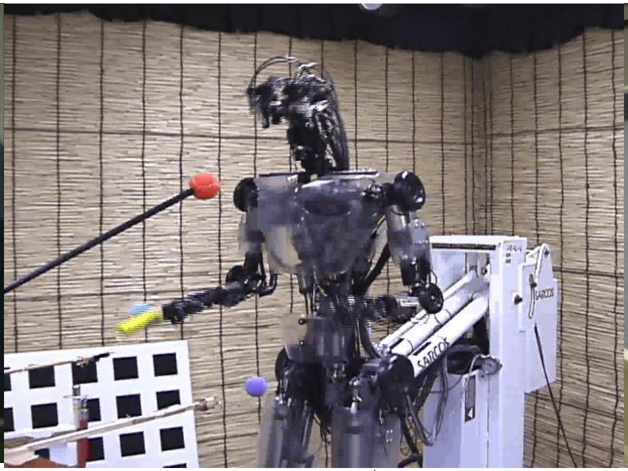
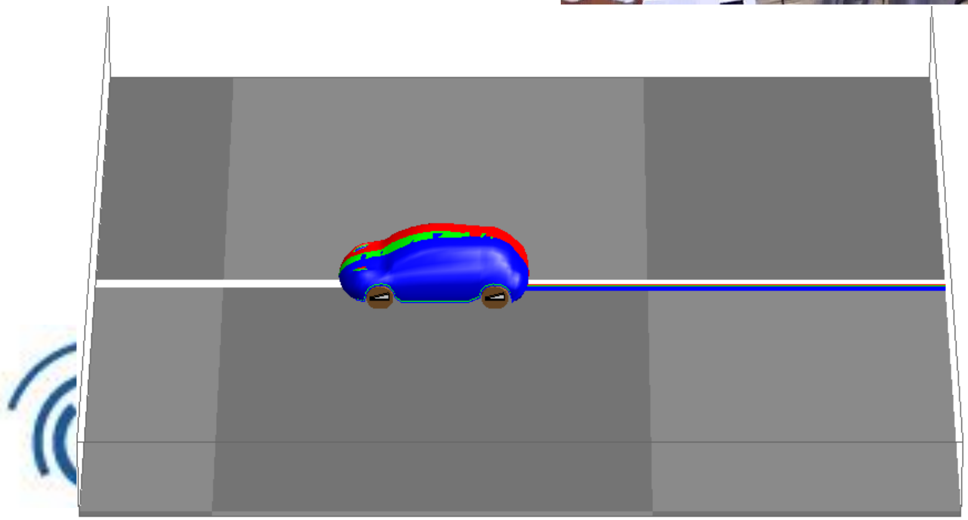
確率推論による順・逆強化学習

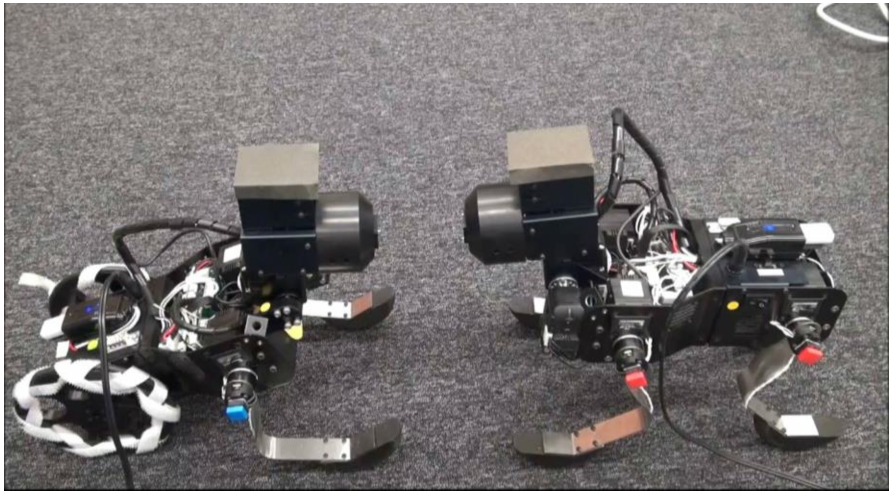
内部英治
ATR 脳情報通信研究所
ブレインロボットインターフェース研究室
主幹研究員

Reinforcement learning

- Computational algorithm to learn a policy (controller) by trial and error



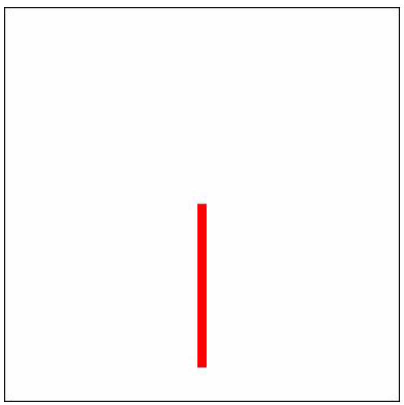




Components

- Inverted Pendulum swing-up and balancing task

environment



state: joint angle and angular velocity

$\boldsymbol{x}_t = [\theta_t, \omega_t]$

policy (controller)

$\pi(\, u \mid \boldsymbol{x} \,)$

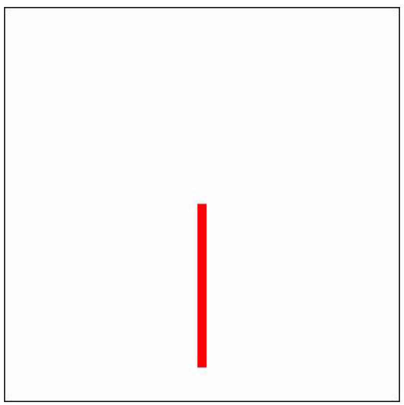
$u_t = \tau_t + \varepsilon_t$

action: torque with exploration noise

Components

- Inverted Pendulum swing-up and balancing task

environment



state transition

$\boldsymbol{x}_{t+1}$

policy (controller)

$\pi(u \mid \boldsymbol{x})$

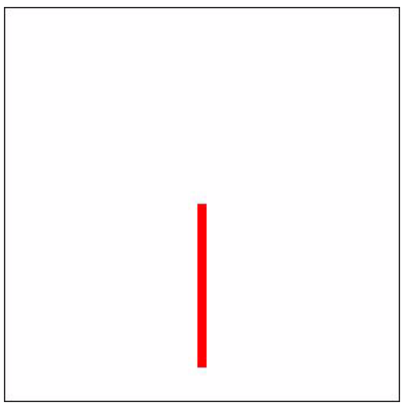
$u_t = \tau_t + \varepsilon_t$

action: torque with exploration noise

Components

- Inverted Pendulum swing-up and balancing task

environment



state transition

$\boldsymbol{x}_{t+1}$

$u_t = \tau_t + \varepsilon_t$

action: torque with exploration noise

reward (immediate evaluation)

$r(\boldsymbol{x}_t, u_t, \boldsymbol{x}_{t+1}) = \cos\theta_t$

value function (performance)

$V(\boldsymbol{x})$, $Q(\boldsymbol{x}, u)$

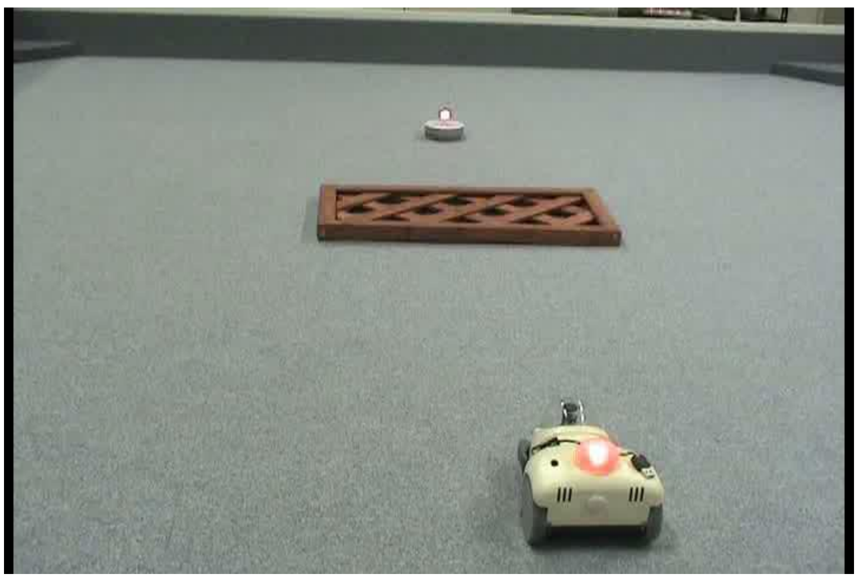
policy (controller)

$\pi(u \mid \boldsymbol{x})$

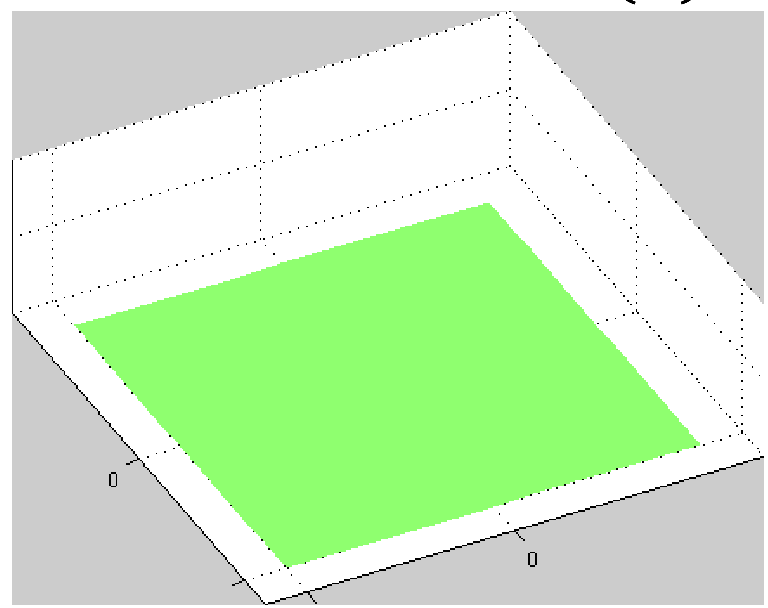
Example

- Task: to get the battery pack while avoiding collisions with an obstacle

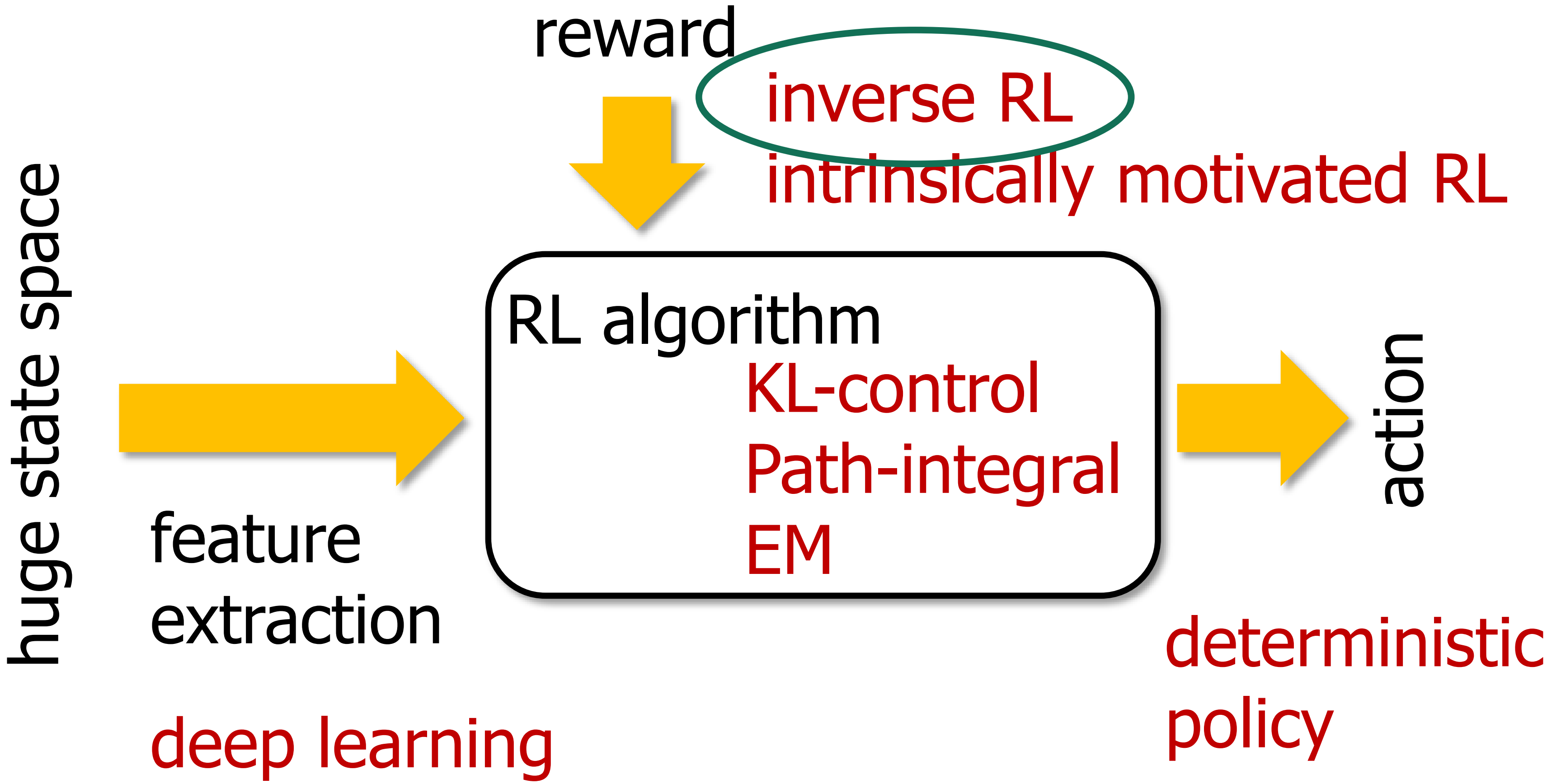
environment



value function $V(x)$



Open Problems in RL

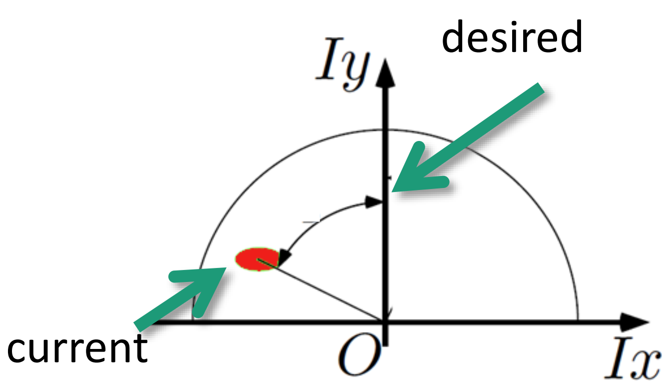


NEDO

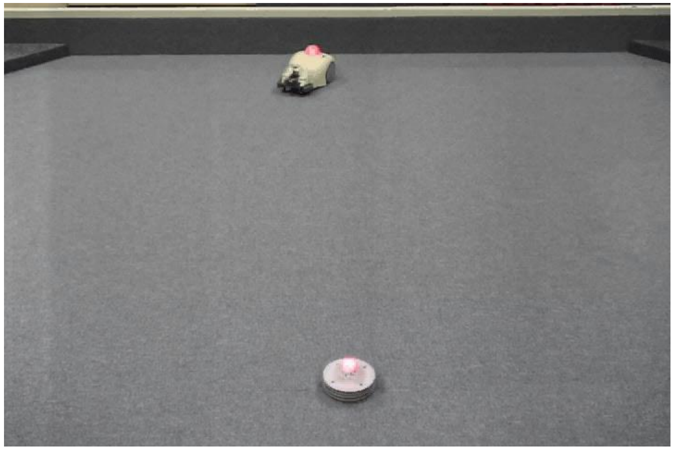
ATR

What is a good reward for learning agents?

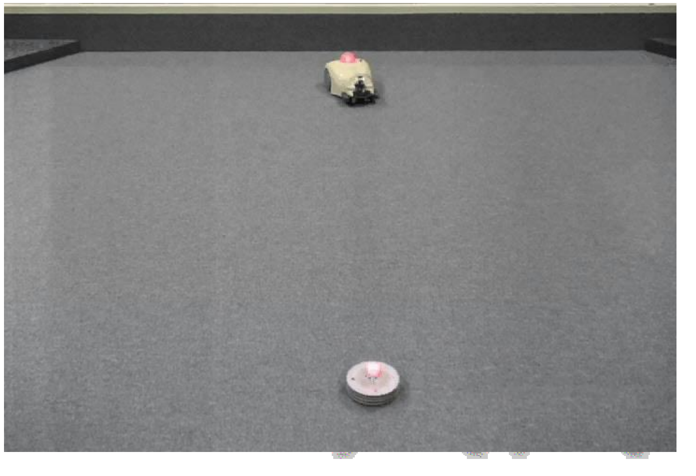
- Original reward for catching a battery pack
- Visual reward calculated from image features



original reward

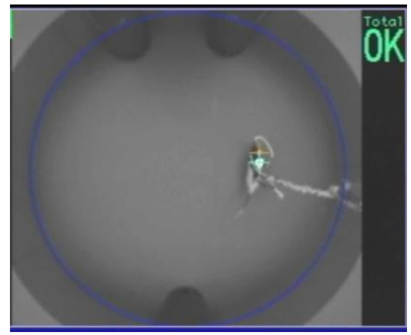


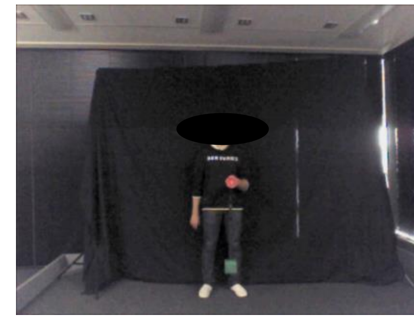
+ visual reward

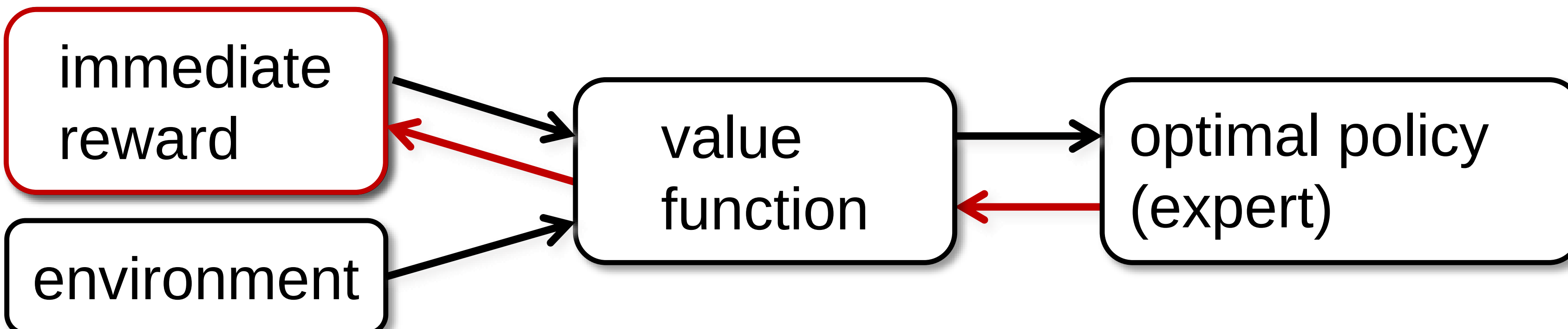


Design of rewards for RL

- How should we prepare an objective function?







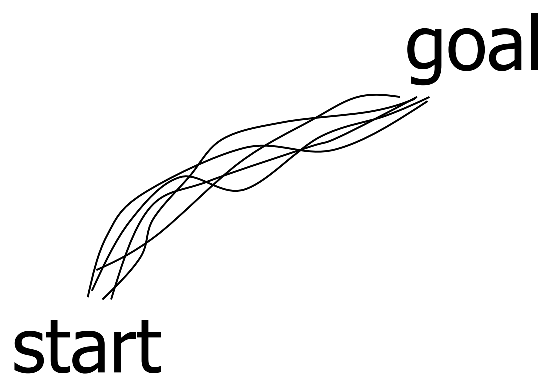
- Inverse Reinforcement Learning: infer the cost function from observed behaviors from the experts

Problems of Inverse Reinforcement Learning

forward RL

inverse RL

[Abbeel and Ng 2004]
[Ratliff et al. 2009]



[Boularias et al. 2011]
[Kalakrishnan et al. 2013]

environmental model

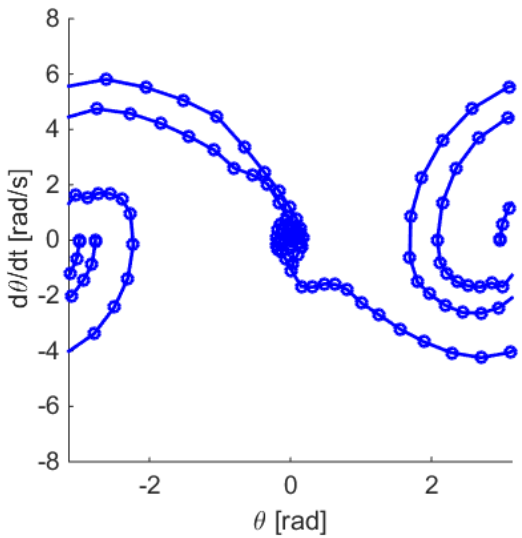
evaluation of partition func.

[Dvijotham and Todorov 2010][Ziebart et al. 2009]

- We propose a data-efficient, model-free inverse reinforcement learning

Standard formulation

- Reward is estimated from a dataset $\mathcal{D}^{\pi} = \{\tau_j^{\pi}\}_{j=1}^{N^{\pi}}$ sampled from the optimal policy π



$$\tau_j^{\pi} = \{\boldsymbol{x}_{j,1}^{\pi}, \boldsymbol{u}_{j,1}^{\pi}, \boldsymbol{x}_{j,2}^{\pi}, \dots, x_{j,T}^{\pi}\}$$

$$\boldsymbol{u}_{j,t}^{\pi} \sim \pi(\cdot | \boldsymbol{x}_{j,t}^{\pi})$$

$$\boldsymbol{x}_{j,t+1}^{\pi} \sim \Pr(\cdot | \boldsymbol{x}_{j,t}^{\pi}, \boldsymbol{u}_{j,t}^{\pi})$$

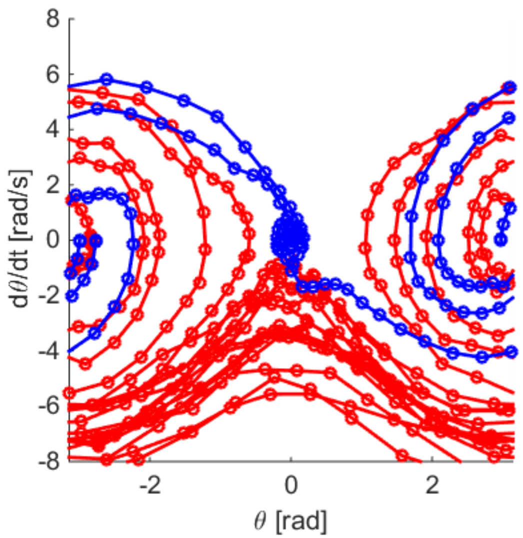
Estimate a reward

- Solve as a density estimation problem

$$p(\tau) \propto \exp\left[\sum_{t=1}^{T} r(\boldsymbol{x}_t^{\pi}, \boldsymbol{u}_t^{\pi})\right]$$

Our formulation

- Reward is estimated from two datasets $\mathcal{D}^{\pi}$ from π and $\mathcal{D}^{b}$ sampled from a baseline policy b



$$\mathcal{D}^{\pi} = \left\{\left(\boldsymbol{x}_j^{\pi}, \boldsymbol{u}_j^{\pi}, \boldsymbol{y}_j^{\pi}\right)\right\}_{j=1}^{N^{\pi}}$$

$$\boldsymbol{u}_j^{\pi} \sim \pi(\cdot \mid \boldsymbol{x}_j^{\pi}) \quad \boldsymbol{y}_j^{\pi} \sim P_T(\cdot \mid \boldsymbol{x}_j^{\pi}, \boldsymbol{u}_j^{\pi})$$

$$\mathcal{D}^{b} = \left\{\left(\boldsymbol{x}_j^{b}, \boldsymbol{u}_j^{b}, \boldsymbol{y}_j^{b}\right)\right\}_{j=1}^{N^{b}}$$

$$\boldsymbol{u}_j^{b} \sim b(\cdot \mid \boldsymbol{x}_j^{b}) \quad \boldsymbol{y}_j^{b} \sim P_T(\cdot \mid \boldsymbol{x}_j^{b}, \boldsymbol{u}_j^{b})$$

Estimate a reward and a value function

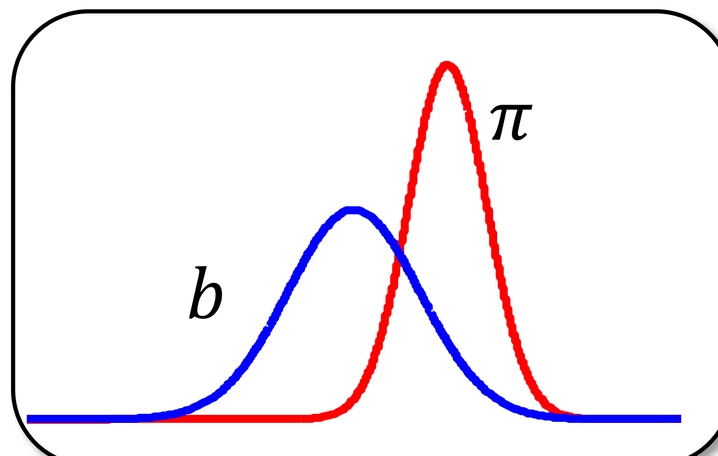
- Solve as a density ratio estimation problem

Reward function restricted by KL divergence

- An action cost is measured by KL divergence between the optimal policy and a baseline policy

$$r(\boldsymbol{x},\boldsymbol{u}) = q(\boldsymbol{x}) - \frac{1}{\beta}\mathrm{KL}\big(\pi(\boldsymbol{u} \mid \boldsymbol{x}) \parallel b(\boldsymbol{u} \mid \boldsymbol{x})\big)$$

  - β: inverse temperature
  - $q(\boldsymbol{x})$: state reward
- Similar constraints used in path-integral RL, KL-control, LMDP, and so on.



[Todorov 2009; Azar et al. 2012]

Bellman Equation for IRL

- Under the constraint on the reward

$$V(\boldsymbol{x}) = \max_{\pi} \int \pi(\boldsymbol{u} \mid \boldsymbol{x}) \left[q(\boldsymbol{x}) - \frac{1}{\beta} \ln \frac{\pi(\boldsymbol{u} \mid \boldsymbol{x})}{b(\boldsymbol{u} \mid \boldsymbol{x})} + \gamma \int P_T(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{u}) V(\boldsymbol{y}) d\boldsymbol{y} \right] d\boldsymbol{u}$$

  - $V(\boldsymbol{x})$: state value func.
  - γ: discount factor
  - P_T: state transition prob.

Minimize the R.H.S. by the Lagrangian multiplier method

$$\boxed{\ln \frac{\pi(\boldsymbol{u} \mid \boldsymbol{x})}{b(\boldsymbol{u} \mid \boldsymbol{x})} = \beta \left[q(\boldsymbol{x}) + \gamma \int P_T(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{u}) V(\boldsymbol{y}) d\boldsymbol{y} - V(\boldsymbol{x}) \right]}$$

Bellman Equation for IRL

- When the action $\boldsymbol{u}$ is observable

$$\ln \frac{\pi(\boldsymbol{u} \mid \boldsymbol{x})}{b(\boldsymbol{u} \mid \boldsymbol{x})} = \beta \left[q(\boldsymbol{x}) + \gamma \int P_T(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{u}) V(\boldsymbol{y}) d\boldsymbol{y} - V(\boldsymbol{x}) \right]$$

- When the action $\boldsymbol{u}$ is unobservable

$$\ln \frac{\pi(\boldsymbol{y} \mid \boldsymbol{x})}{b(\boldsymbol{y} \mid \boldsymbol{x})} = \beta [q(\boldsymbol{x}) + \gamma V(\boldsymbol{y}) - V(\boldsymbol{x})]$$

- This can be considered as a density ratio estimation problem [Sugiyama et al. 2012]

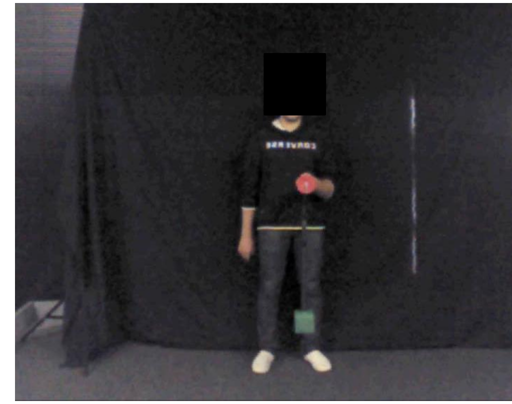
NEDO ATR

Comparison

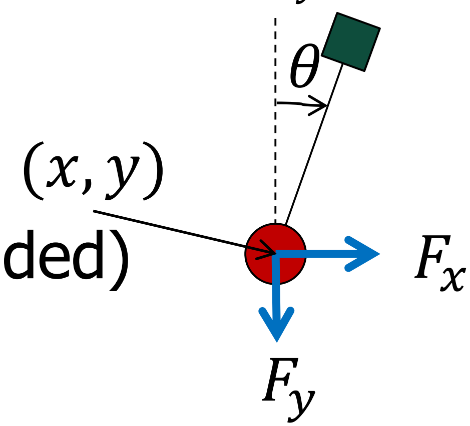
	Proposed	OptV	MaxEnt	RelEnt
model-free?	Yes	No	No	Yes
data	state transition		trajectory	
forward RL?	No	No	Yes	No
partition function?	No	Yes	Yes	partially yes

Inverted pendulum task

- The goal is to swing up and keep the pole upright for more than 3 [s]
- Task conditions:
  - length: long (73 cm), short (29 cm)
  - 15 trials for each pole
  - 40 [s] for each trial



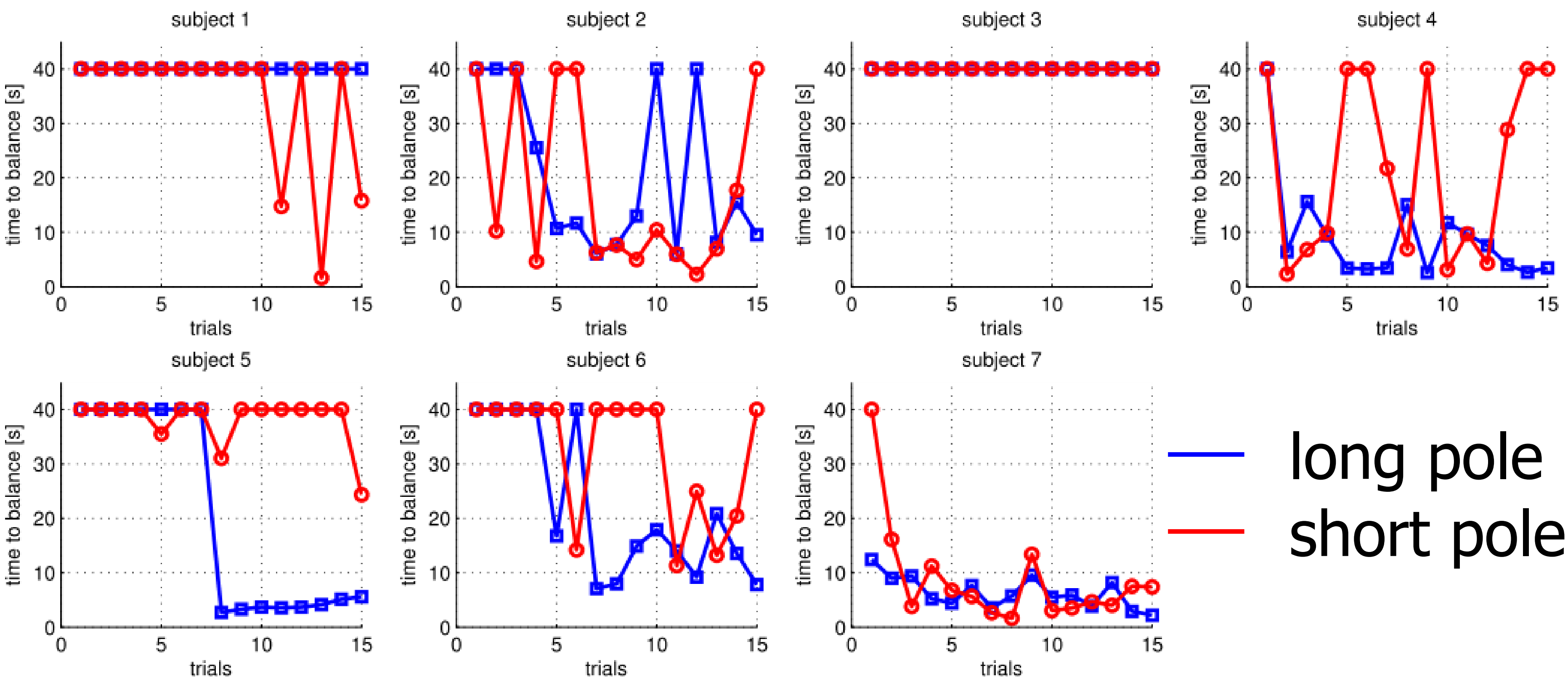
- State: $(x, \dot{x}, y, \dot{y}, \theta, \dot{\theta})$
- Action: (F_x, F_y)



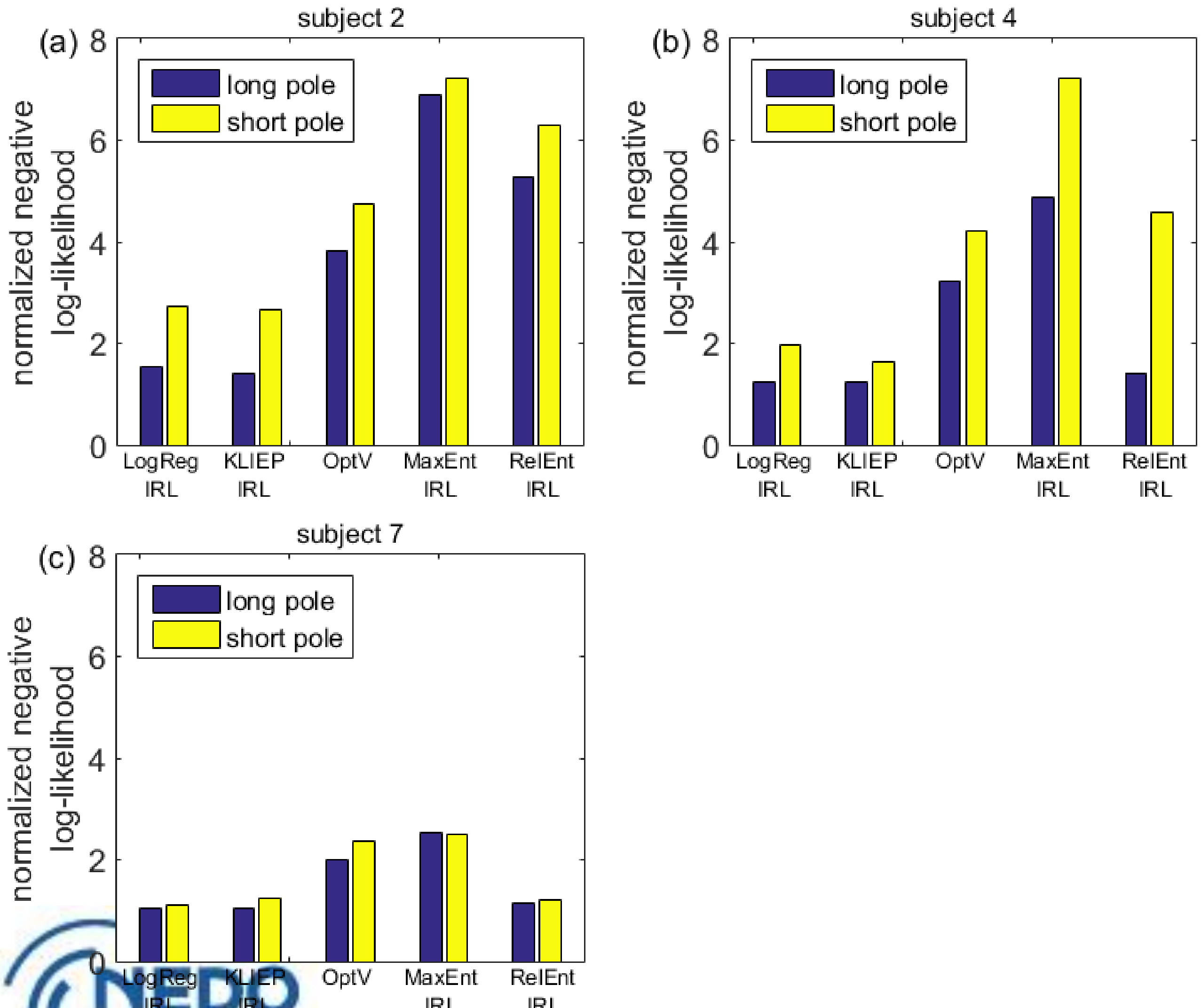
long pole → short pole

  - 7 subjects (5: right-handed, 2: left-handed)

Time to balance the pendulum

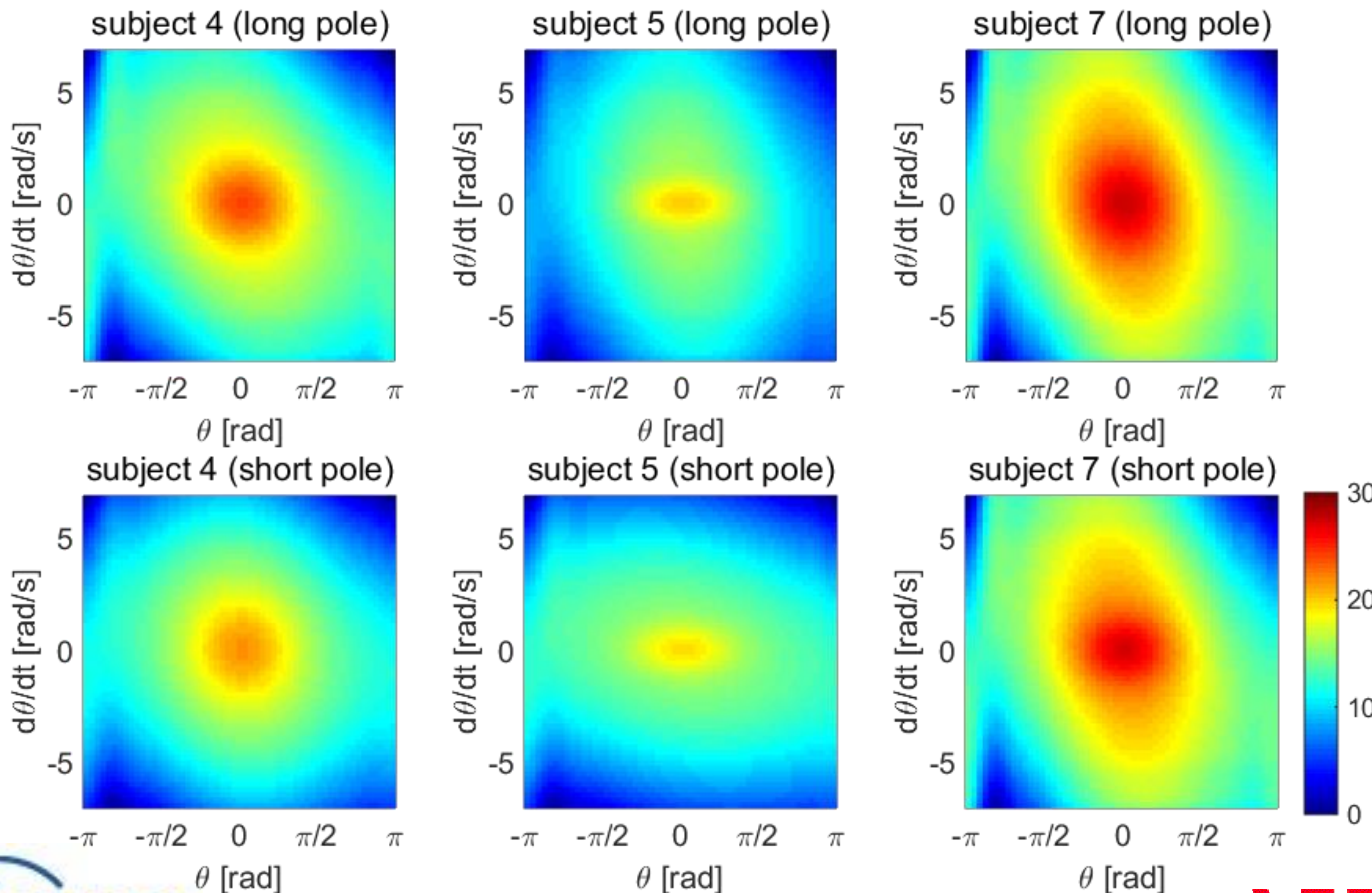


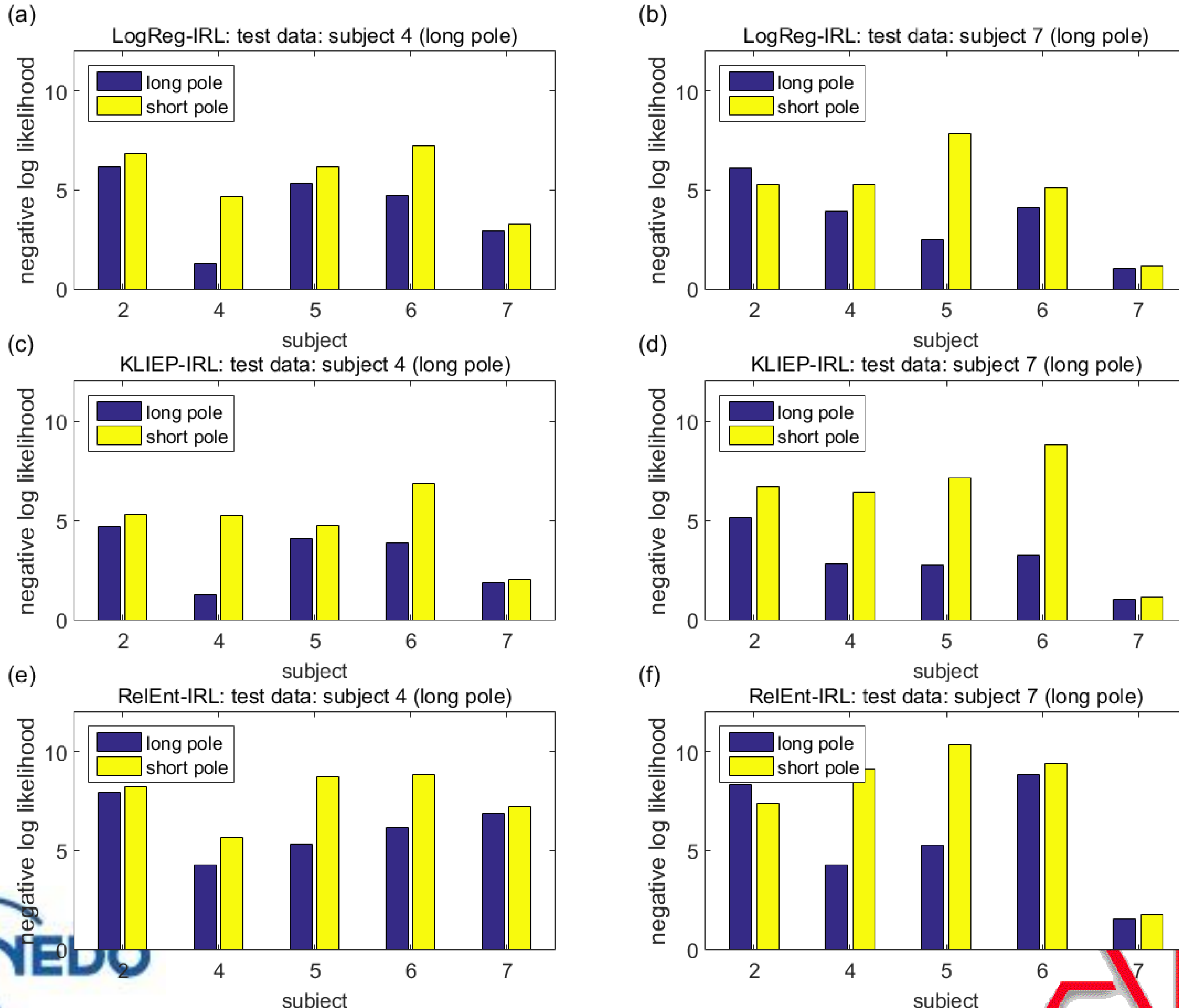
Comparison among the methods



- Proposed: LogReg-IRL KLIEP-IRL

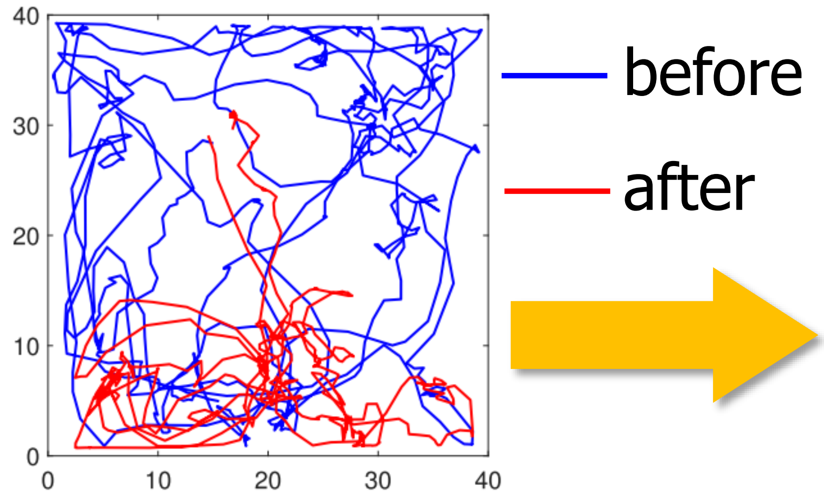
Estimated reward functions

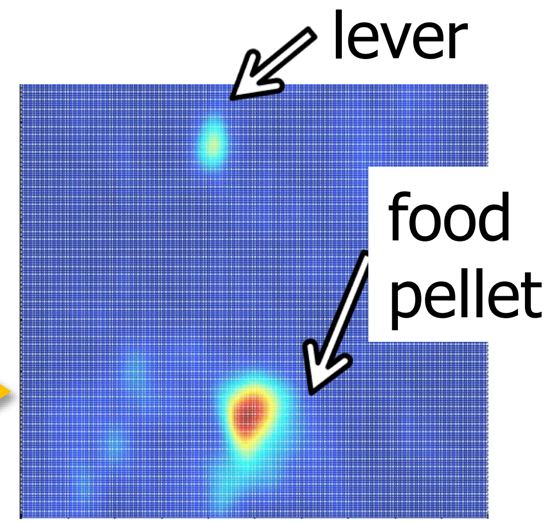



(a)
LogReg-IRL: test data: subject 4 (long pole)
(b)
LogReg-IRL: test data: subject 7 (long pole)
(c)
KLIEP-IRL: test data: subject 4 (long pole)
(d)
KLIEP-IRL: test data: subject 7 (long pole)
(e)
RelEnt-IRL: test data: subject 4 (long pole)
(f)
RelEnt-IRL: test data: subject 7 (long pole)
long pole
short pole
negative log likelihood
subject
0
5
10
2
4
5
6
7

Analysis on rat's behavior

environment





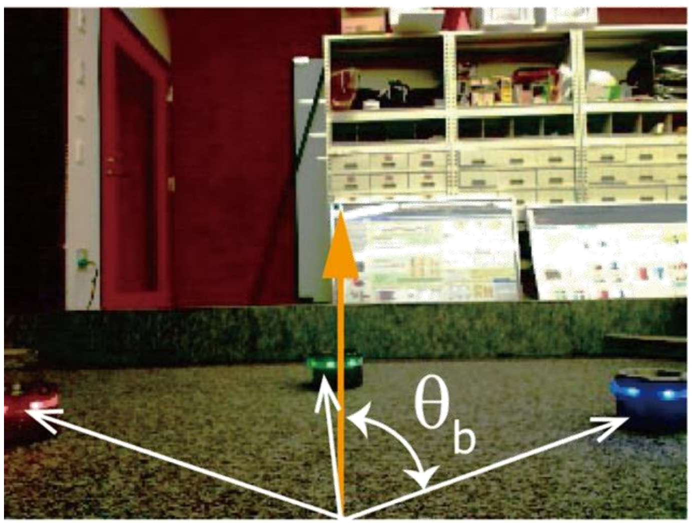
estimated reward

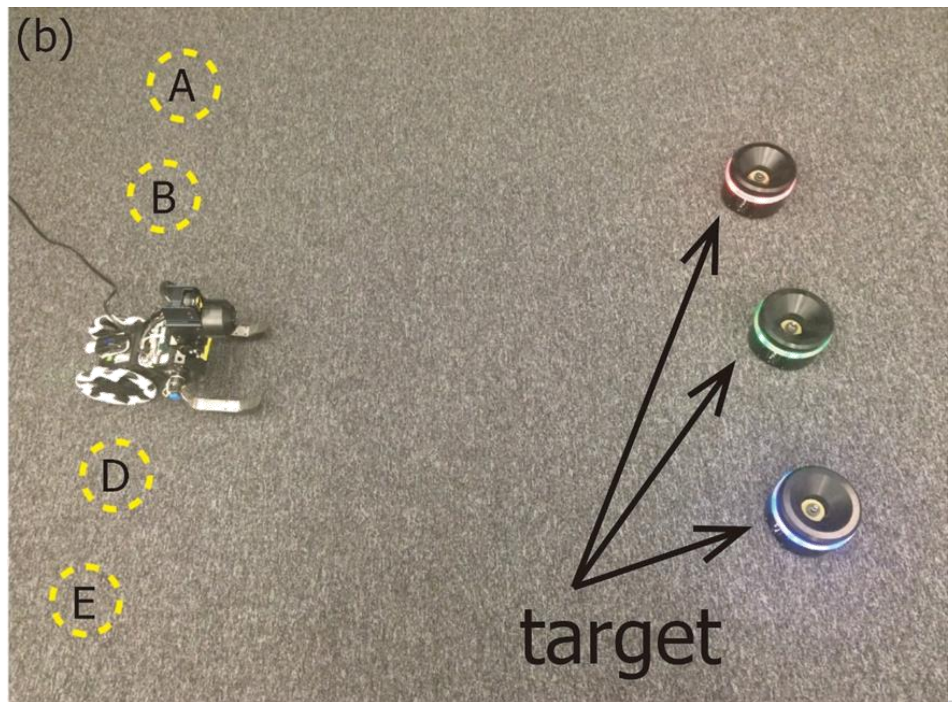
- A rat learned to press an appropriate lever according to a tone stimulus
- We collected the behaviors of the rat before and after learning

Robot Navigation Task

- The task is to reach the green target
  - training data: start position (A-C, E)
  - test data: start position (D)

(a)





Robot Navigation Task

- π and b were given by experimenters

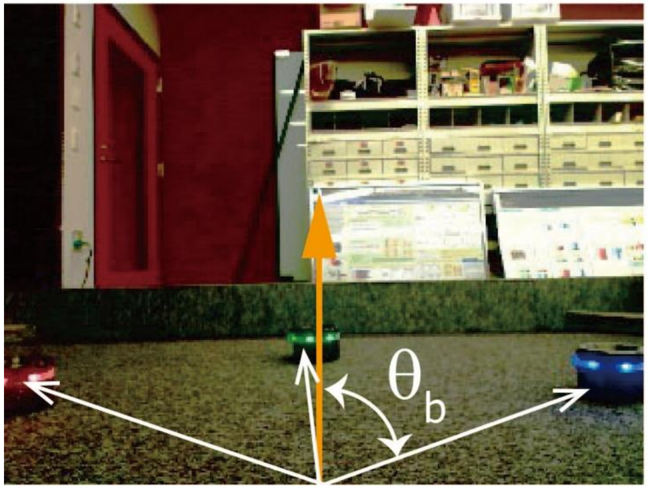
b: baseline

π: optimal

- For every starting point, 10 trajectories were collected to create the datasets.

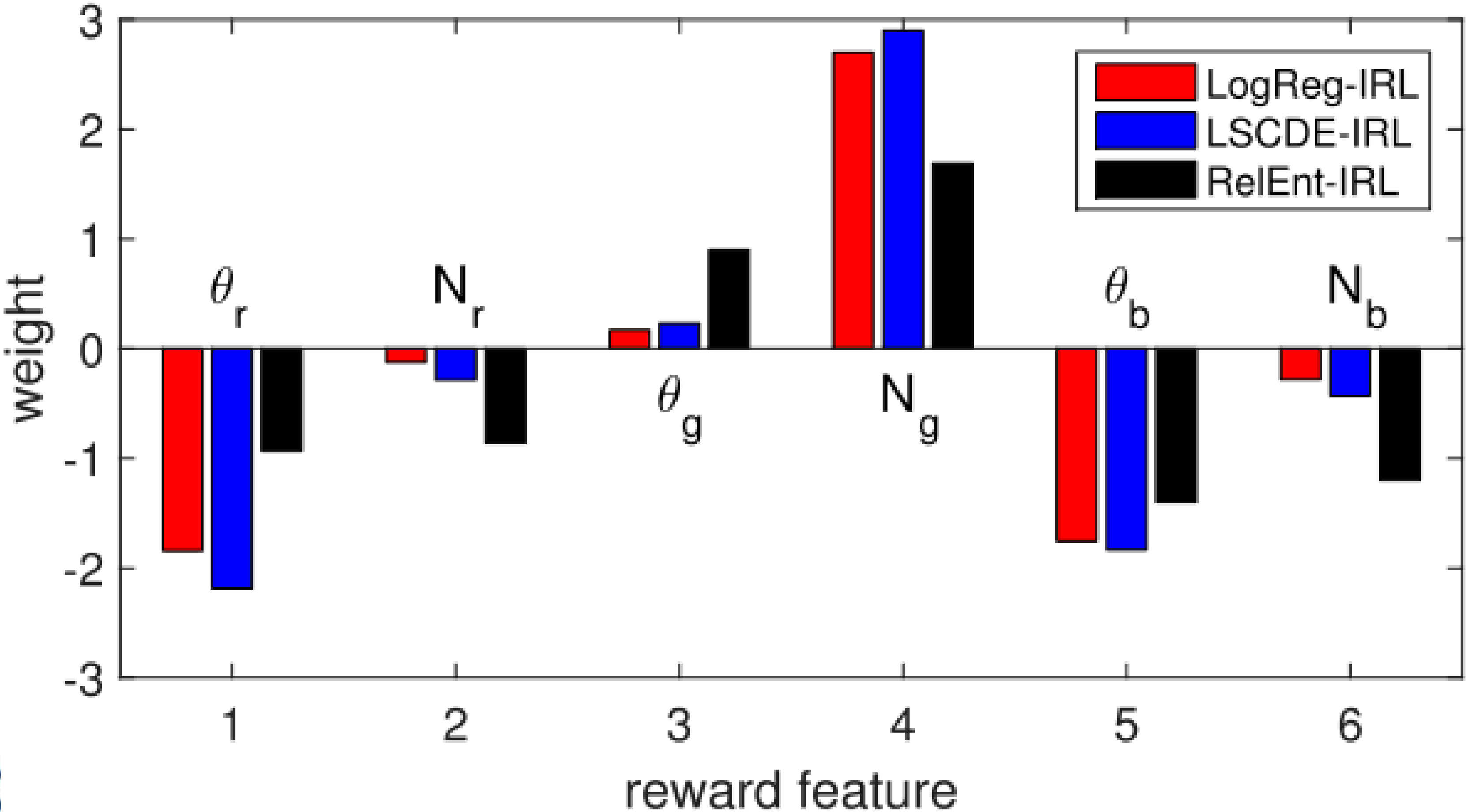
Robot Navigation Task

- state vector: $\boldsymbol{x} = [\theta_r, N_r, \theta_g, N_g, \theta_b, N_b, \theta_{\text{pan}}, \theta_{\text{tilt}}]^{\mathsf{T}}$
  - θ_i $(i = r, g, b)$: angle to the target
  - N_i $(i = r, g, b)$: blob size
  - $\theta_{\text{pan}}, \theta_{\text{tilt}}$: angles of the camera
- basis function for $V(\boldsymbol{x})$
  $\psi_{V,i}(\boldsymbol{x}) = \exp(-\|\boldsymbol{x} - \boldsymbol{c}_i\|^2 / 2\sigma^2)$
  - $\boldsymbol{c}_i$: center position selected from the data set
- basis function for $q(\boldsymbol{x})$
  $\boldsymbol{\psi}_q(\boldsymbol{x}) = [f_{\mathrm{g}}(\theta_r), f_s(N_r), f_{\mathrm{g}}(\theta_g), f_s(N_g), f_{\mathrm{g}}(\theta_b), f_s(N_b)]^{\mathsf{T}}$
  - f_g: Gaussian function, f_s: sigmoid function



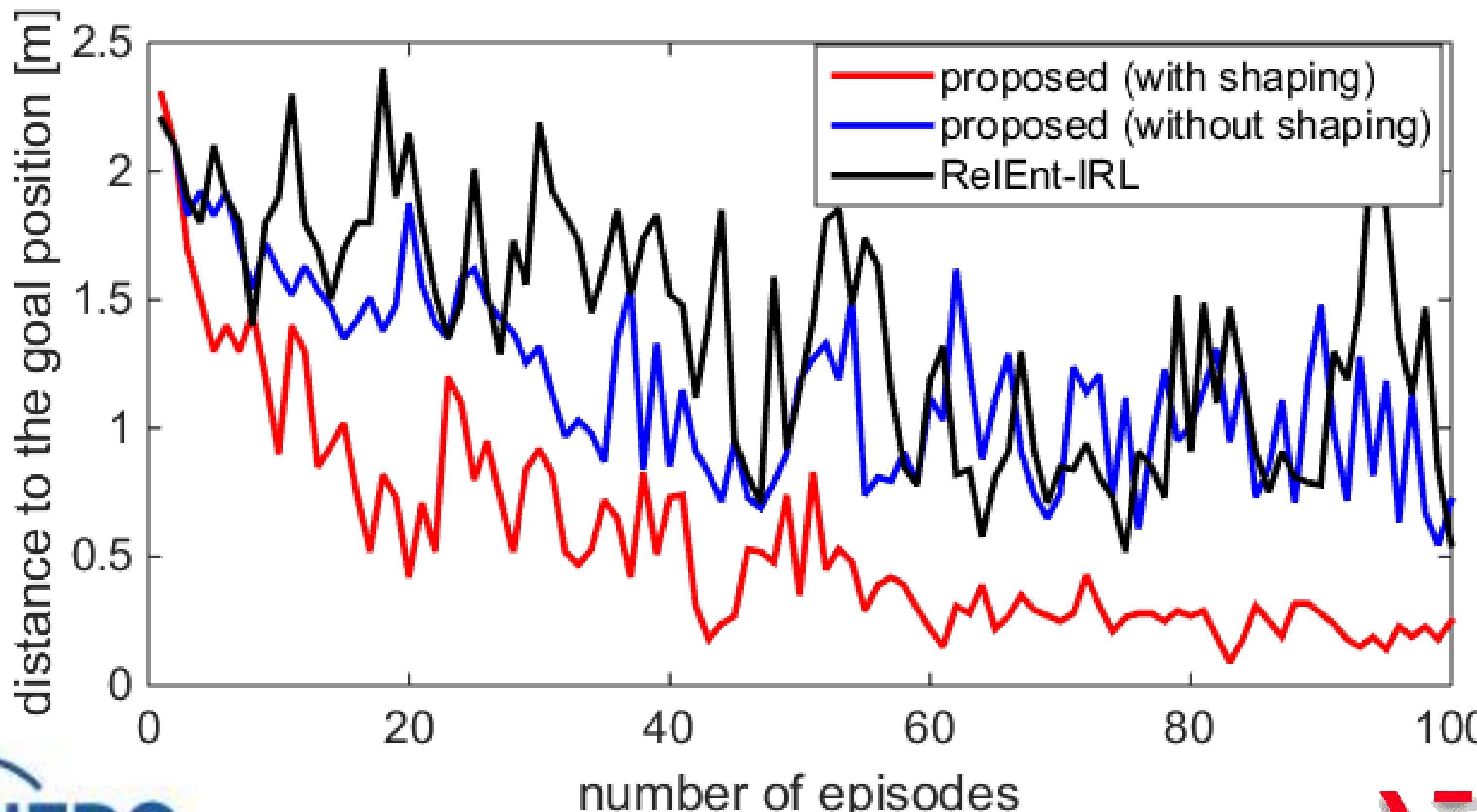
Estimated weights

- There were no significant differences



Acceleration by shaping

- original reward: $q(\boldsymbol{x})$
- shaping reward: $q(\boldsymbol{x}) + \gamma V(\boldsymbol{y}) - V(\boldsymbol{x})$



Conclusion

- We propose the inverse reinforcement learning algorithm based on density ratio estimation
- Our methods successfully recovered the policies from observed behaviors as compared with previous methods
- The estimated value function can be used as a potential function for accelerating the learning process

Acknowledgements

- This work was supported by MEXT/JSPS KAKENHI Grant Number 24500249 and Grant-in-Aid for Scientific Research on Innovative Areas: Prediction and Decision Making (24120527 and 26120727).
- This is based on results obtained from a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO).