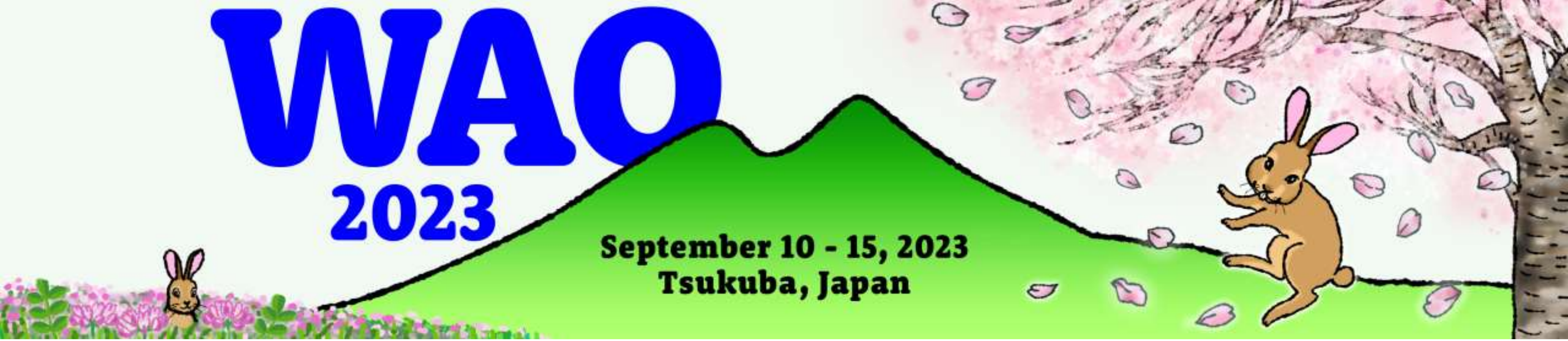


Online reinforcement learning control of beam collision at IP for BEPCII

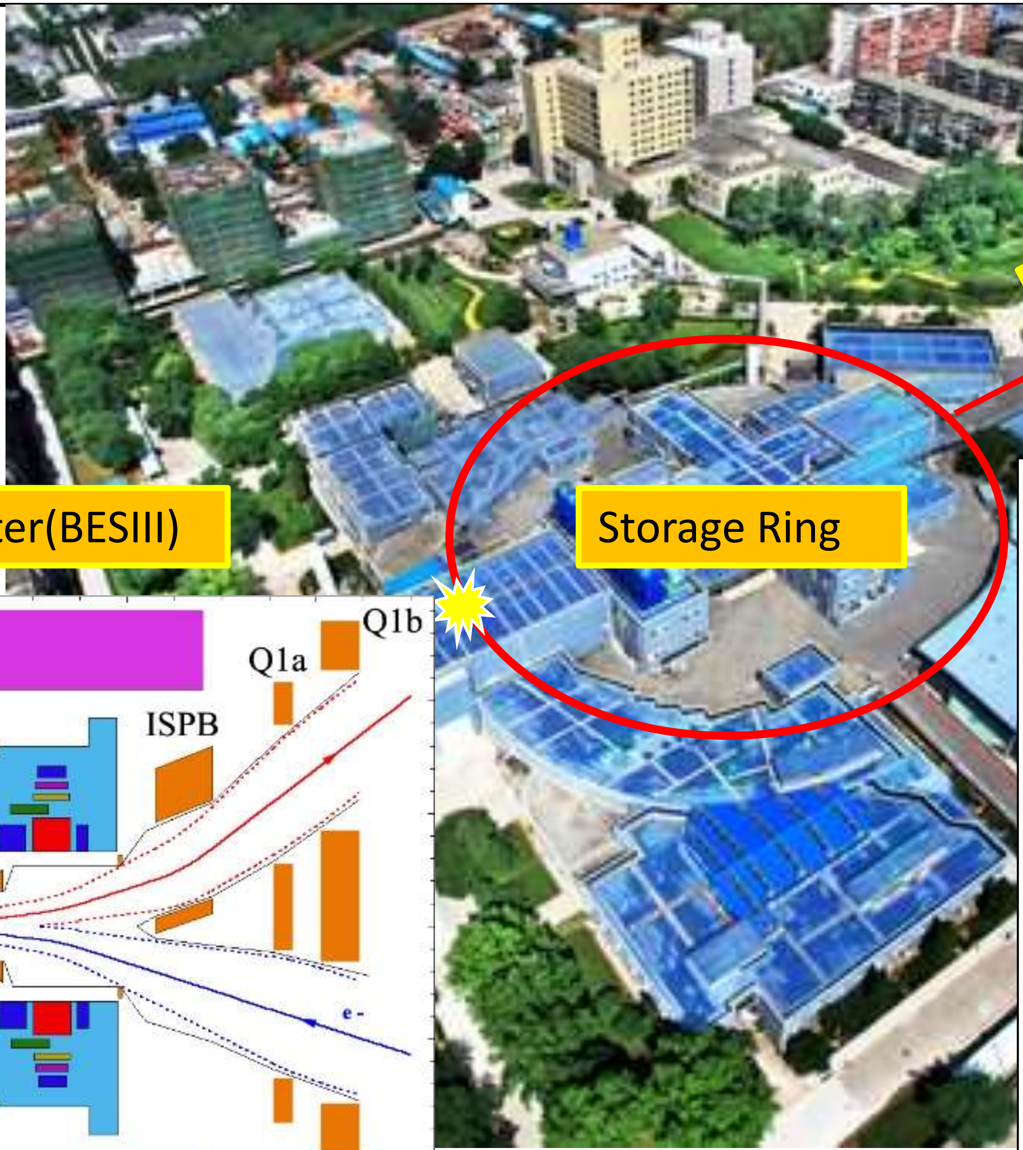
Jiaqi Fan

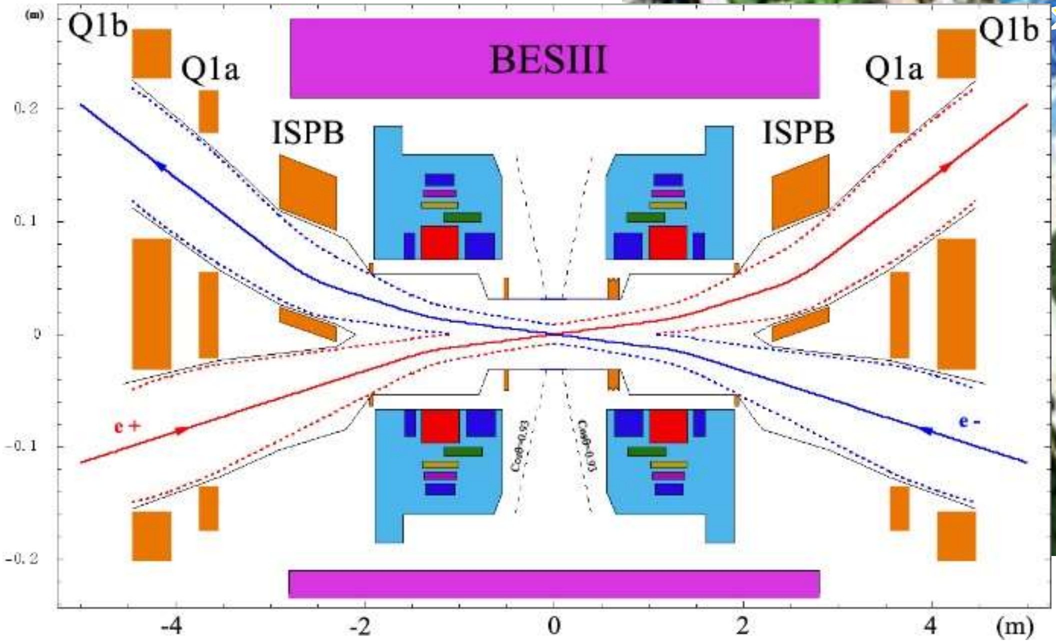
How does the Machine Learning integrate with Operation?
WAO2023,12/09/2023

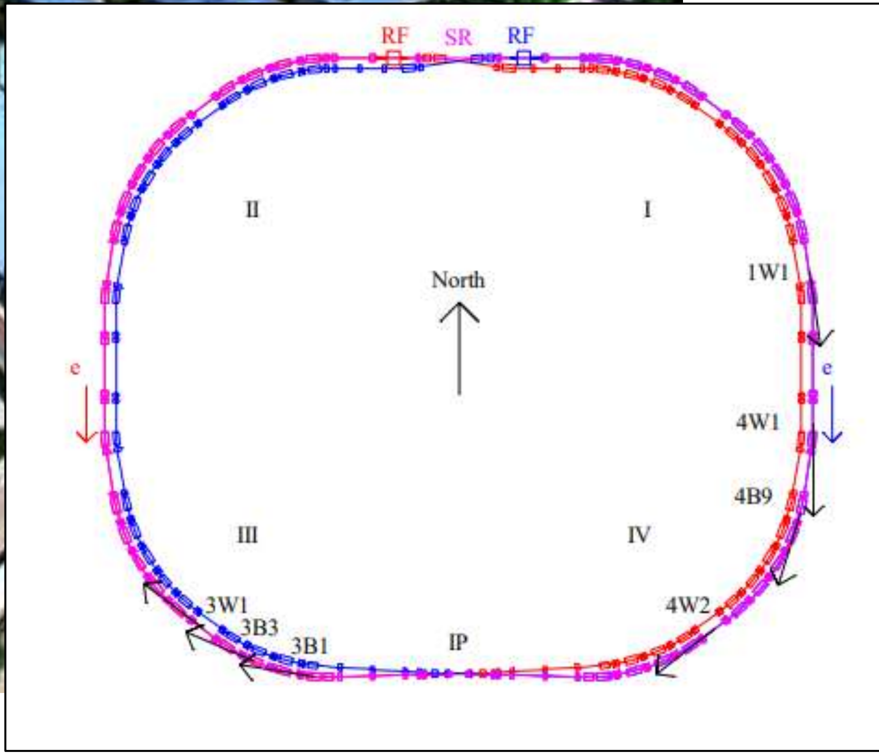
中国科学院高能物理研究所
Institute of High Energy Physics

Introduction

The upgrade project of Beijing Electron–Positron Collider (BEPCII)





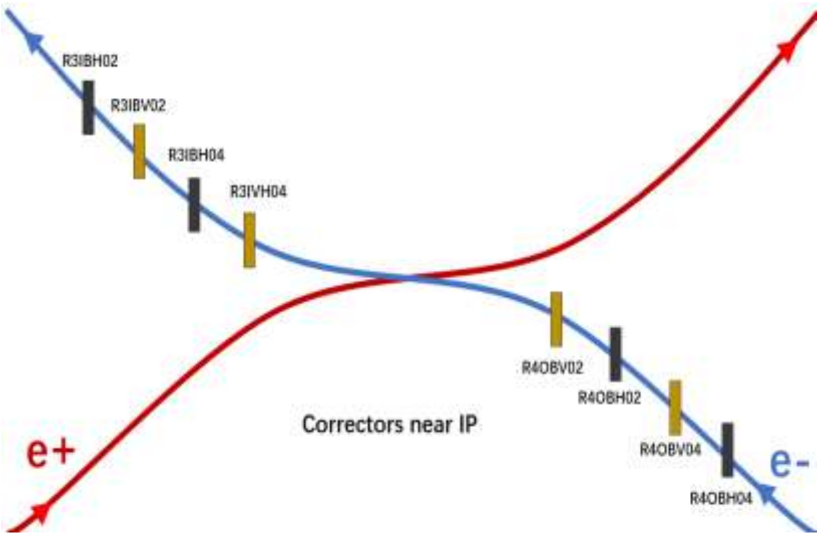


Introduction

Transverse offset in displacement and angular deviation (Offset)

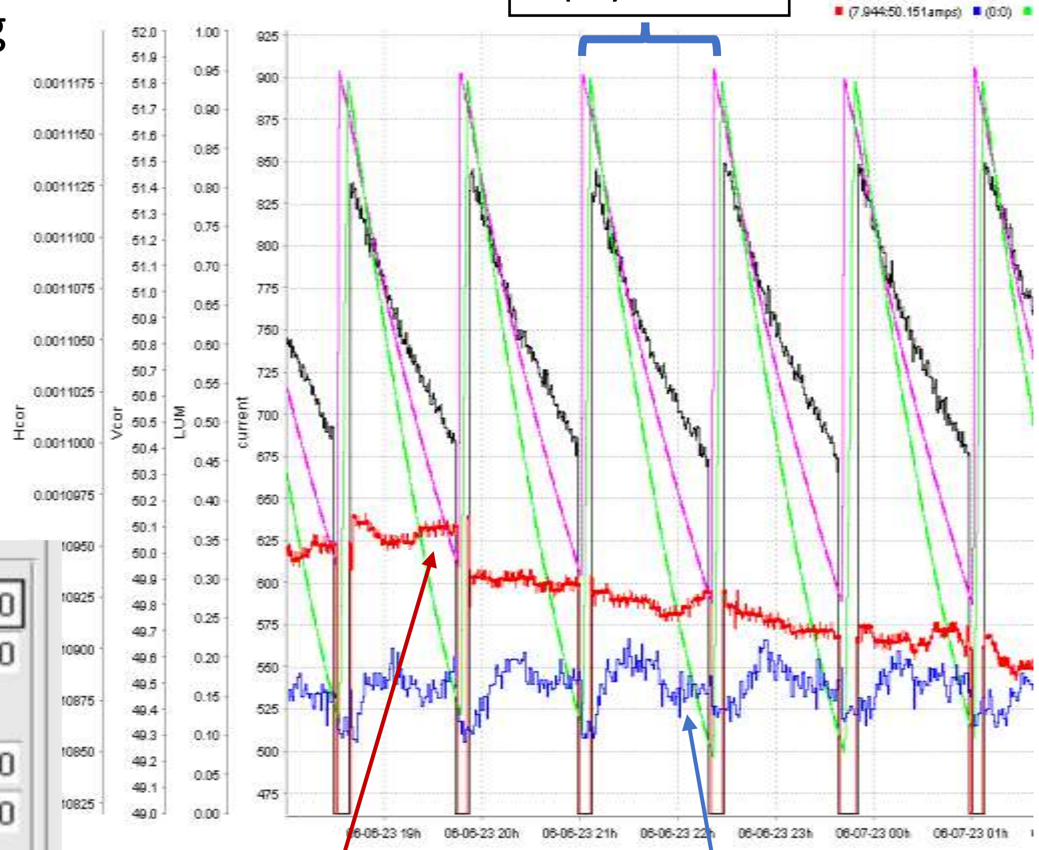
- Four knobs (x, x', y, y') make of eight correctors for each ring
- The most frequently used parameters
- Always only tune the knobs of electron ring
- Tune manually
- Depends on orbit and current, need continuous optimization

Manual operation!
Scan one by one!



Offset knobs of e^- ring

A physics run



Y plane offset

X plane Offset

Introduction

Control method:

Feedback Method: Control beam orbits around the IP directly

Our machine: small ring、34 correctors、no precise bpm around IP

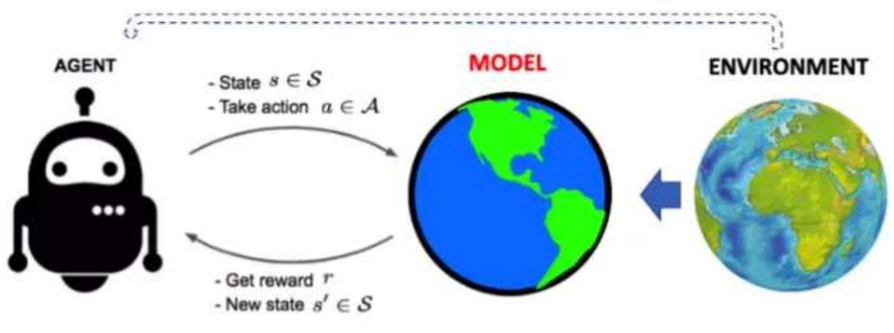
Optimization Method: Luminosity optimization (luminosity-driven system)

Optimization: can only find a temporary optimal result in a dynamic environment.

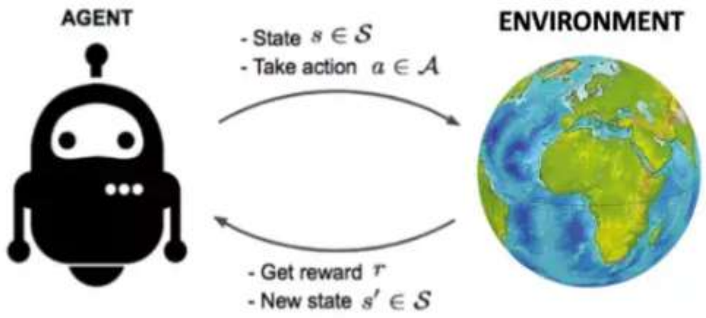
Machine learning: Data-driven、model-free

Reinforcement learning

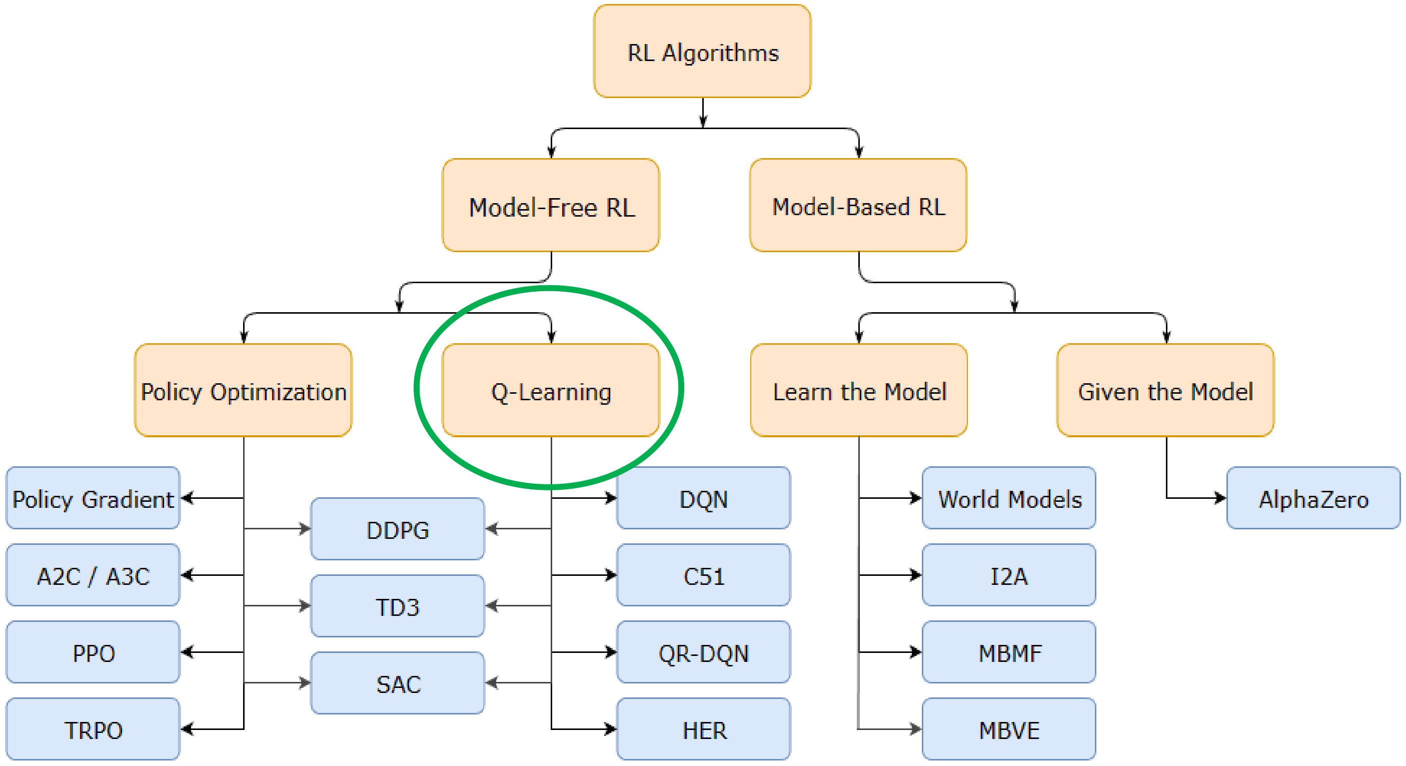
- **RL:** Train an **Agent** to make the decision of what action to take to get more reward from environment



Model-based RL



Model-free RL



Deep Q-Network(DQN)

For each step:

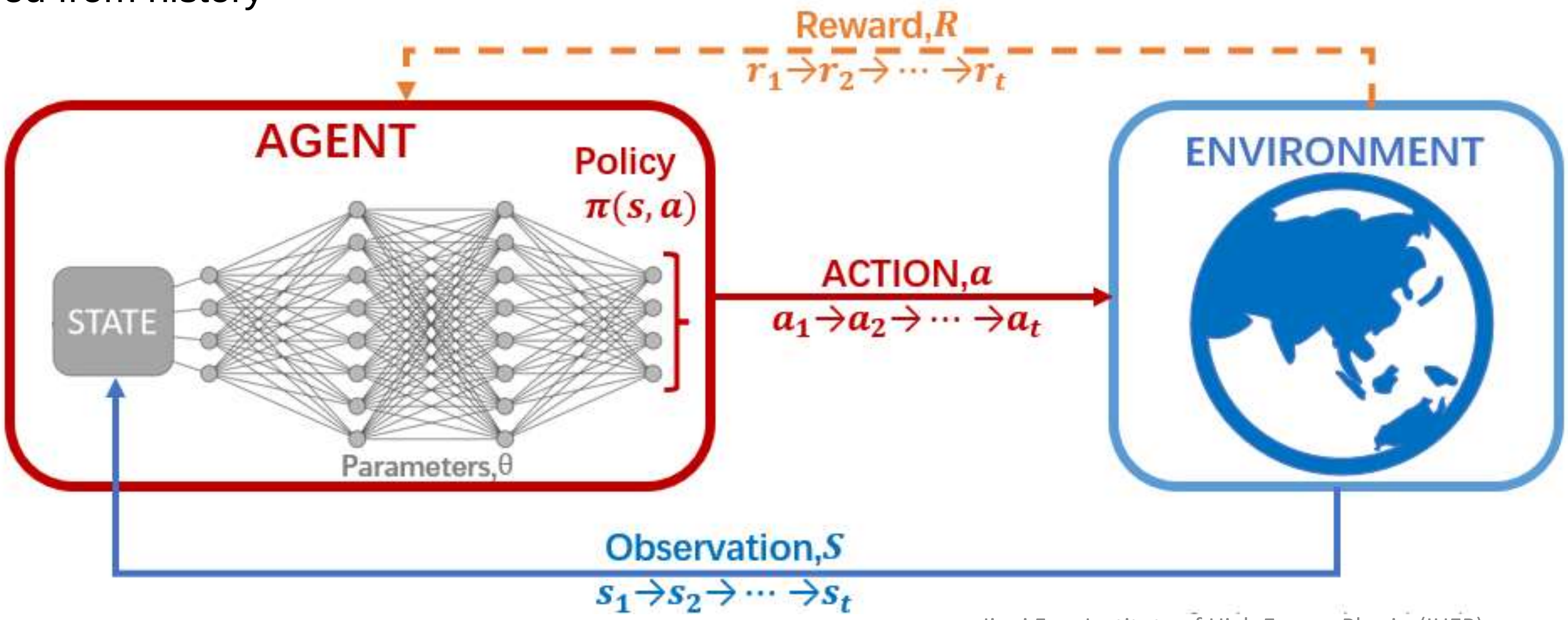
1. Agent receive observation S_t
2. Calculate Q for each action on S_t
3. Choose the action with greatest Q, or choose random actions with a small probability
4. executes the action and observe new state S_{t+1} and reward R_t, sort $[S_t, A_t, R_t, S_{t+1}]$ into history dataset
5. Update NN each N steps with minibatch of $[S_t, A_t, R_t, S_{t+1}]$ sampled from history

$\boldsymbol{Q(s,a)}$ **= QUALITY OF STATE/ACTION PAIR**

$$\boldsymbol{Q_t = R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \gamma^3 R_{t+3} + \cdots}$$
$$\boldsymbol{= R_t + \gamma Q_{t+1} \quad , \gamma = 0 \sim 1}$$

NN update:

$$\boldsymbol{\theta_{t+1} = \theta_t + a[R_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}; \theta_t) - Q(s_t, a_t; \theta_t)] \nabla Q(s_t, a_t; \theta_t)}$$



Deep Q-Network(DQN)

Algorithm 1: DQN

Initialize replay memory D to capacity N

Initialize action-value function Q with random weights θ

Initialize target action-value function $\hat{Q}$ with weights $\hat{\theta} = \theta$

For episode =1,M **do**

observe initial state s_0

For $t =, T$ **do**

With probability ϵ select a random action a_t

Otherwise select action $a_t = \max_a Q(s_t, a_t; \theta\ )$

Execute action a_t and observe reward R_t and new state s_{t+1}

Sort transition (s_t, A, s_{t+1}, R_t) in D

Sample a minibatch of transitions $[S_j, A_j, R_j, S_{j+1}]$ from D

Set $y_j = \begin{cases} r_j & \text{if episode terminates at step } j \\ r_j + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}; \theta_t) & \text{otherwise} \end{cases}$

Perform a gradient on $(y_j - Q(s_t, a_t; \theta_t))$ with respect to NN parameters θ

Reset $\hat{\theta} = \theta$ every C steps

End For

End For

How to choose parameters?

State: more parameters, more data to train

Less parameters —— ever-changing environment

[current ,offset value ,orbit value] – 18 dims

Action: [x, x', y, y'] → [a0,a1,a2,a3,a4,a5,a6,a7]

Reward: fast response and low noise

small-angle luminosity

How to train our model?

Random policy search? ✗

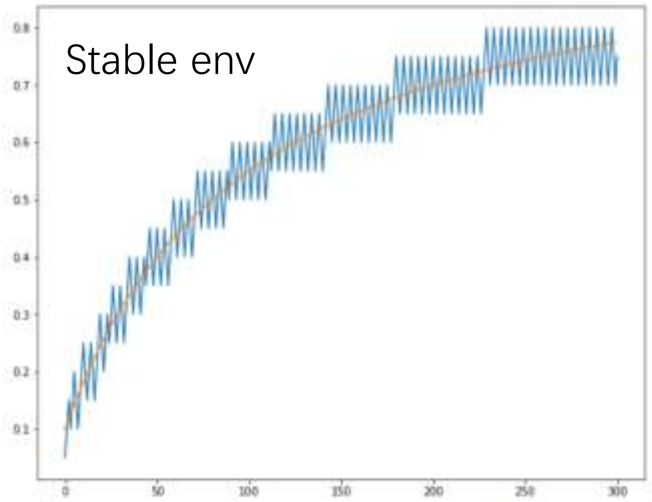
History data of manual operation? ✗

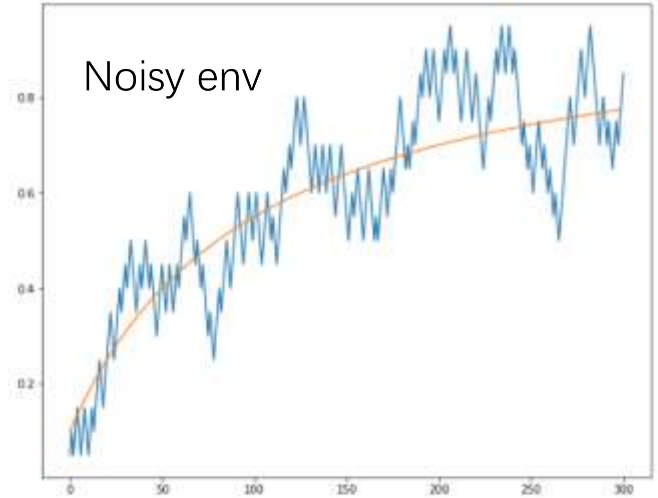
Data from simulation ? ✗

Reinforcement learning control for BEPCII

How to get perfect history data? —— Dithering search method

Algorithm 2: Dithering Search
Initialize replay memory D to capacity N
Initialize step length array M with the same dimensions as knobs
Observe initial state S_0 initial reward R_0 and action $A = A_0$
For $t = 1, T$ **do**
    Initialize activate dimension pointer $d = 0$
    Set $A[d] = A[d] + M[d]$ #run a step on dimension d
    Execute action A and observe reward R_t and new state s_{t+1}
    Sort transition (s_t, A, s_{t+1}, R_t) in D
        If $R_t < R_0$ **do**   #If target falls, turn around and continue
            $M[d] = -M[d]$
        Else do   #If target improve, jump to another dimension
            d=d+1
        End If
End For

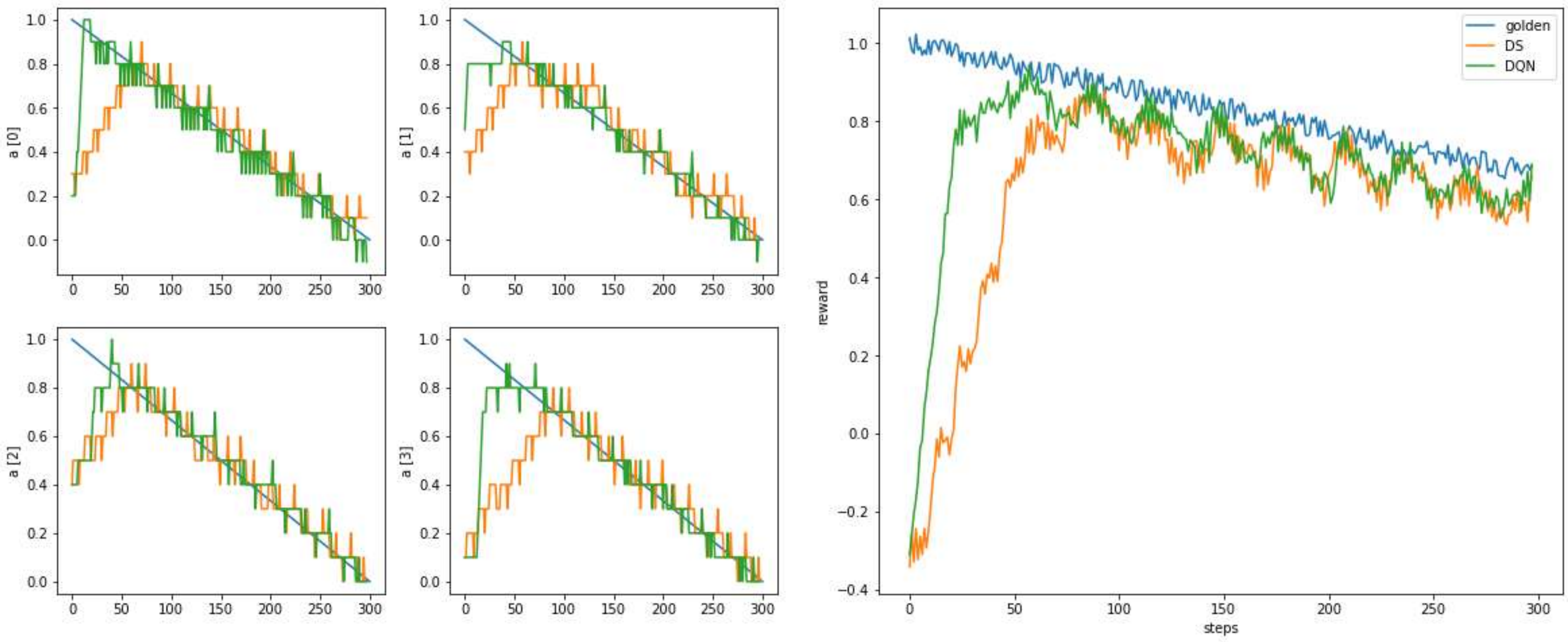




Reinforcement learning control for BEPCII

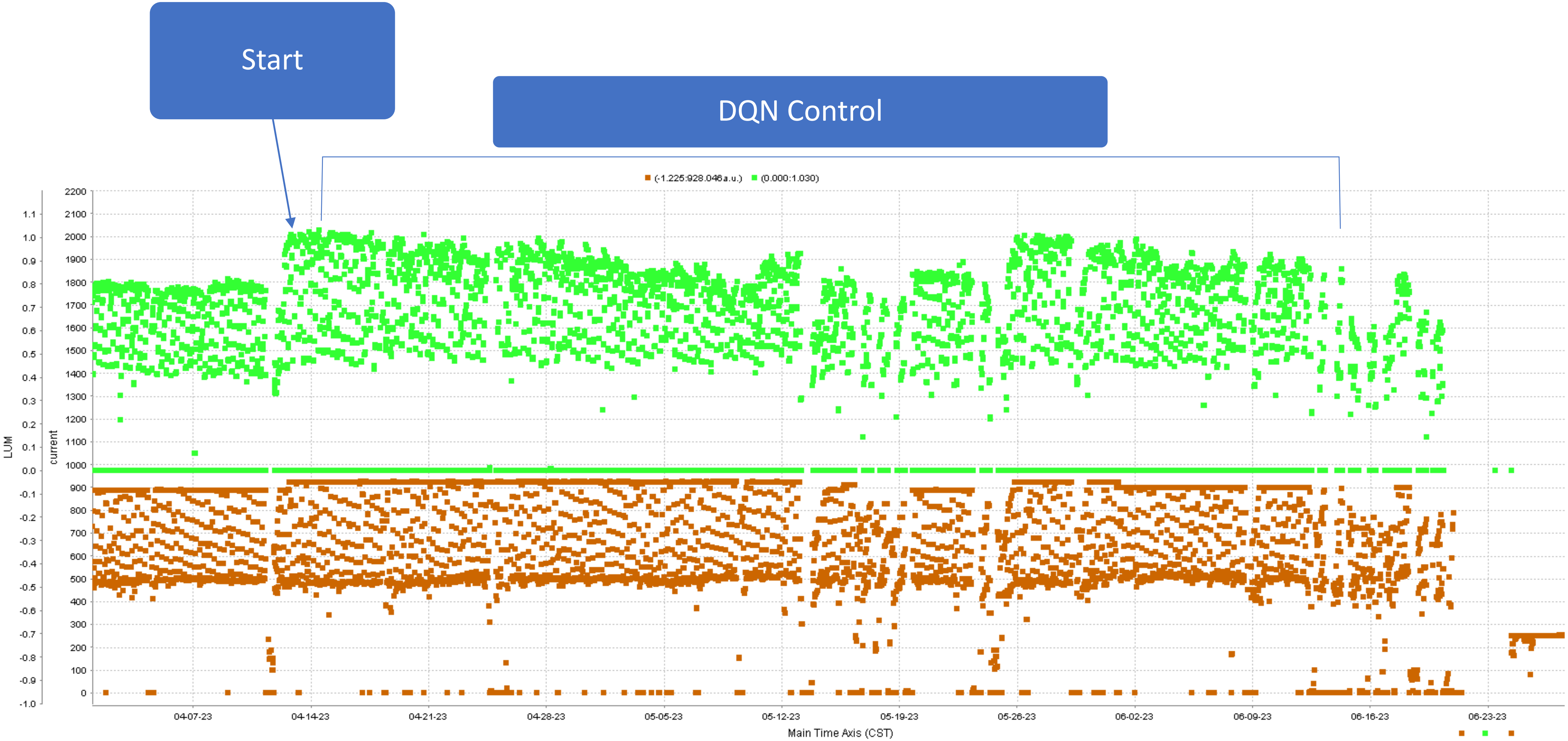
Simulation:

```
L=I[0]/900-abs(knob[0]-(I[0]-500)/300)-abs(knob[1]-(I[1]-750)/300)\
        -abs(knob[2]-(I[0]-700)/300)-abs(knob[3]-(I[1]-650)/300)
```

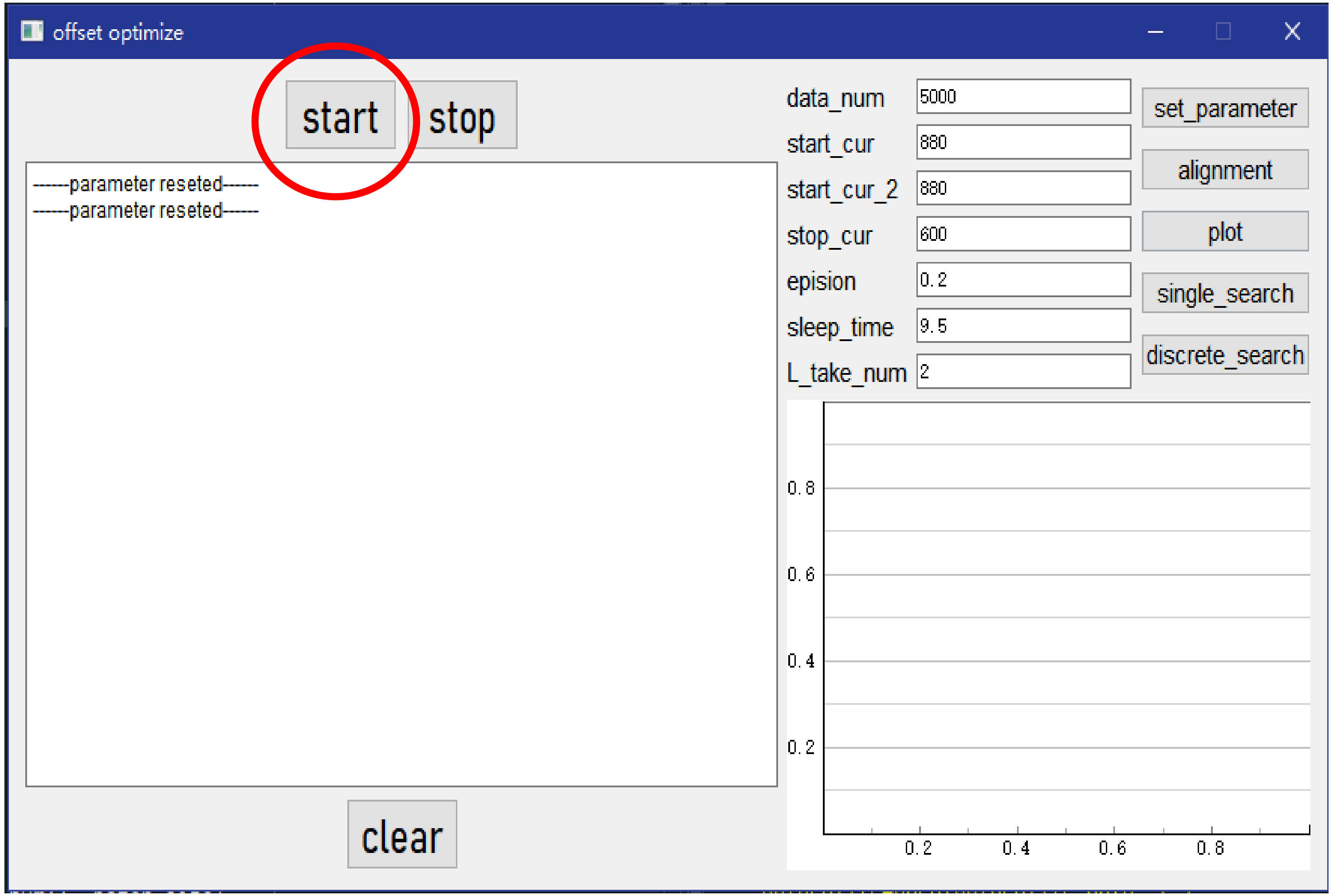


Reinforcement learning control for BEPCII

The method has been used about 1 months:

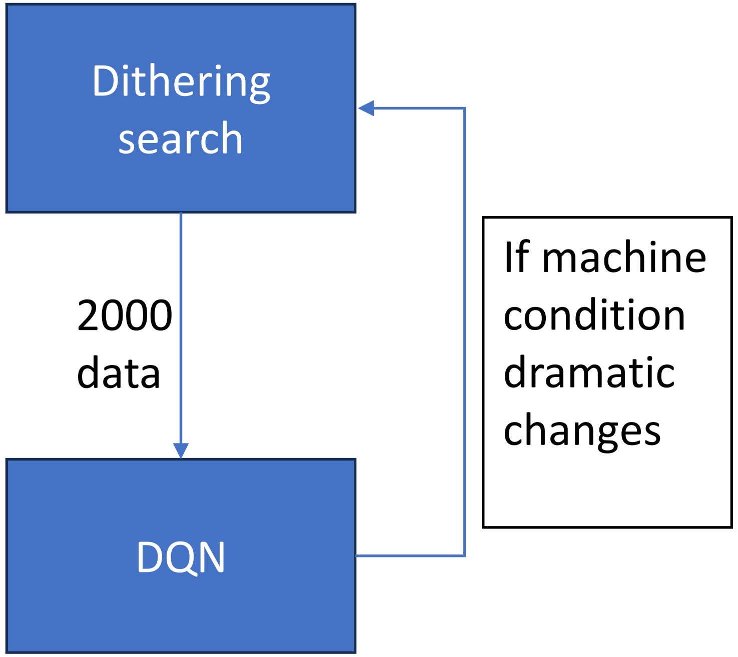


Reinforcement learning control for BEPCII



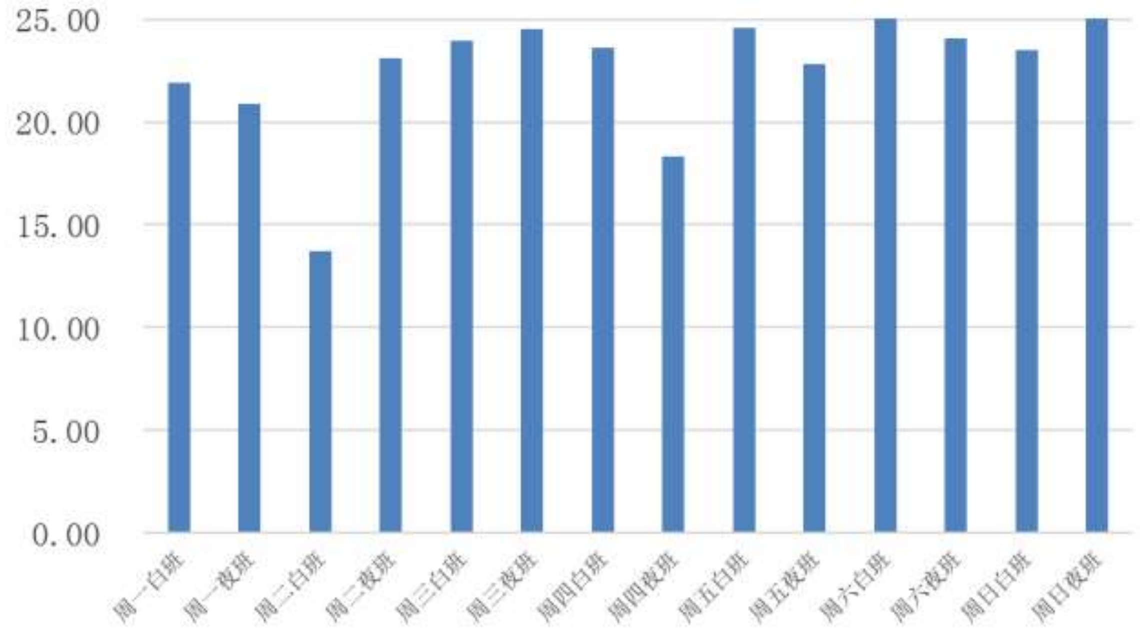
Hyper-parameters:

- Training data num: 5000
- Start current: 880
- Stop current: 600
- Exploration rate: 0.2
- Gamma: 0.5
- Waiting time: 7.5
- Lum get times: 3

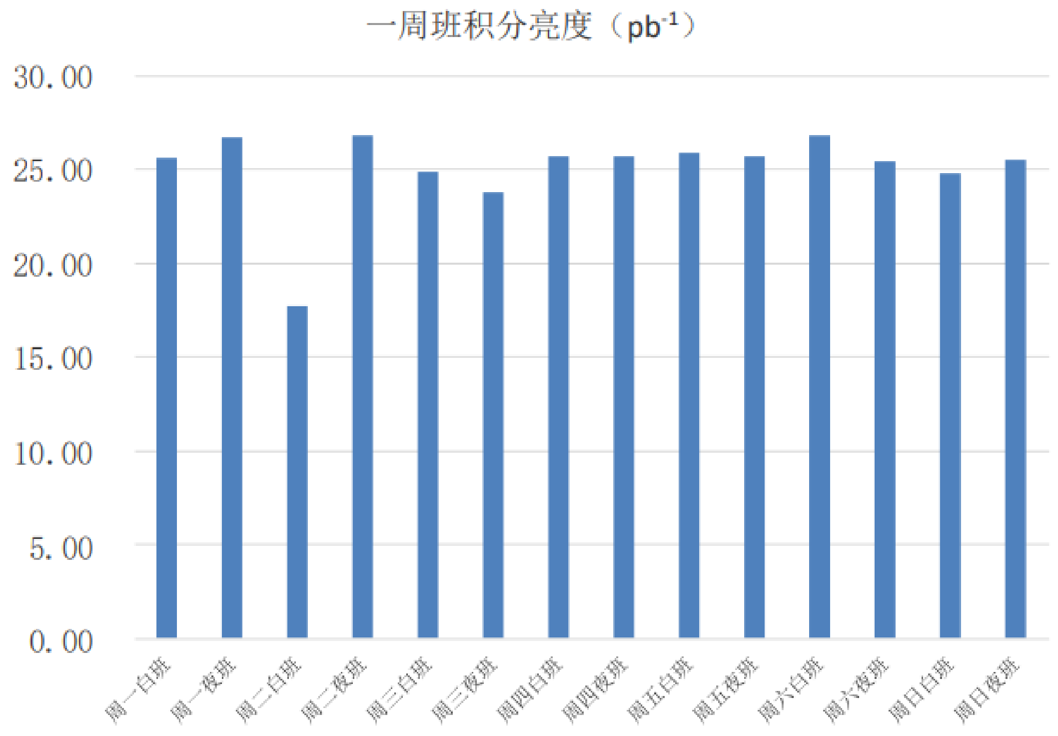


Result

Make different operator to reach the same operation level



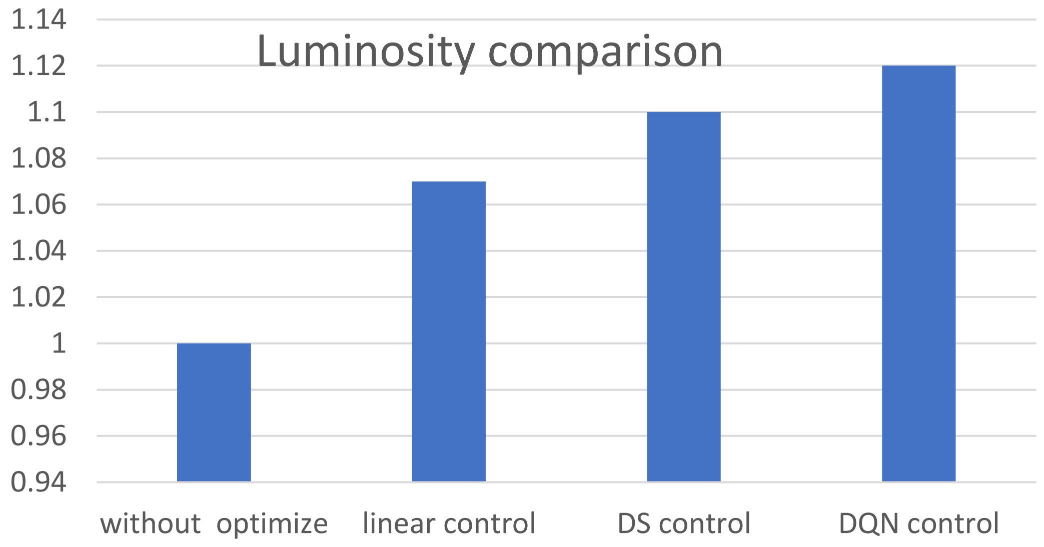
Before



After

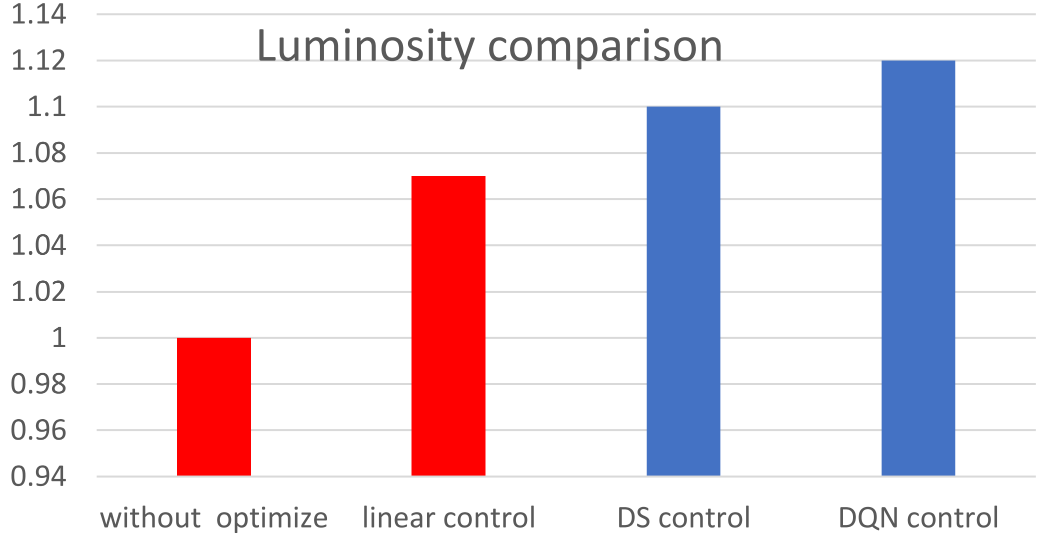
Results

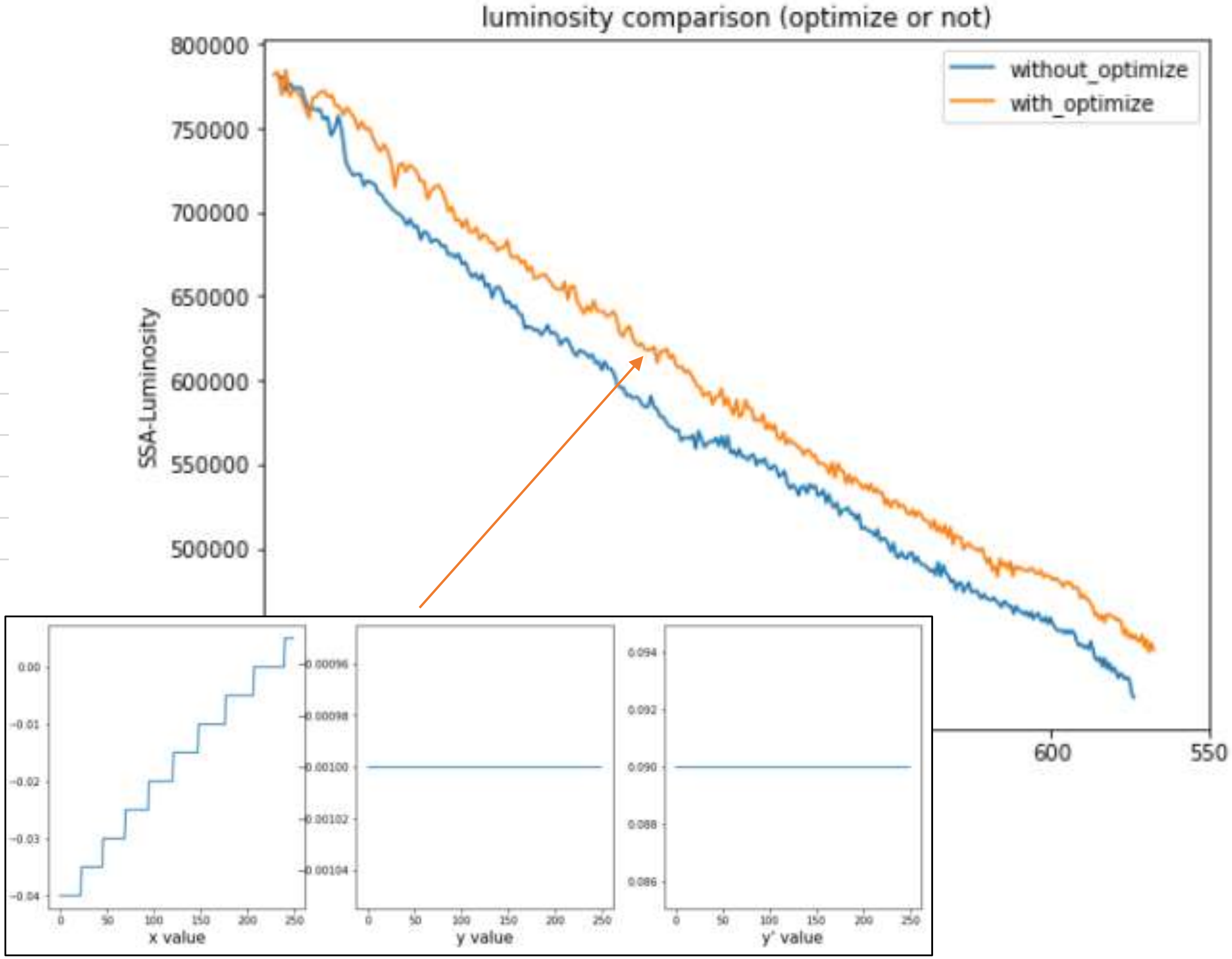
Increase on Luminosity.



Results

Increase on Luminosity.

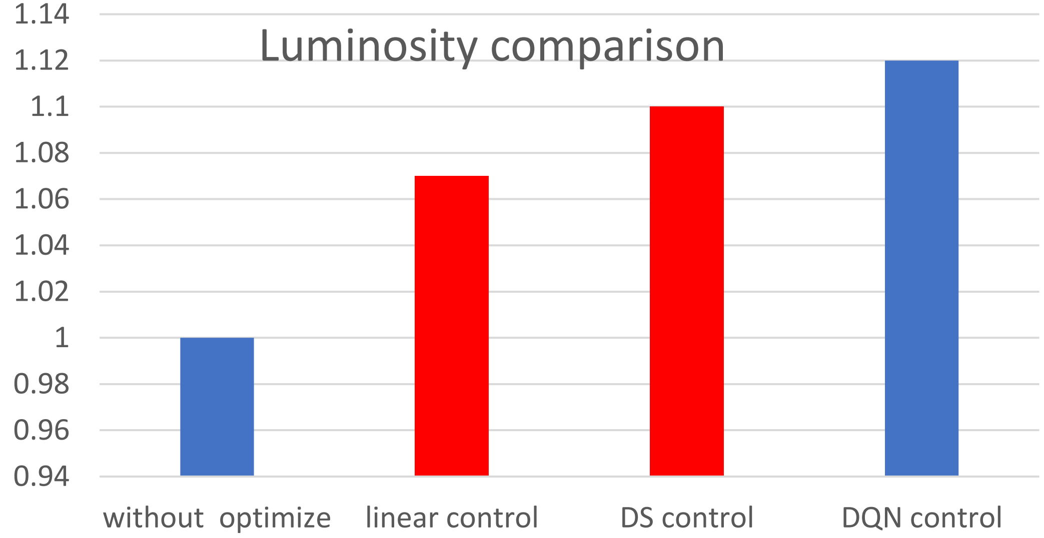


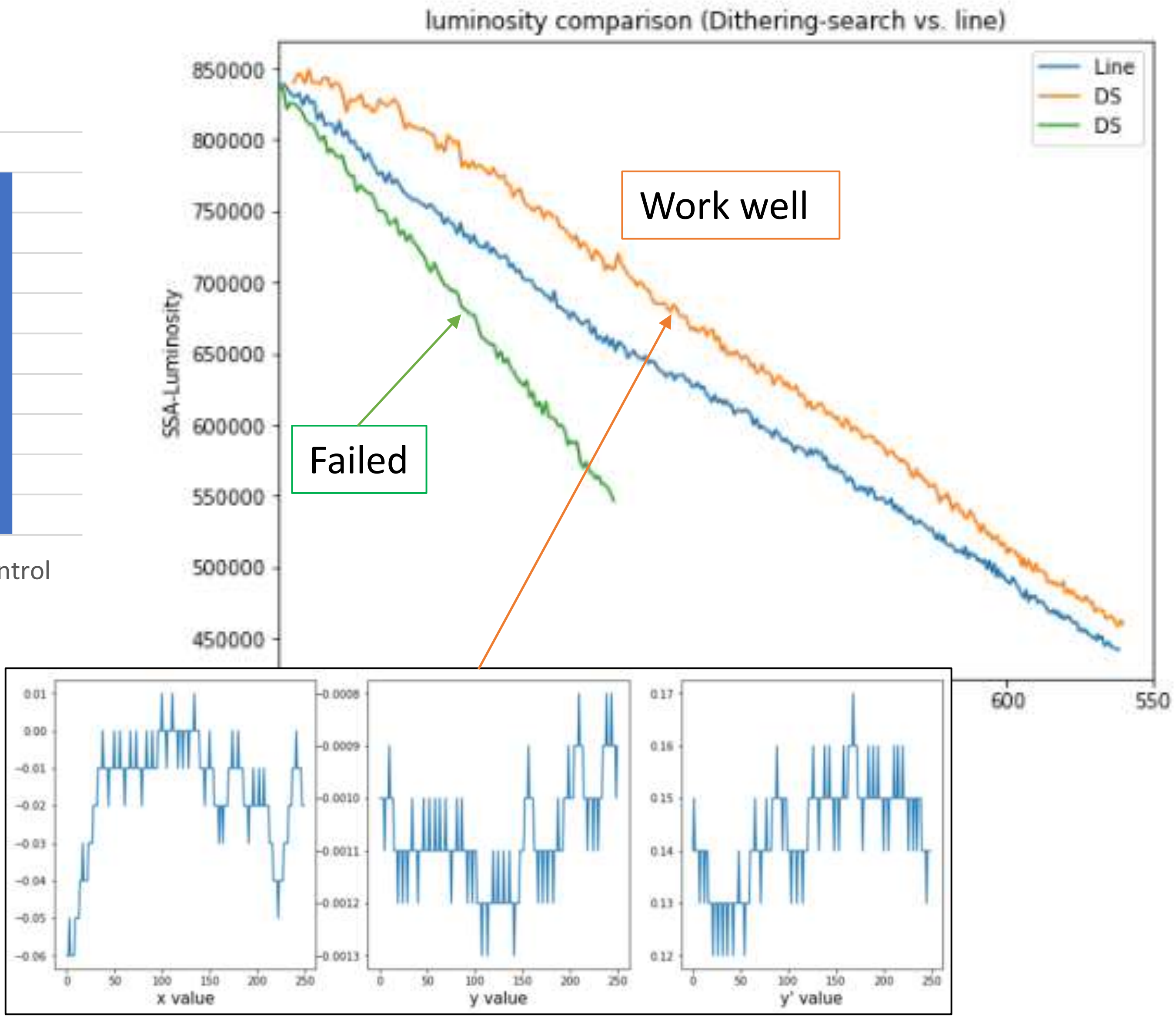


Linear method make about 5%-10% improvement with no control

Results

Increase on Luminosity.

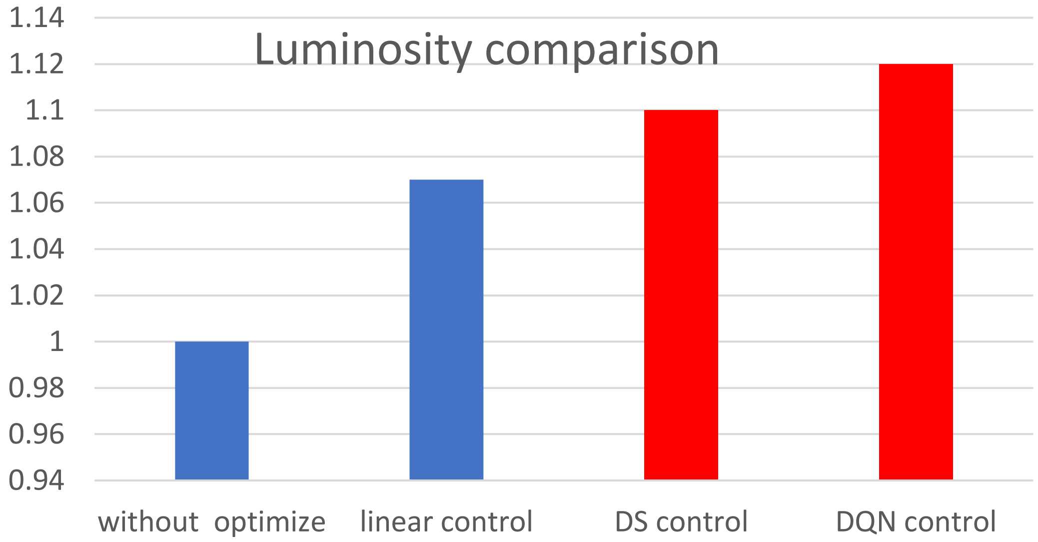


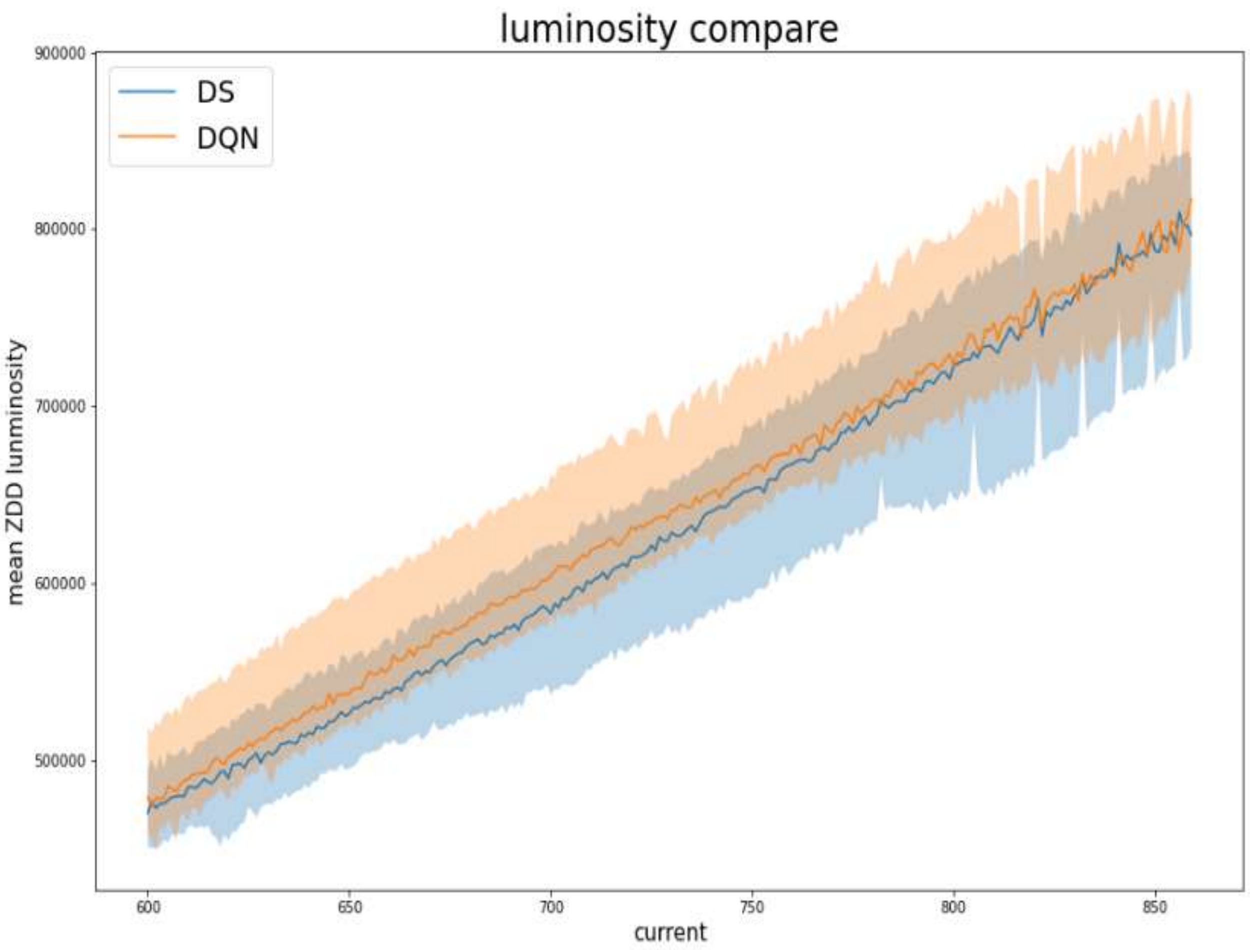


DS method make about 2%-4% improvement with linear control

Results

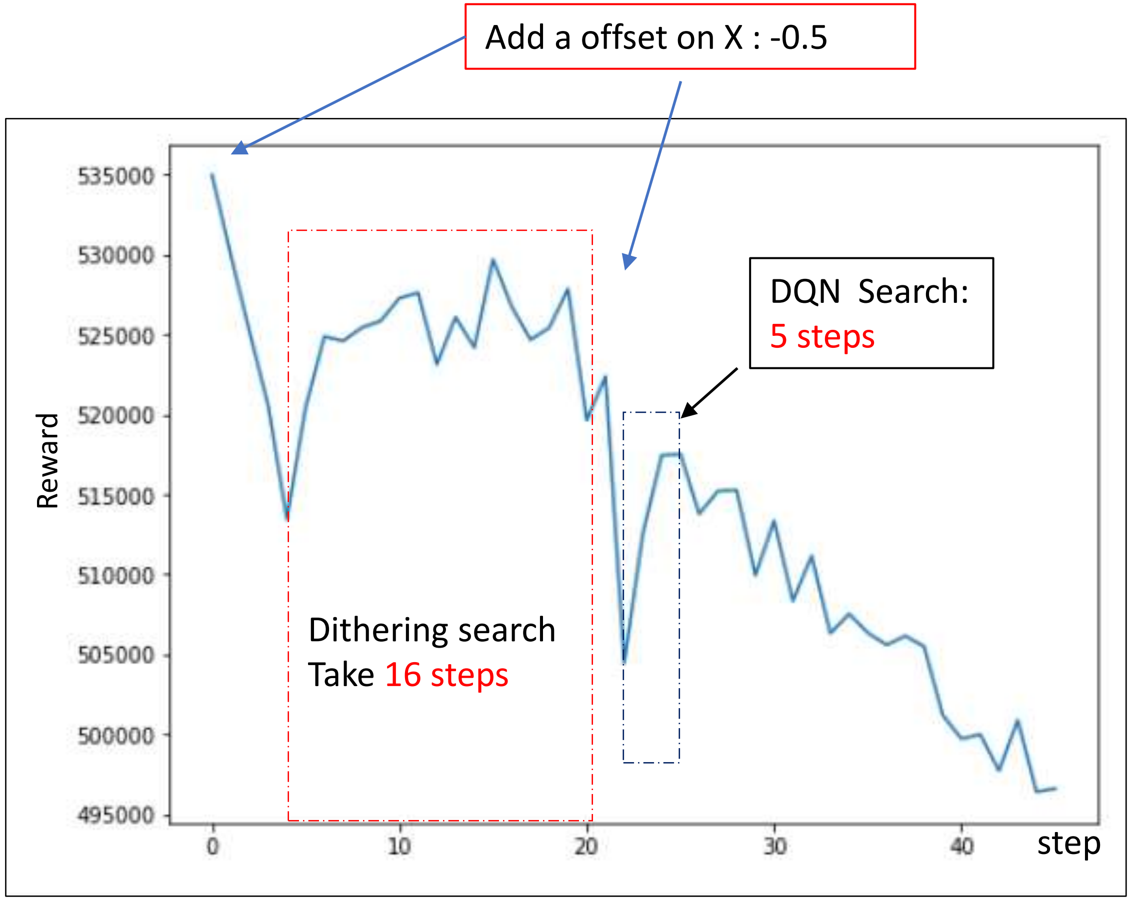
Increase on Luminosity.

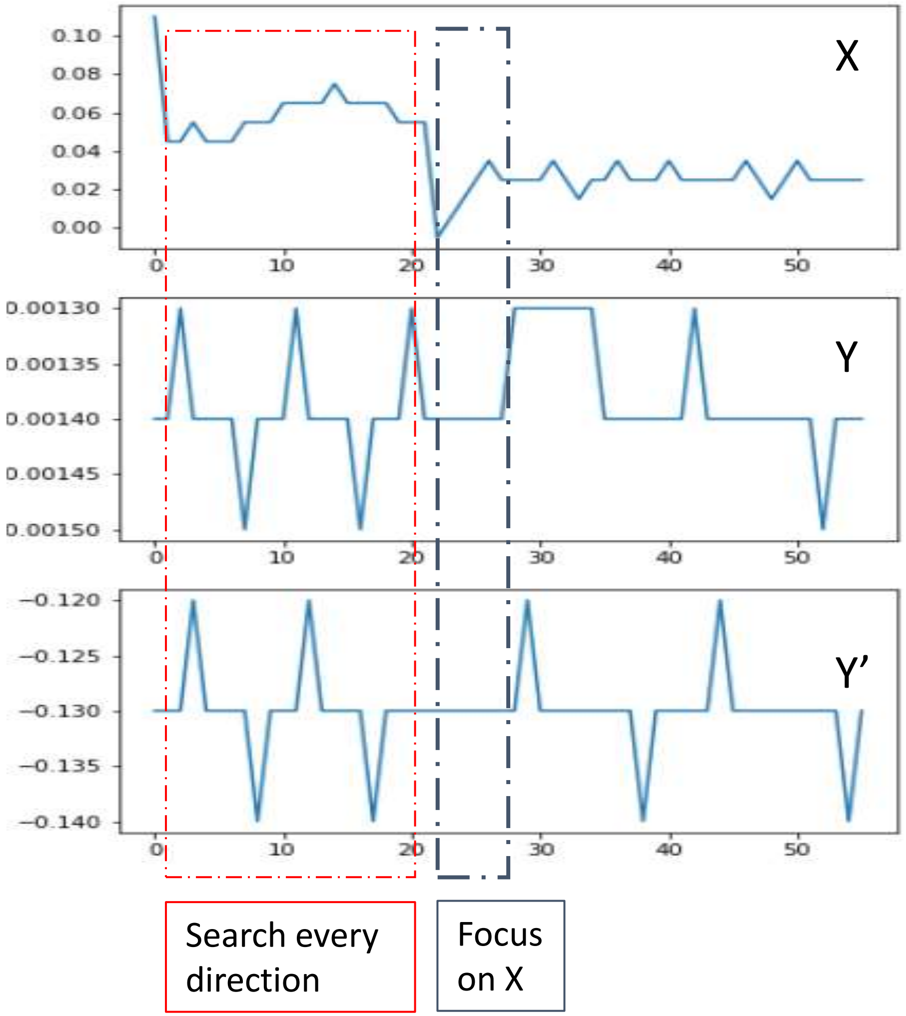




DQN method make about 1.5% improvement with DS control

Results

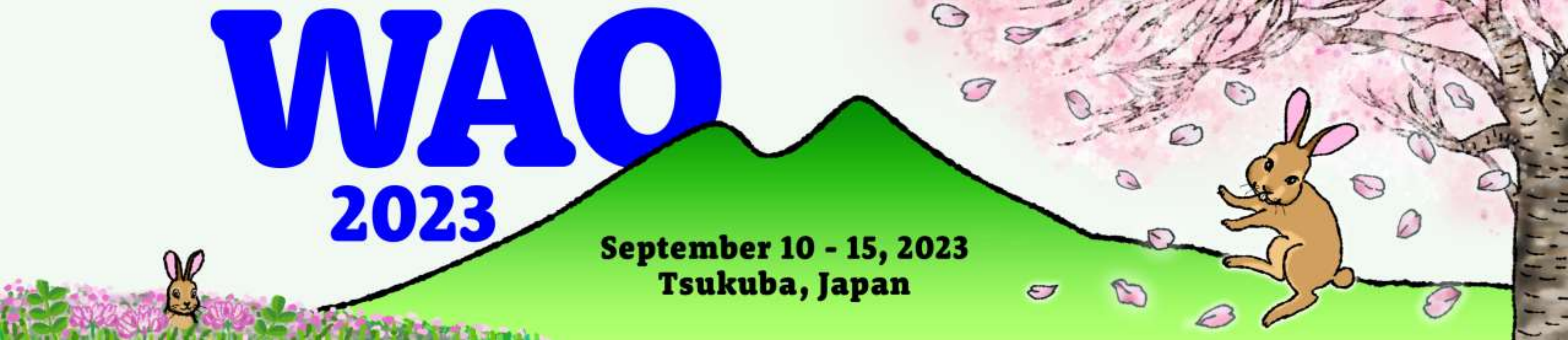

Add a offset on X : -0.5
Reward
535000
530000
525000
520000
515000
510000
505000
500000
495000
0
10
20
30
40
step
Dithering search
Take 16 steps
DQN Search:
5 steps


X
Y
Y'
0.10
0.08
0.06
0.04
0.02
0.00
0.00130
0.00135
0.00140
0.00145
0.00150
−0.120
−0.125
−0.130
−0.135
−0.140
0
10
20
30
40
50
Search every direction
Focus on X

Summary

- Machine learning provide a new approach to solve control problems.
- A reinforcement learning method has been made to control the offset for BEPCII and bringing considerable benefits.
- Operators' experience helps a lot on this task, maybe they don't know machine learning, they know operation more than anyone else.
- Most our operators believe in machine learning even they don't know how it works.

- What is the next?
- Machine learning method used online is always restricted by data. We use a small observation input to reduce the amount of data required, so the environment we made is change slowly. Take more parameters into account or make parameters out of the environment stable is what we are going to do next.



Thank you!