

Benchmark Examples

1 RL training details

During the training phase, we initially tried to train a DNN with 1-3 hidden layers with the same hyperparameters and found that the sampled trajectories were difficult to get into the target set. Therefore, we continue to expand the network structure, and finally achieve good results when the hidden layer is increased to 5 layers with 128 nodes per layer and with ReLu activation function. Both Actor network and Critic network are the same above structure. Concretely, Actor learning rate is $1e^{-4}$, critic learning rate is $2e^{-4}$, and reward decay coefficient gamma is 0.99. The delay update between the target network and the predict network of the DDPG algorithm adopts the soft method, and each update range is $\tau=0.1$; batch size=128 and replay buffer size=5000. The discrete time step of the continuous system is $t=0.05s$, and the maximum duration is $T=150s$.

The condition for the end of collecting a sequence is: starting from the initial area, (i) if the number of steps exceeds 3000 steps, that is, the total time exceeds 150s, then the collection of the track data will be stopped. (ii) If the track enters the unsafe area, stop collecting. (iii) If the trajectory almost stops moving after entering the target area, it is considered to have little contribution to the subsequent algorithm training. Therefore, if the variance of the state x of the detected 50 sample points at the end of the trajectory is less than $1e^{-5}$, the collection is also terminated early. We set 25 trajectories as 1 epoch, train 100 epochs, and finally we take the network parameters of the epoch with the highest average reward as the final parameters.

Finally, we only need to use the parameters obtained by $r(x)$ network as the initial parameters to train a small DNN controller to ensure the safety and reachability of the system. With a well-trained initialization parameter, the training efficiency of small networks will be greatly improved, so we only need to sample a small amount of data to fine-tune the network. In this example, we use a neural network with 30 node hidden layers to achieve good results. At the same time, in order to fully ensure the security and reachability of the controller, we choose to take the initial sample points on the boundary of the initial region, that is the points on the boundary are most likely to violate the security or reachability properties. Concretely, the learning rates of Actor and Critical networks are set to $2e^{-5}$ and $4e^{-5}$, and the number of training rounds is set to 10 epochs. In this way, the original controller will be expressed as the

sum of a polynomial controller p and a DNN controller k .

2 Results

2.1 Pendulum system [4]

The equation of motion for Pendulum system is

$$\ddot{\alpha} = -\frac{g}{l}\sin(\alpha) - \frac{d}{ml^2}\dot{\alpha} + \frac{u}{ml^2}$$

with the constants set as $g = 10, l = 1, m = 1, d = 0.1$, and the state is defined as $\mathbf{x} = [\alpha, \dot{\alpha}]$. The system domain, initial set, unsafe set and target set are as follows:

$$\Psi = \{\mathbf{x} \in \mathbb{R}^2 \mid [-\pi, -5]^T \leq \mathbf{x} \leq [\pi, 5]^T\}$$

$$X_0 = \{\mathbf{x} \in \mathbb{R}^2 \mid 0 \leq \|\mathbf{x}\|_2 \leq 2\}$$

$$X_u = \{\mathbf{x} \in \mathbb{R}^2 \mid 2.5 \leq \|\mathbf{x}\|_2 \leq 3\}$$

$$X_g = \{\mathbf{x} \in \mathbb{R}^2 \mid 0 \leq \|\mathbf{x}\|_2 \leq 0.1\}.$$

We train a DNN of 4 hidden layers with 128 nodes in each layer through reinforcement learning. The polynomial $p(\mathbf{x})$ after approximation are as follows:

$$\begin{aligned} p(\mathbf{x}) = & -0.72054 - 2.66857x_1 - 10.6991x_2 - 0.13387x_1^2 + 1.31456x_1x_2 \\ & + 1.04989x_2^2 \end{aligned}$$

The small NN $k(\mathbf{x})$ is a single hidden layer neural network with 20 nodes and relu activation function. We give the parameters of this network in the form of TensorFlow code in the following link: <https://juzi1.github.io/RL/SNN/C1.py>.

Under the constructed hybrid controller $p(\mathbf{x}) + k(\mathbf{x})$, the closed-loop system can be verified to satisfy the safety and goal-reaching properties by the following computed barrier certificate $B(\mathbf{x})$ and Lyapunov-like $V(\mathbf{x})$ function respectively.

$$B(\mathbf{x}) = -0.118x_1^2 - 0.034x_1x_2 - 0.015x_1 - 0.129x_2^2 - 0.031x_2 + 0.633,$$

$$V(\mathbf{x}) = 6.1379x_1^2 + 10.7155x_1x_2 - 0.0051x_1 + 13.9549x_2^2 + 0.0068x_2 - 0.1711.$$

2.2 Vehicle path tracking system [2]

The equation of motion for Vehicle path tracking system is

$$\begin{aligned} \dot{s} &= \frac{v \cos(\theta_e)}{1 - d_e k(s)} \\ \dot{d}_e &= v \sin(\theta_e) \\ \dot{\theta}_e &= \frac{v \tan(u)}{L} - \frac{v k(s) \cos(\theta_e)}{1 - d_e k(s)} \end{aligned}$$

with the constants set as $v = 6, L = 1$ and the state is defined as $\mathbf{x} = [\dot{d}_e, \dot{\theta}_e]$.

The system domain, initial set, unsafe set and target set are as follows:

$$\Psi = \{\mathbf{x} \in \mathbb{R}^2 \mid [-0.8, 0.8]^T \leq \mathbf{x} \leq [0.8, 0.8]^T\}$$

$$X_0 = \{\mathbf{x} \in \mathbb{R}^2 \mid 0 \leq \|\mathbf{x}\|_2 \leq 0.5\}$$

$$X_u = \{\mathbf{x} \in \mathbb{R}^2 \mid 0.6 \leq \|\mathbf{x}\|_2 \leq 0.8\}$$

$$X_g = \{\mathbf{x} \in \mathbb{R}^2 \mid 0 \leq \|\mathbf{x} - \mathbf{x}_g\|_2 \leq 0.2\} \text{ with } \mathbf{x}_g = [-0.2, 0]^T.$$

We train a DNN of 4 hidden layers with 64 nodes in each layer through reinforcement learning. The polynomial $p(\mathbf{x})$ after approximation are as follows:

$$p(\mathbf{x}) = -0.286x_1 - 1.3352x_2 + 0.2347x_1^2 + 0.3372x_1x_2 - 0.1027x_2^2.$$

The small NN $k(\mathbf{x})$ is a single hidden layer neural network with 20 nodes and relu activation function. We give the parameters of this network in the form of TensorFlow code in the following link: <https://juzi1.github.io/RL/SNN/C2.py>.

Under the constructed hybrid controller $p(\mathbf{x}) + k(\mathbf{x})$, the closed-loop system can be verified to satisfy the safety and goal-reaching properties by the following computed barrier certificate $B(\mathbf{x})$ and Lyapunov-like $V(\mathbf{x})$ function respectively.

$$B(\mathbf{x}) = -0.6573x_1^2 + 0.0128x_1x_2 + 0.008x_1 - 0.812x_2^2 + 0.012x_2 + 0.2239,$$

$$V(\mathbf{x}) = -29.8168x_1^2 + 26.4377x_1x_2 - 11.9429x_1 - 29.8581x_2^2 + 5.305x_2 - 0.5327.$$

2.3 Cartpole system [4]

The equation of motion for cartpole system is

$$\begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{u + m_p \sin \theta (l \dot{\theta}^2 - g \cos \theta)}{m_c + m_p \sin^2 \theta} \\ \frac{u \cos \theta + m_p l \dot{\theta}^2 \cos \theta \sin \theta - (m_c + m_p) g \sin \theta}{l(m_c + m_p \sin^2 \theta)} \end{bmatrix}$$

with the constants set as $m_c = 1, m_p = 1, g = 1, l = 1$ and the state is defined as $\mathbf{x} = [x, \theta, \dot{x}, \dot{\theta}]$.

The system domain, initial set, unsafe set and target set are as follows:

$$\Psi = \{\mathbf{x} \in \mathbb{R}^4 \mid [-1.3, -1.3, -1.3, -1.3]^T \leq \mathbf{x} \leq [1.3, 1.3, 1.3, 1.3]^T\}$$

$$X_0 = \{\mathbf{x} \in \mathbb{R}^4 \mid 0.0 \leq \|\mathbf{x}\|_2 \leq 0.7\}$$

$$X_u = \{\mathbf{x} \in \mathbb{R}^4 \mid 1.0 \leq \|\mathbf{x}\|_2 \leq 1.3\}$$

$$X_g = \{\mathbf{x} \in \mathbb{R}^4 \mid 0.0 \leq \|\mathbf{x}\|_2 \leq 0.1\}.$$

We train a DNN of 5 hidden layers with 128 nodes in each layer through reinforcement learning. The polynomial $p(\mathbf{x})$ after approximation are as follows:

$$\begin{aligned} p(\mathbf{x}) = & -1.1237x_1 - 0.1558x_2 - 2.1002x_3 - 1.0579x_4 - 0.3102x_1^2 - 0.121x_1x_2 \\ & - 1.1061x_1x_3 - 0.2505x_1x_4 + 0.2049x_2^2 - 0.112x_2x_3 - 0.4603x_2x_4 \\ & - 0.5411x_3^2 - 0.4735x_3x_4 - 0.2273x_4^2 \end{aligned}$$

The small NN $k(\mathbf{x})$ is a single hidden layer neural network with 40 nodes and relu activation function. We give the parameters of this network in the form of TensorFlow code in the following link: <https://juzi1.github.io/RL/SNN/C3.py>.

Under the constructed hybrid controller $p(\mathbf{x}) + k(\mathbf{x})$, the closed-loop system can be verified to satisfy the safety and goal-reaching properties by the following computed barrier certificate $B(\mathbf{x})$ and Lyapunov-like $V(\mathbf{x})$ function respectively.

$$\begin{aligned} B(\mathbf{x}) = & \underbrace{-0.1666x_1^4 + 0.1745x_1^3x_2 - 0.3928x_1^3x_3 \cdots + 0.0052x_4 + 0.1428}_{70 \text{ terms}}, \\ V(\mathbf{x}) = & \underbrace{0.0836x_1^4 - 0.1599x_1^3x_2 + 0.2863x_1^3x_3 + \cdots + 0.0052x_4 + 0.1428}_{70 \text{ terms}} \end{aligned}$$

You can click the following link to view the complete parameters.

For $B(\mathbf{x})$: <https://juzi1.github.io/RL/Barriers/C3.txt>

For $V(\mathbf{x})$: <https://juzi1.github.io/RL/Lyapunov/C3.txt>

2.4 UAV control [4]

The motion of equation for a UAV flying in planar [4] is given by

$$\begin{aligned} \ddot{x} &= \frac{-(u_1 + u_2) \sin \theta}{m} \\ \ddot{y} &= \frac{(u_1 + u_2) \cos \theta - mg}{m} \\ \ddot{\theta} &= \frac{r(u_1 - u_2)}{I} \end{aligned}$$

with the constants set as $m = 0.1, g = 0.1, I = 0.1, r = 0.1$, and the state is defined as $\mathbf{x} = [x, y, \theta, \dot{x}, \dot{y}, \dot{\theta}]^T$, and the control variable is defined as $\mathbf{u} = [u_1, u_2]^T$.

The system domain, initial set, unsafe set and target set are as follows:

$$\Psi = \{\mathbf{x} \in \mathbb{R}^6 \mid [-1, -1, -1, -1]^T \leq \mathbf{x} \leq [1, 1, 1, 1]^T\}$$

$$X_0 = \{\mathbf{x} \in \mathbb{R}^6 \mid 0 \leq \|\mathbf{x}\|_2 \leq 0.3\}$$

$$X_u = \{\mathbf{x} \in \mathbb{R}^6 \mid 0.9 \leq \|\mathbf{x}\|_2 \leq 1\}$$

$$X_g = \{\mathbf{x} \in \mathbb{R}^6 \mid 0 \leq \|\mathbf{x}\|_2 \leq 0.1\}.$$

We train a DNN of 5 hidden layers with 128 nodes in each layer through reinforcement learning. The polynomial $p(\mathbf{x})$ after approximation are as follows:

$$\begin{aligned} p_1(\mathbf{x}) = & 0.81914x_1 + 0.82545x_2 - 1.98596x_3 + 2.44036x_4 - 0.21937x_5 - 1.84627x_6 \\ & + 0.01162x_1^2 + 0.09664x_1x_2 - 0.03796x_1x_3 + 0.05884x_1x_4 + 0.0289x_1x_5 \\ & - 0.06582x_1x_6 - 0.04115x_2^2 - 0.18628x_2x_3 - 0.02347x_2x_4 - 0.29327x_2x_5 \\ & - 0.04066x_2x_6 + 0.03341x_3^2 - 0.09389x_3x_4 - 0.09845x_3x_5 + 0.06914x_3x_6 \\ & - 0.05852x_4^2 - 0.12168x_4x_5 + 0.09176x_4x_6 - 0.03962x_5^2 - 0.04278x_5x_6 \\ & + 0.02667x_6^2 \end{aligned}$$

$$\begin{aligned} p_2(\mathbf{x}) = & 0.91598x_1 - 1.09666x_2 - 1.82359x_3 - 0.09224x_4 - 5.19114x_5 + 0.31039x_6 \\ & - 0.02493x_1^2 - 0.02696x_1x_2 + 0.09007x_1x_3 - 0.00106x_1x_4 + 0.08078x_1x_5 \\ & + 0.08805x_1x_6 + 0.01969x_2^2 + 0.02197x_2x_3 + 0.08885x_2x_4 + 0.09529x_2x_5 \\ & - 0.03581x_2x_6 - 0.06899x_3^2 - 0.02862x_3x_4 - 0.14822x_3x_5 - 0.15205x_3x_6 \\ & + 0.17876x_4^2 - 0.00101x_4x_5 - 0.02506x_4x_6 - 0.11621x_5^2 - 0.16913x_5x_6 \\ & - 0.03915x_6^2 \end{aligned}$$

The small NN $k(\mathbf{x})$ is a single hidden layer neural network with 50 nodes and relu activation function. We give the parameters of this network in the form of TensorFlow code in the following link: <https://juzi1.github.io/RL/SNN/C4.py>

Under the constructed hybrid controller $p(\mathbf{x}) + k(\mathbf{x})$, the closed-loop system can be verified to satisfy the safety and goal-reaching properties by the following computed barrier certificate $B(\mathbf{x})$ and Lyapunov-like $V(\mathbf{x})$ function respectively.

$$\begin{aligned} B(\mathbf{x}) = & \underbrace{-0.4769x_1^6 - 1.7986x_1^5x_2 + 0.172x_1^5x_3 + \cdots - 0.2371x_6 + 0.2337}_{924 \text{ terms}}, \\ V(\mathbf{x}) = & \underbrace{2.6516x_1^4 + 4.5457x_1^3x_2 - 5.1226x_1^3x_3 + \cdots + 0.1621x_6 + 0.0003}_{210 \text{ terms}}. \end{aligned}$$

You can click the following link to view the complete parameters.

For $B(\mathbf{x})$: <https://juzi1.github.io/RL/Barriers/C4.txt>

For $V(\mathbf{x})$: <https://juzi1.github.io/RL/Lyapunov/C4.txt>

2.5 Academic 3D [3]

The equation of motion for Academic 3D is

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} z + 8y \\ -y + z \\ -z - x^2 + u \end{bmatrix}.$$

with the state is defined as $\mathbf{x} = [x, y, z]$.

The system domain, initial set, unsafe set and target set are as follows:

$$\Psi = \{\mathbf{x} = (x, y, z)^T \in \mathbb{R}^3 \mid 5 \leq x, y, z \leq 5\}$$

$$X_0 = \{\mathbf{x} \in \mathbb{R}^3 \mid \|\mathbf{x} - \mathbf{c}_0\|_2 \leq 0.35\},$$

$$X_u = \{\mathbf{x} \in \mathbb{R}^3 \mid \|\mathbf{x} - \mathbf{c}_u\|_2 \leq 0.30\},$$

$$X_g = \{\mathbf{x} \in \mathbb{R}^3 \mid \|\mathbf{x} - \mathbf{c}_g\|_2 \leq 0.1\}.$$

The vectors $\mathbf{c}_0 = [-0.75, -1, -0.4]^T$, $\mathbf{c}_u = [-0.3, -0.36, 0.2]^T$, $\mathbf{c}_g = [0, 0, 0]^T$.

We train a DNN of 5 hidden layers with 128 nodes in each layer through reinforcement learning. The polynomial $p(\mathbf{x})$ after approximation are as follows:

$$\begin{aligned} p(\mathbf{x}) = & 0.1247 - 3.3332x - 5.726y - 10.6688z + 1.9106x^2 + 1.2121xy \\ & + 2.1376xz - 1.3317y^2 - 10.0699yz - 12.9515z^2 \end{aligned}$$

The small NN $k(\mathbf{x})$ is a single hidden layer neural network with 30 nodes and relu activation function. We give the parameters of this network in the form of TensorFlow code in the following link: <https://juzi1.github.io/RL/SNN/C5.py>.

Under the constructed hybrid controller $p(\mathbf{x}) + k(\mathbf{x})$, the closed-loop system can be verified to satisfy the safety and goal-reaching properties by the following computed barrier certificate $B(\mathbf{x})$ and Lyapunov-like $V(\mathbf{x})$ function respectively.

$$B(\mathbf{x}) = \underbrace{0.1028x^4 - 0.3968x^3y + 0.1576x^3z + 0.1135x^3 + \dots - 0.0762z + 0.0095}_{35 \text{ terms}},$$

$$\begin{aligned} V(\mathbf{x}) = & -0.2351x^2 - 0.2706xy + 0.0923xz + 0.0026x + 0.0221y^2 - 0.1154yz \\ & + 0.0039y - 0.3102z^2 + 0.0033z + 0.0002. \end{aligned}$$

You can click the following link to view the complete parameters.

For $B(\mathbf{x})$: <https://juzi1.github.io/RL/Barriers/C5.txt>.

2.6 Dubin's Car [6]

The equation of motion for Dubin's Car system is

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} \sin(x_2) \\ -u \end{bmatrix}$$

with the state is defined as $\mathbf{x} = [x_1, x_2]$.

The system domain, initial set, unsafe set and target set are as follows:

$$\Psi = \{\mathbf{x} \in \mathbb{R}^2 \mid \{-6 \leq x_1 \leq 6, -7\pi/10 \leq x_2 \leq 7\pi/10\}\}$$

$$X_0 = \{\mathbf{x} \in \mathbb{R}^2 \mid \{-1 \leq x_1 \leq 1, -\pi/16 \leq x_2 \leq \pi/16\}\}$$

$$X_u = \{\mathbf{x} \in \mathbb{R}^2 \mid \{-5 \leq x_1 \leq 5, -\pi/2 \leq x_2 \leq \pi/2\}\}$$

$$X_g = \{\mathbf{x} \in \mathbb{R}^2 \mid 0 \leq \|\mathbf{x}\|_2 \leq 0.1\}.$$

We train a DNN of 4 hidden layers with 64 nodes in each layer through reinforcement learning. The polynomial $p(\mathbf{x})$ after approximation are as follows:

$$p(\mathbf{x}) = 0.057 + 4.8268x_1 + 7.1136x_2 - 0.3278x_1^2 - 2.6499x_1x_2 - 3.1174x_2^2$$

The small NN $k(\mathbf{x})$ is a single hidden layer neural network with 20 nodes and relu activation function. We give the parameters of this network in the form of TensorFlow code in the following link: <https://juzi1.github.io/RL/SNN/C6.py>

Under the constructed hybrid controller $p(\mathbf{x}) + k(\mathbf{x})$, the closed-loop system can be verified to satisfy the safety and goal-reaching properties by the following computed barrier certificate $B(\mathbf{x})$ and Lyapunov-like $V(\mathbf{x})$ function respectively.

$$B(\mathbf{x}) = -0.0993x_1^2 - 0.071x_1x_2 - 0.0118x_1 - 0.1016x_2^2 + 0.0603x_2 + 0.1174,$$

$$V(\mathbf{x}) = 0.0254x_1^2 + 0.0096x_1x_2 + 0.0002x_1 + 0.0444x_2^2 - 0.0001x_2 - 0.0001.$$

2.7 Space rendezvous [1]

The equation of motion for Space rendezvous system is

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{v}_x \\ \dot{v}_y \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \\ n^2x + 2nv_y + \frac{\mu}{r^2} - \frac{\mu}{r^3}(r+x) + \frac{v_x}{m_c} \\ n^2y - 2nv_x - \frac{\mu}{r^3}y + \frac{v_y}{m_c} \end{bmatrix}$$

with the constants set as

$$\mu = 3.986 \times 10^{14} \times 60^2 [m^3/min^2],$$

$$r = 42164 \times 10^3 [m],$$

$$m_c = 500 [kg], n = \sqrt{\frac{\mu}{r^3}}$$

,and the state is defined as $\mathbf{x} = [x, y, v_x, v_y]$.

The system domain,initial set,unsafe set and target set are as follows:

$$\Psi = \{\mathbf{x} \in \mathbb{R}^4 \mid [-200, -200, -200, -200]^T \leq \mathbf{x} \leq [200, 200, 200, 200]^T\}$$

$$X_0 = \{\mathbf{x} \in \mathbb{R}^4 \mid [-50, -50, 0, 0]^T \leq \mathbf{x} \leq [25, 25, 0, 0]^T\}$$

$$X_u = \{\mathbf{x} \in \mathbb{R}^4 \mid [-100, -100, -200, -200]^T \leq \mathbf{x} \leq [100, 100, 200, 200]^T\}$$

$$X_g = \{\mathbf{x} \in \mathbb{R}^4 \mid [-10, -10, -10, -10]^T \leq \mathbf{x} \leq [10, 10, 10, 10]^T\}.$$

We train a DNN of 5 hidden layers with 128 nodes in each layer through reinforcement learning. The polynomial $p(\mathbf{x})$ after approximation are as follows:

$$\begin{aligned} p_1(\mathbf{x}) = & -28.8306x_1 + 0.1045x_2 - 1450.0508x_3 + 0.1559x_4 - 0.0x_1^2 + 0.0x_1x_2 \\ & - 0.0002x_1x_3 - 9e-05x_1x_4 + 0.0x_2^2 + 0.001x_2x_3 - 0.0006x_2x_4 - 0.0097x_3^2 \\ & + 0.0367x_3x_4 - 0.0343x_4^2 \end{aligned}$$

$$\begin{aligned} p_2(\mathbf{x}) = & -0.0873x_1 - 33.2556x_2 + 0.0086x_3 - 1451.4999x_4 - 0.0x_1^2 + 0.0x_1x_2 \\ & - 0.0002x_1x_3 - 7e-05x_1x_4 - 0.0x_2^2 + 0.001x_2x_3 - 0.0009x_2x_4 - 0.0104x_3^2 \\ & + 0.0404x_3x_4 - 0.0388x_4^2 \end{aligned}$$

The small NN $k(\mathbf{x})$ is a single hidden layer neural network with 50 nodes and relu activation function. We give the parameters of this network in the form of TensorFlow code in the following link: <https://juzi1.github.io/RL/SNN/C7.py>

Under the constructed hybrid controller $p(\mathbf{x}) + k(\mathbf{x})$, the closed-loop system can be verified to satisfy the safety and goal-reaching properties by the following computed barrier certificate $B(\mathbf{x})$ and Lyapunov-like $V(\mathbf{x})$ function respectively.

$$B(\mathbf{x}) = \underbrace{-0.4765x_1^4 + 0.6794x_1^3x_2 - 1.0763x_1^3x_3 + \cdots - 0.0737x_4 + 370.7638}_{70 \text{ terms}},$$

$$V(\mathbf{x}) = \underbrace{1.9678x_1^4 - 1.0938x_1^3x_2 - 4.7666x_1^3x_3 + \cdots - 1.9973x_4 - 48.4569}_{70 \text{ terms}}$$

You can click the following link to view the complete parameters.

For $B(\mathbf{x})$: <https://juzi1.github.io/RL/Barriers/C7.txt>

For $V(\mathbf{x})$: <https://juzi1.github.io/RL/Lyapunov/C7.txt>

2.8 Oscillator [7]

Van der Pol's oscillator is a 2-dimensional non-linear benchmark system, which can be expressed as

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ \gamma(1 - x_1^2)x_2 - x_1 + u \end{bmatrix}.$$

with the constants set as $\gamma = 1$, and the state is defined as $\mathbf{x} = [x_1, x_2]$.

The system domain, initial set, unsafe set and target set are as follows:

$$\Psi = \{\mathbf{x} \in \mathbb{R}^2 \mid [-2, -2]^T \leq \mathbf{x} \leq [2, 2]^T\}$$

$$X_0 = \{\mathbf{x} \in \mathbb{R}^2 \mid [-0.51, 0.51]^T \leq \mathbf{x} \leq [-0.49, 0.49]^T\}$$

$$X_u = \{\mathbf{x} \in \mathbb{R}^2 \mid [-0.4, 0.2]^T \leq \mathbf{x} \leq [0.1, 0.35]^T\}$$

$$X_g = \{\mathbf{x} \in \mathbb{R}^2 \mid [-0.05, 0.05]^T \leq \mathbf{x} \leq [-0.05, 0.05]^T\}.$$

We train a DNN of 5 hidden layers with 64 nodes in each layer through reinforcement learning. The polynomial $p(\mathbf{x})$ after approximation are as follows:

$$p(\mathbf{x}) = -0.0133 + 1.2057x_1 + 1.2299x_2 + 13.5323x_1^2 + 81.1611x_2x_1 + 57.9627x_2^2$$

The small NN $k(\mathbf{x})$ is a single hidden layer neural network with 20 nodes and relu activation function. We give the parameters of this network in the form of TensorFlow code in the following link: <https://juzi1.github.io/RL/SNN/C8.py>.

Under the constructed hybrid controller $p(\mathbf{x}) + k(\mathbf{x})$, the closed-loop system can be verified to satisfy the safety and goal-reaching properties by the following computed barrier certificate $B(\mathbf{x})$ and Lyapunov-like $V(\mathbf{x})$ function respectively.

$$B(\mathbf{x}) = 0.1837x_1^2 + 0.1128x_1x_2 + 0.083x_1 + 0.1181x_2^2 - 0.0197x_2 + 0.0115,$$

$$V(\mathbf{x}) = 0.0352x_1^2 - 0.027x_1x_2 - 0.0001x_1 + 0.0448x_2^2 - 0.0001x_2 - 0.0001.$$

2.9 Bicycle Steering [3]

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} x_2 \\ \frac{ml}{J}(g \cdot \sin(x_1)) + \frac{v^2}{b} \cos(x_1) \tan(x_3) \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{amlv}{Jb} \cdot \frac{\cos(x_1)}{\cos^2(x_3)} \\ 1 \end{bmatrix} u$$

where u is the control input. By introducing a new neural network controller \tilde{u} such that $u = \tilde{u} \cos^2(x_3) - 20 \cos(x_3) \sin(x_3)$, the original system is transformed

into the following one:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} x_2 \\ 30 \sin(x_1) + 15\tilde{u} \cos(x_1) \\ -20 \cos(x_3) \sin(x_3) + \tilde{u} \cos^2(x_3) \end{bmatrix}$$

with the constants set as

$$m = 20, l = 1, b = 1, J = \frac{mb^2}{3}, v = 10, g = 10, a = 0.5.$$

Define the state of space

$$\mathbf{x} = [x_1, x_2, x_3],$$

. The system domain, initial set, unsafe set and target set are as follows:

$$\Psi = \{\mathbf{x} \in \mathbb{R}^3 \mid -2.2 \leq x_1 \leq 2.2, -2.2 \leq x_2 \leq 2.2, -2.2 \leq x_3 \leq 2.2\}$$

$$X_0 = \{\mathbf{x} \in \mathbb{R}^3 \mid -0.2 \leq x_1 \leq 0.2, -0.2 \leq x_2 \leq 0.2, -0.2 \leq x_3 \leq 0.2\}$$

$$X_u = \Psi - \{\mathbf{x} \in \mathbb{R}^3 \mid -2 \leq x_1 \leq 2, -2 \leq x_2 \leq 2, -2 \leq x_3 \leq 2\}$$

$$X_g = \{\mathbf{x} \in \mathbb{R}^3 \mid -0.1 \leq x_1 \leq 0.1, -0.1 \leq x_2 \leq 0.1, -0.1 \leq x_3 \leq 0.1\}.$$

We train a DNN of 5 hidden layers with 128 nodes in each layer through reinforcement learning. The polynomial $p(\mathbf{x})$ after approximation are as follows:

$$\begin{aligned} p(\mathbf{x}) = & -7.15341x_1 - 4.73806x_2 - 15.26188x_3 + 1.4062x_1^2 - 6.96049x_1x_2 \\ & + 0.41919x_1x_3 - 4.14752x_2^2 + 6.70594x_2x_3 - 6.14518x_3^2 \end{aligned}$$

The small NN $k(\mathbf{x})$ is a single hidden layer neural network with 30 nodes and relu activation function. We give the parameters of this network in the form of TensorFlow code in the following link: <https://juzi1.github.io/RL/SNN/C9.py>

Under the constructed hybrid controller $p(\mathbf{x}) + k(\mathbf{x})$, the closed-loop system can be verified to satisfy the safety and goal-reaching properties by the following computed barrier certificate $B(\mathbf{x})$ and Lyapunov-like $V(\mathbf{x})$ function respectively.

$$B(\mathbf{x}) = 8.1107x_1^2 - 9.9155x_1x_2 - 2.4467x_1x_3 + 3.066x_1 + 0.0911x_2^2 + 7.0226x_2x_3 + 8.5096x_2 + 7.0552x_3^2 + 13.$$

$$V(\mathbf{x}) =$$

You can click the following link to view the complete parameters.

For $V(\mathbf{x})$: <https://juzi1.github.io/RL/Lyapunov/C9.txt>

2.10 LALO20 [5]

The Laub-Loomis model is for studying a class of enzymatic activities. The dynamics can be defined by the following ODE with 7 variables.

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \\ \dot{x}_6 \\ \dot{x}_7 \end{bmatrix} = \begin{bmatrix} 1.4x_3 - 0.9x_1 \\ 2.5x_5 - 1.5x_2 + u \\ 0.6x_7 - 0.8x_2x_3 \\ 2 - 1.3x_3x_4 \\ 0.7x_1 - x_4x_5 \\ 0.3x_1 - 3.1x_6 \\ 1.8x_6 - 1.5x_2x_7 \end{bmatrix}$$

with the state is defined as $\mathbf{x} = [x_1, x_2, x_3, x_4, x_5, x_6, x_7]$.

The system domain, initial set, unsafe set and target set are as follows:

$$\Psi = \{\mathbf{x} \in \mathbb{R}^7 \mid [-3.8, -3.95, -3.5, -2.6, -4., -4.9, -4.55]^T \leq \mathbf{x} \leq [6.2, 6.05, 6.5, 7.4, 6., 5.1, 5.45]^T\}$$

$$X_0 = \{\mathbf{x} \in \mathbb{R}^7 \mid [1.15, 1., 1.45, 2.35, 0.95, 0.05, 0.4]^T \leq \mathbf{x} \leq [1.25, 1.1, 1.55, 2.45, 1.05, 0.15, 0.5]^T\}$$

$$X_u = \Psi - \{\mathbf{x} \in \mathbb{R}^7 \mid [-3.3, -3.45, -3., -2.1, -3.5, -4.4, -4.05]^T \leq \mathbf{x} \leq [1.25, 1.1, 1.55, 2.45, 1.05, 0.15, 0.5]^T\}$$

$$X_g = \{\mathbf{x} \in \mathbb{R}^7 \mid [0.77, 0.27, 0.46, 2.65, 0.12, -0.02, 0.17]^T \leq \mathbf{x} \leq [0.97, 0.47, 0.66, 2.85, 0.32, 0.18, 0.37]^T\}.$$

We train a DNN of 6 hidden layers with 128 nodes in each layer through reinforcement learning. The polynomial $p(\mathbf{x})$ after approximation are as follows:

$$\begin{aligned} p(\mathbf{x}) = & 0.62175x_1 + 0.72265x_2 - 0.08394x_3 + 0.32562x_4 - 1.12218x_5 + 0.07453x_6 \\ & - 0.41217x_7 - 0.02857x_1^2 - 0.48429x_1x_2 - 0.24386x_1x_3 - 0.08897x_1x_4 \\ & + 0.47079x_1x_5 - 0.37944x_1x_6 + 0.6159x_1x_7 + 0.07929x_2^2 + 0.318x_2x_3 \\ & - 0.15821x_2x_4 - 0.31944x_2x_5 + 0.58502x_2x_6 - 0.21606x_2x_7 + 0.09618x_3^2 \\ & + 0.02111x_3x_4 - 0.15265x_3x_5 + 0.10732x_3x_6 - 0.33442x_3x_7 - 0.03321x_4^2 \\ & + 0.24253x_4x_5 - 0.64637x_4x_6 - 0.01769x_4x_7 + 0.28253x_5^2 + 0.98225x_5x_6 \\ & - 0.11167x_5x_7 - 0.63756x_6^2 + 0.56567x_6x_7 + 0.42151x_7^2 \end{aligned}$$

The small NN $k(\mathbf{x})$ is a single hidden layer neural network with 50 nodes and relu activation function. We give the parameters of this network in the form of TensorFlow code in the following link: <https://juzi1.github.io/RL/SNN/C10.py>

Under the constructed hybrid controller $p(\mathbf{x}) + k(\mathbf{x})$, the closed-loop system can be verified to satisfy the safety and goal-reaching properties by the following computed barrier certificate $B(\mathbf{x})$ and Lyapunov-like $V(\mathbf{x})$ function respectively.

$$\begin{aligned}
B(\mathbf{x}) &= \underbrace{-2.768x_1^2 + 0.3756x_1x_2 - 9.0604x_1x_3 + \cdots + 4.7161x_7 - 4.7758}_{36 \text{ terms}}, \\
V(\mathbf{x}) &= \underbrace{-4.3416x_1^2 - 4.5311x_1x_2 - 1.7773x_1x_3 + \cdots - 0.749x_7 + 37.0249}_{36 \text{ terms}}.
\end{aligned}$$

You can click the following link to view the complete parameters.

For $B(\mathbf{x})$: <https://juzi1.github.io/RL/Barriers/C10.txt>

For $V(\mathbf{x})$: <https://juzi1.github.io/RL/Lyapunov/C10.txt>

References

- [1] Chan, N., Mitra, S.: Verifying safety of an autonomous spacecraft rendezvous mission. arXiv preprint arXiv:1703.06930 (2017)
- [2] Chang, Y.C., Roohi, N., Gao, S.: Neural lyapunov control. In: Neural Information Processing Systems (2019)
- [3] Deshmukh, J.V., Kapinski, J.P., Yamaguchi, T., Prokhorov, D.: Learning deep neural network controllers for dynamical systems with safety guarantees: Invited paper. In: 2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD) (2019)
- [4] Jin, W., Wang, Z., Yang, Z., Mou, S.: Neural certificates for safe control policies (2020)
- [5] Laub, M.T., Loomis, W.F.: A molecular network that produces spontaneous oscillations in excitable cells of dictyostelium. *Molecular biology of the cell* **9**(12), 3521–3532 (1998)
- [6] Zhao, H., Zeng, X., Chen, T., Liu, Z., Woodcock, J.: Learning safe neural network controllers with barrier certificates. *Formal Aspects of Computing* pp. 1–19 (2021)
- [7] Zhu, H., Xiong, Z., Magill, S., Jagannathan, S.: An inductive synthesis framework for verifiable reinforcement learning. In: Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation. pp. 686–701 (2019)