2 $\times$20	$\dot{x_1} = -x_1(0.1 + (0.1 + 0.2)^2$ $\dot{x_2} = (u + x_1)(0.1 + (x_1 + x_2)^2)$.	$x_1 \in [0.8, 0.9]$ $x_2 \in [0.4, 0.5]$
B4	3	2 $\times$20	$\dot{x_1} = -x_1 + x_2 - x_3$, $\dot{x_2} = -x_1(x_3 + 1) - x_2$, $\dot{x_3} = -x_1 + u$	$x_1 \in [0.25, 0.27]$, $x_2 \in [0.08, 0.1]$, $x_3 \in [0.25, 0.27]$
B5	3	4 $\times$100	$\dot{x_1} = x_1{}^3 - x_2$ $\dot{x_2} = x_3, \dot{x_3} = u$	$x_1 \in [0.38, 0.4]$, $x_2 \in [0.45, 0.7]$ $x_3 \in [0.25, 0.7]$
B6	4	3 $\times$20	$\dot{x_1} = x_2$, $\dot{x_2} = -x_1 + 0.1\sin(x_3)$, $\dot{x_3} = x_4, \dot{x_4} = u$	$x_1 \in [-0.77, -0.75]$, $x_2 \in [-0.45, -0.43]$ $x_3 \in [0.51, 0.54]$,$x_4 \in [-0.3, -0.28]$

in the three (x, y, z) directions and (p, q, r) are the body rotation rates.

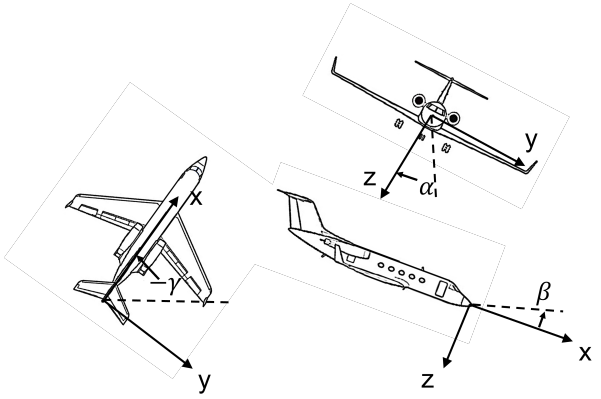

Figure 4: Demonstration of position and gesture of an air plane. The coordinates x, y, z describe the position of the plane, the Euler angles α, β, γ describe the gesture.

After normalization for simplicity, we apply the mass of airplane $m = 1$ and the moment of inertia $I_x = I_y = I_z = 1, I_{xz} = 0$. The initial position and gesture angles are zero and the velocity and rotation rates can change according to Equation (25)

$$\begin{aligned}
\dot{u} &= -g\sin\beta + F_x/m - qw + rv \\
\dot{v} &= g\cos\beta\sin\alpha + F_y/m - ru + pw \\
\dot{w} &= g\cos\beta\cos\alpha + F_z/m - pv + qu \\
I_x\dot{p} + Ixz\dot{r} &= M_x - (I_z - I_y)qr - Ixzpq \\
I_y\dot{q} &= M_y - Ixz(r^2 - p^2) - (I_x - I_z)pr \\
Ixz\dot{p} + I_z\dot{r} &= M_z - (I_y - I_x)gp - I_{xz}rq
\end{aligned} \tag{25}$$

and kinematic equations (26)(27)

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{z}\end{bmatrix} = \begin{bmatrix}\cos\gamma - \sin\gamma\, 0\\ \sin\gamma\ \ \cos\gamma\ \ 0\\ 0\ \ \ 0\ \ \ 1\end{bmatrix} \begin{bmatrix}\cos\beta\ \ 0\sin\beta\\ 0\ \ \ 1\ \ \ 0\\ -\sin\beta\, 0\cos\beta\end{bmatrix} \begin{bmatrix}1\ \ 0\ \ \ 0\\ 0\cos\alpha - \sin\alpha\\ 0\sin\alpha\ \ \cos\alpha\end{bmatrix} = \begin{bmatrix}u\\ v\\ w\end{bmatrix} \tag{26}$$

$$\begin{bmatrix}\alpha\\ \beta\\ \gamma\end{bmatrix} = \begin{bmatrix}1 & \tan\beta\sin\alpha & \tan\beta\cos\alpha\\ 0 & \cos\alpha & \sin\alpha\\ 0 & \sec\beta\sin\alpha & \sec\beta\cos\alpha\end{bmatrix} \begin{bmatrix}p\\ q\\ r\end{bmatrix} \tag{27}$$

The initial states and input ranges are

$$\begin{aligned}
x = y = z = r = p = q = 0, \\
[u, v, w, \alpha, \beta, \gamma] = [0, 1]^6
\end{aligned} \tag{28}$$

The goal set is $y \in [-0.5, 0, 5]$ and $[\alpha, \beta, \gamma] = [-1, 1]^3$ when $t < 2s$. The controller here takes an input of 12 dimensions and generates a six-dimensional control input $F_x, F_y, F_z, M_x, M_y, M_z$. When handling the model as an optimization problem, we treat the inner structure of the model as a black box. We only consider the perturbed initial states $[u, v, w, \alpha, \beta, \gamma] = [0, 1]^6$ and the problem is transformed to a six-dimensional black-box optimization problem. For simplicity, we choose a small subset of the initial input $[u, v, w, \alpha, \beta, \gamma] = [0.9, 1]^6$.

Our approach can handle large and complex systems. Unlike other baseline methods, which need a layer-by-layer analysis of the NNC, we take the whole closed-loop system as a black box. We only consider the changeable initial states as input to the optimization problem and solve the problem through Lipschitz optimization.

nvergence of the one-dimensional case and Theorem 3 of the inductive step, we prove the convergence of the multidimensional case.

References

Bellman, R. 1943. The stability of solutions of linear differential equations. *Duke Mathematical Journal*, 10(4): 643–647.

Johnson, T. T.; Lopez, D. M.; Benet, L.; Forets, M.; Guadalupe, S.; Schilling, C.; Ivanov, R.; Carpenter, T. J.; Weimer, J.; and Lee, I. 2021. ARCH-COMP21 Category Report: Artificial Intelligence and Neural Network Control Systems (AINNCS) for Continuous and Hybrid Systems Plants. In Frehse, G.; and Althoff, M., eds., *8th International Workshop on Applied Verification of Continuous and Hybrid Systems (ARCH21)*, volume 80 of *EPiC Series in Computing*, 90–119. EasyChair.

Meiss, J. D. 2007. *Differential dynamical systems*. SIAM.

Ruan, W.; Huang, X.; and Kwiatkowska, M. 2018. Reachability analysis of deep neural networks with provable guarantees. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI'18)*, 2651–2659.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Tran, H.-D.; Yang, X.; Lopez, D. M.; Musau, P.; Nguyen, L. V.; Xiang, W.; Bak, S.; and Johnson, T. T. 2020. NNV: The neural network verification tool for deep neural networks and learning-enabled cyber-physical systems. In *International Conference on Computer Aided Verification*, 3–17. Springer.