CONTROLLING A QIP SYSTEM USING Q-LEARNING AND LQR

A minimalistic Approach Using NumPy

ENTRY FOR THE 2ND KOH YOUNG AI COMPETITION

30 October, 2024

QUADRUPLE INVERTED PENDULUM SYSTEM

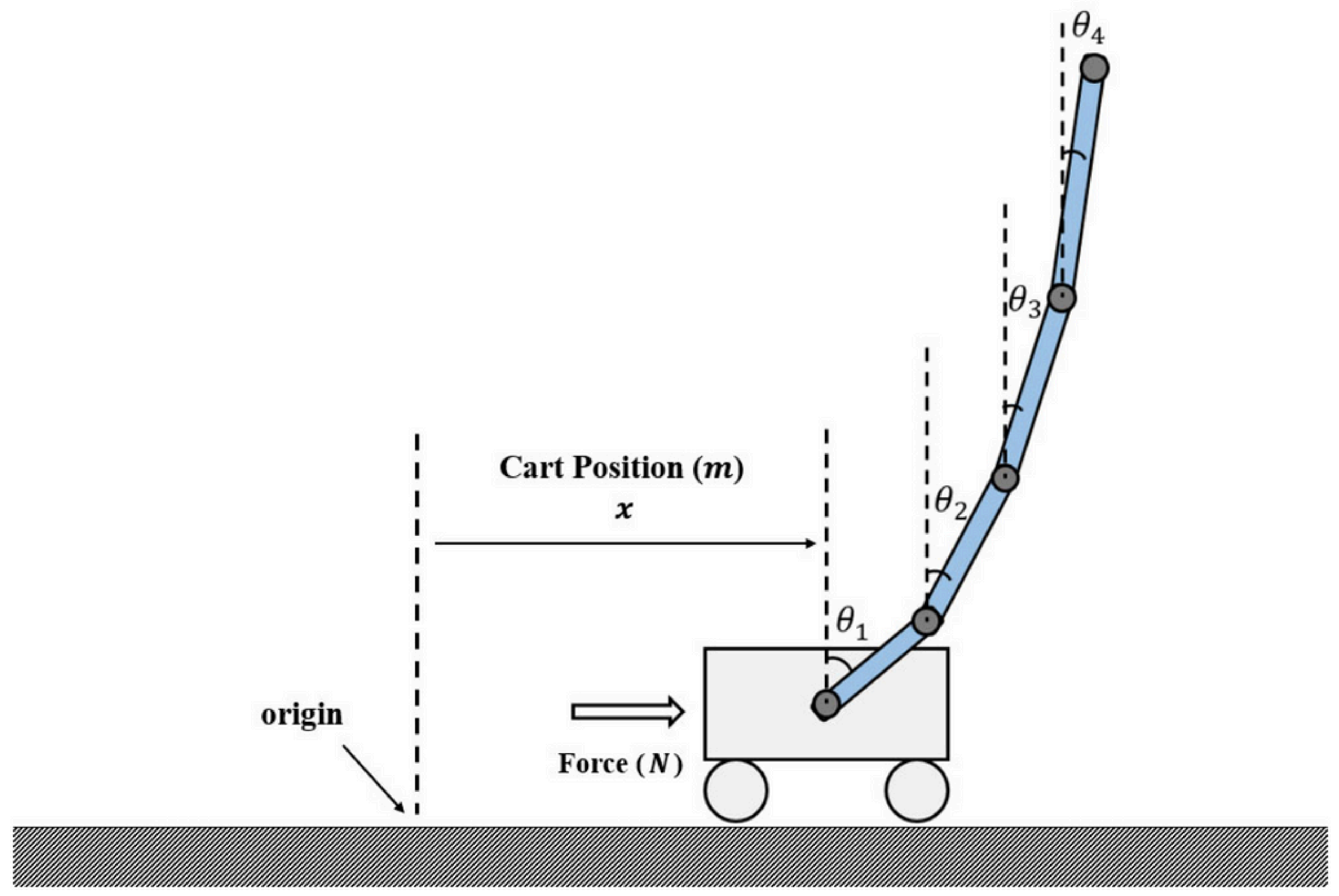


Figure 1. Diagram of the QIP system.

- A cart with four pendulums attached end to end
- Complex, Highly Nonlinear, and Unstable System
- Simulation and Visualization Provided by an existing system
- Basis for testing nonlinear control theory

CHALLENGE

COMPLEXITY

HIGH DEGREES OF FREEDOM
NONLINEARITY

INSTABILITY

EXPONENTIAL EFFECTS OF SMALL DISTURBANCES

CONSTRAINTS

NUMPY LIB

OUR SOLUTION

Q-LEARNING

REINFORCEMENT LEARNING HANDLES COMPLEX, NONLINEAR SYSTEMS

LQR

LINEAR QUADRATIC REGULATOR

COMPATIBILITY WITH CONSTRAINTS

SYSTEM DYNAMICS AND LINEARIZATION

- Derivation Using Lagrangian Mechanics
- Euler-Lagrange Equations

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}_i}\right) - \frac{\partial \mathcal{L}}{\partial q_i} = F_i$$

- NDSolve
- StateSpaceModel

Source: WOLFRAM Mathematica

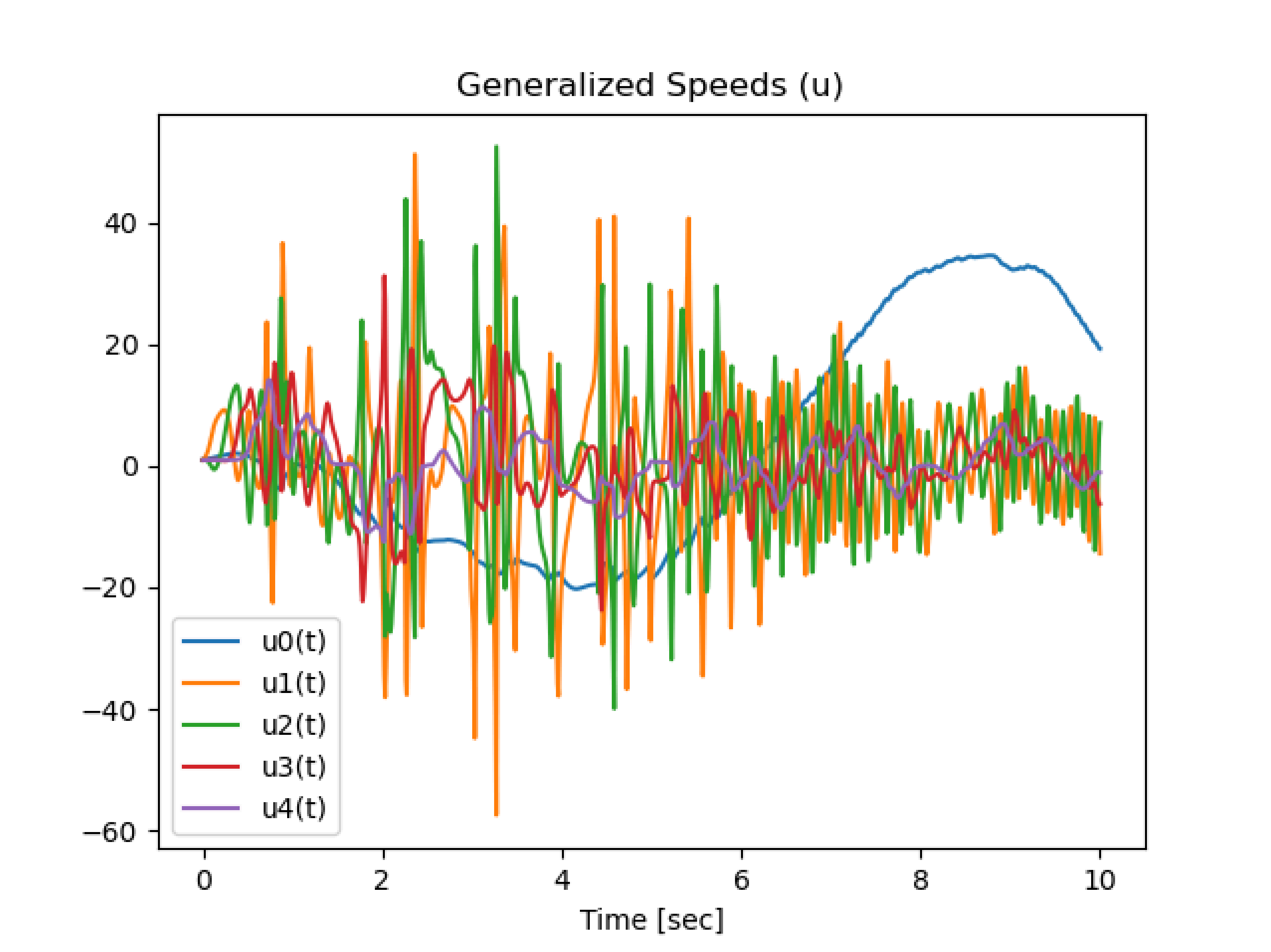
Physical parameters	Cart	Link 1	Link 2	Link 3	Link 4
Mass (kg)	1.3000	0.2000	0.2500	0.4300	0.8100
Length (m)	-	0.1500	0.2000	0.4000	0.8000
Distance from hinge to *CoG (m)	-	0.0750	0.1000	0.2000	0.4000
**MoI of rods about their CoG (kg•)	-	0.0015	0.0033	0.0229	0.1728

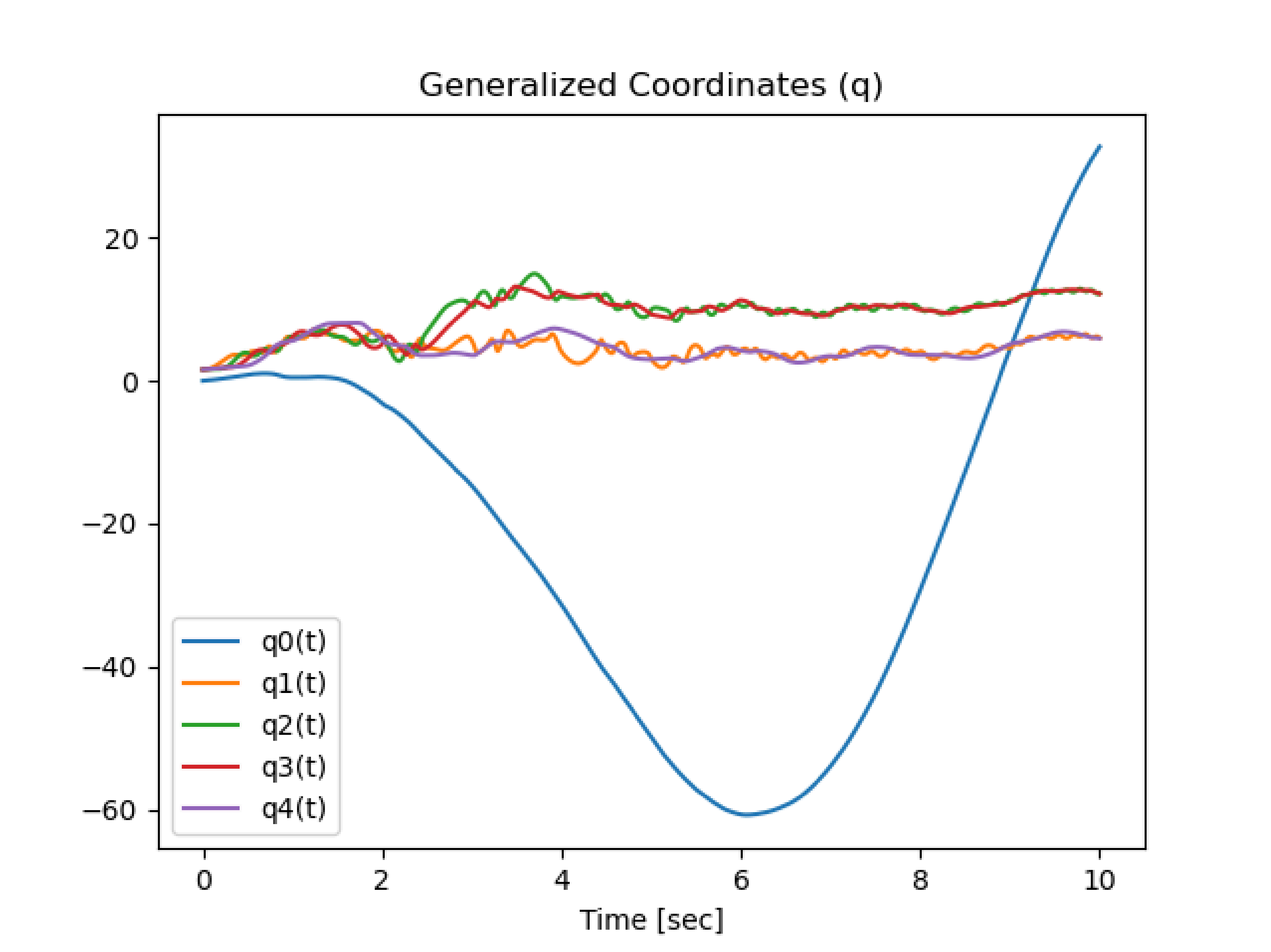
LQR

```
def solve_lqr(self, A, B, Q, R, num_iters=100):
    P = Q
    for _ in range(num_iters):
        P_next = Q + A.T @ P @ A - A.T @ P @ B @ np.linalg.inv(R + B.T @ P @ B) @ B.T @ P @ A
        if np.allclose(P, P_next):
            break
        P = P_next
    K = np.linalg.inv(R + B.T @ P @ B) @ B.T @ P @ A
    return K
```

LQR STABILIZATION

Control Law: u=−K[x−x(g)]





Q - LEARNING METHODOLOGY

System Setup

- State Discretization

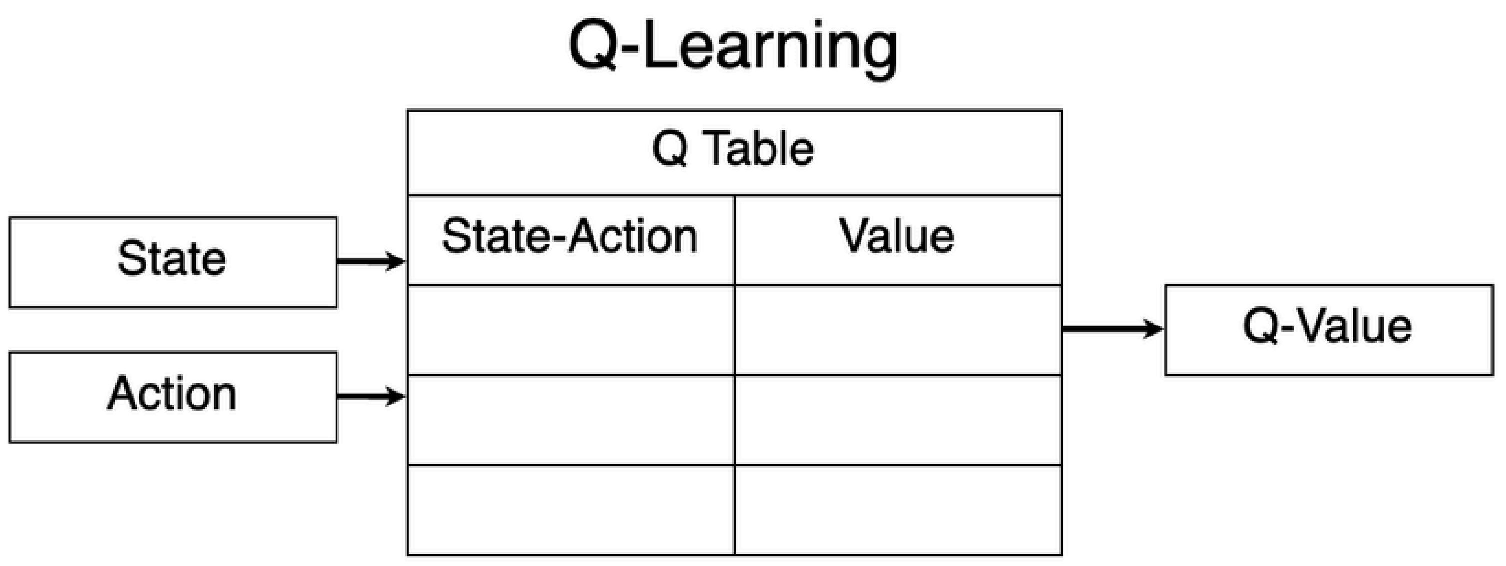
  State Bins for each variable [3, 3, 3, . . . 3]

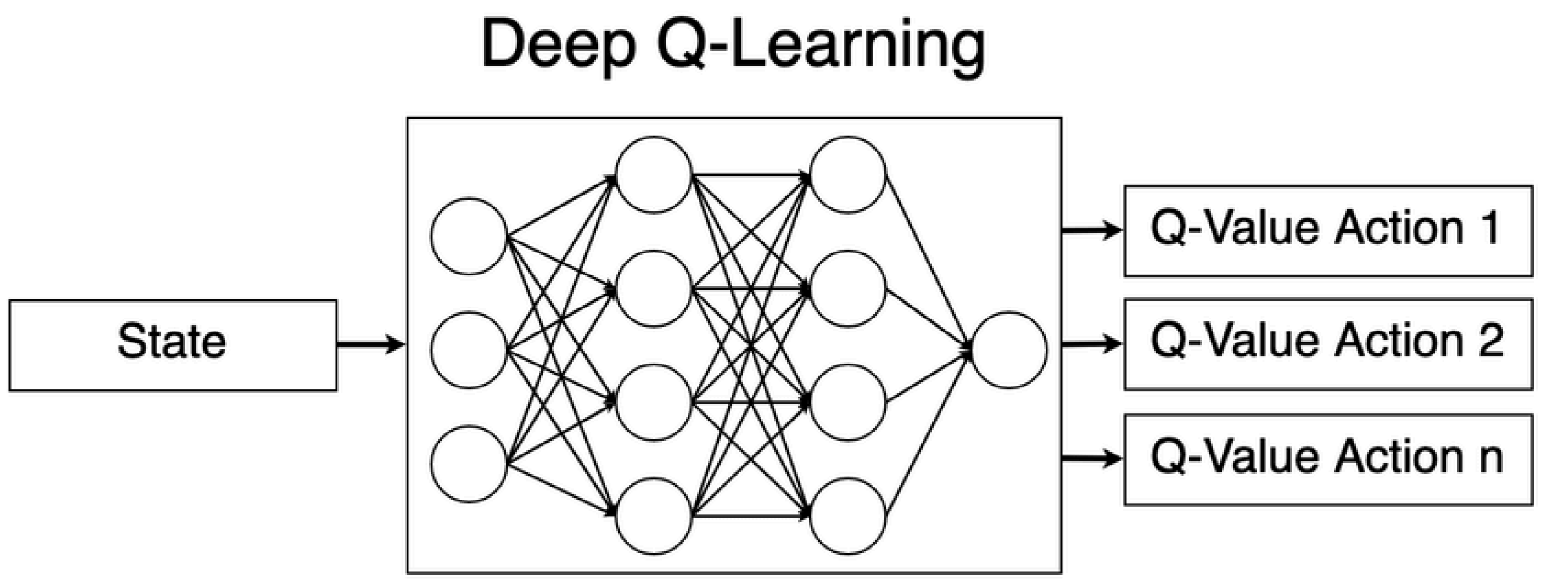
  Angle Wrapping $[-\pi,\pi]$
- Action Space

  Discretized Control Forces: 15 Actions
- Q Learning Parameters

  Epsilon (ϵ): Exploration, Exploitation rate
- Learning rate (α)

  $\alpha = 0.1$





Q - LEARNING METHODOLOGY

Training and Q table updating

- Action Selection

  Exploration (Select a random action)

  Exploitation (Select action with highest Q-value)
- Q-Table Update Rule

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right]$$

- Training Mode:

  Q-table updates are performed only when the controller is in training mode.
- Best Q Table

  Best Cumulative reward

  Success Rate

Q - LEARNING METHODOLOGY.

Reward Function Determination

- Cart Position Penalty

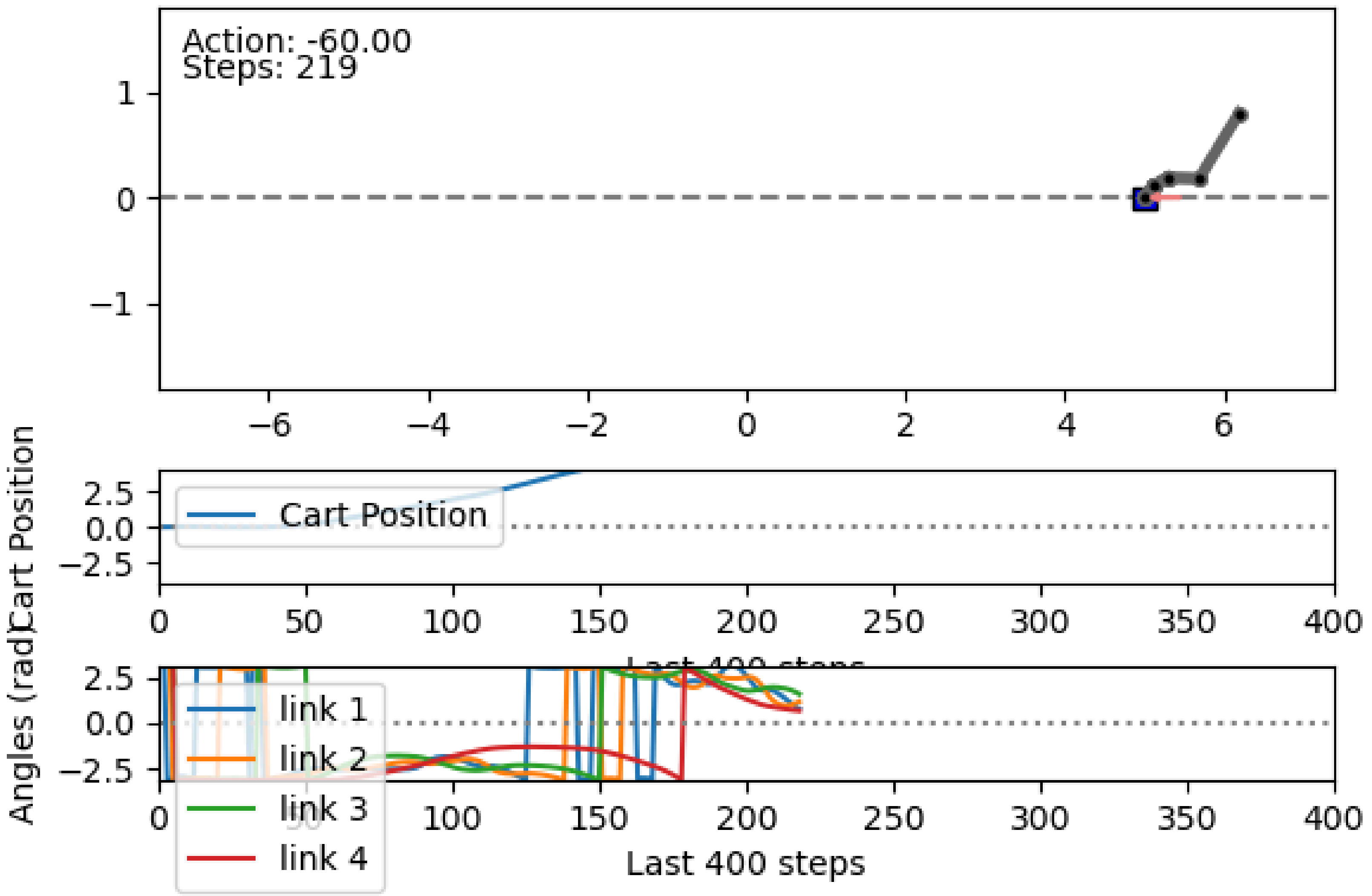
$$\text{Penalty} = -0.1 \times |\text{cart position}|$$

- Angular Velocity Penalty

$$\text{Penalty} = -0.1 \times \sum_{i=1}^{4} |\dot{\theta}_i|$$

- Pendulum Upright Reward

$$\text{Reward} = 1 \times \sum_{i=1}^{4} \left(\frac{1 + \cos(\theta_i)}{2} \right)$$



TRANSITION BETWEEN Q-LEARNING AND LQR

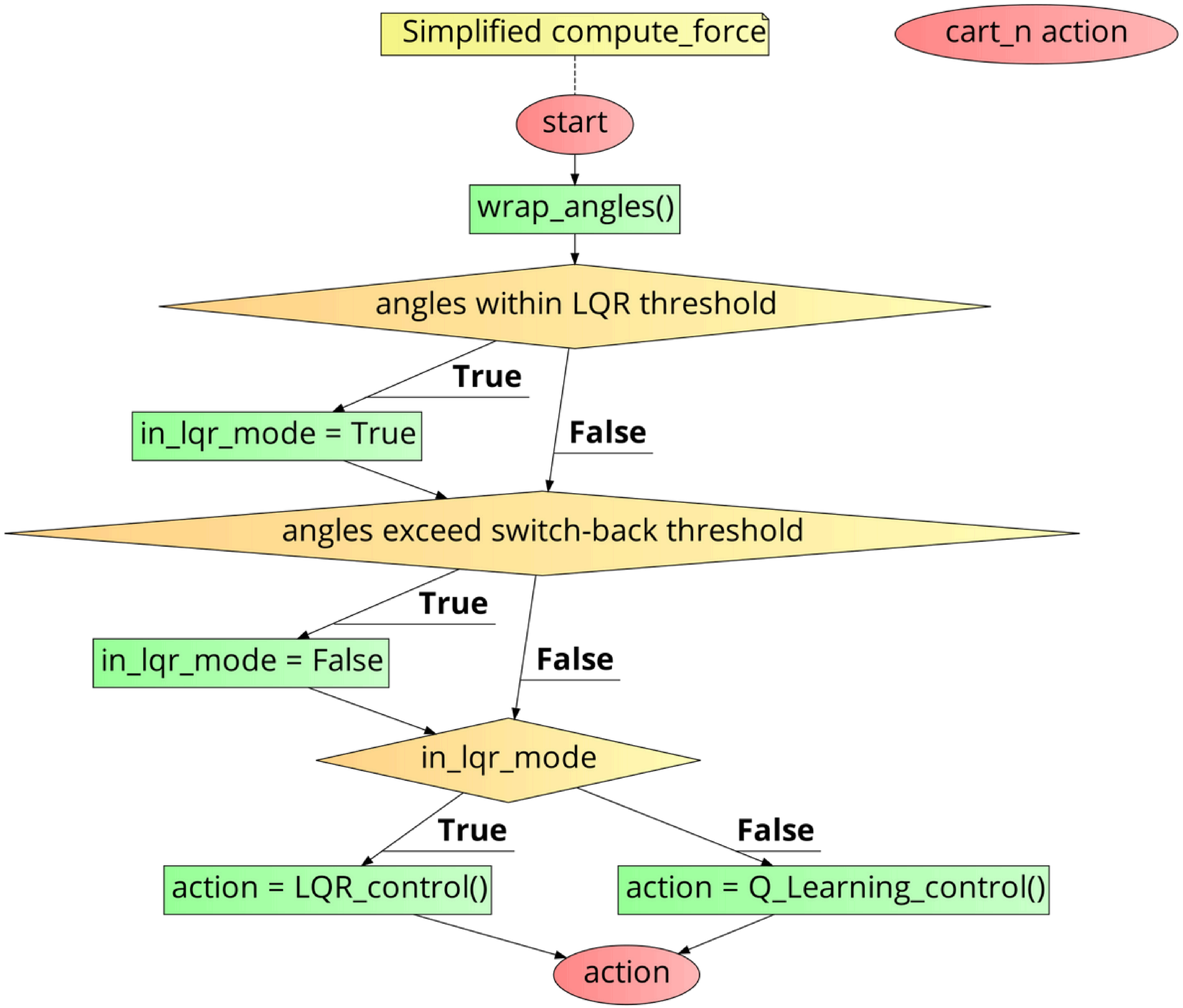
Control Strategy Integration

- Q Learning for Swing-up
- LQR Control for Robust upright stability

Threshold angle switching mechanism

- Activation of LQR
- Deactivation of LQR

Q-LEARNING LQR SYNERGY



DISCUSSIONS

Evaluation

- Swing action in under 10s
- Accurate LQR control near Equilibrium

Challenges

- high Dimensional State space leading to incredibly large Q matrix
- Balancing, exploration and exploitation

CONCLUSION

- Developed a model to swing up (Model Free Q-learning) and balance (Custom Linearization) an inverted pendulum
- Using Minimal tools (NumPy)
- Synergy between reinforcement learning and classical control

Future Work

- A separate Network for the switching mechanism
- Neural Network based learning instead of Q tables

Pluto_X Team
Awai Tersoo Samuel and Kim Jung-Gil

THANK YOU