A Reinforcement Learning Method to Improve the Sweeping Efficiency for an Agent

The University of Electro-Communications
Naoto Ohsaka
Tokyo National College of Technology
Daisuke Kitakoshi and Masato Suzuki

2011/11/09

Outline

- Background Information
- Sweeping Task Planning Problem
- LCRS
- Computer Simulations
- Conclusion

Background Information

Background Information (1)

- Domestic cleaning robot
  - One of general domestic robots
  - It wipes dust off a floor or removes trash while moving about a room
- Concept of typical cleaning robot such as Roomba
  - Sweeping the same field repeatedly

Background Information (2)

- In this research, we focus on
  - sweeping whole field as quickly as possible
- This research proposes a reinforcement learning method
  - *Learning for Controlling Redundant Sweeping* (*LCRS*)
- Objective of this research
  - To evaluate the basic characteristics and sweeping performance of LCRS
    - We carry out several computer simulations

Sweeping Task Planning Problem

Sweeping task planning problem

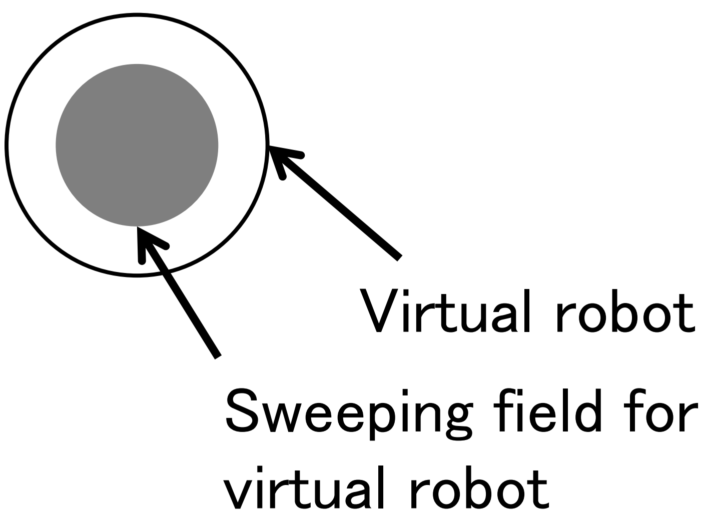
- We define *sweeping task planning* (*STP*) *problem* as
  - problem in which an agent aims to maximize an area of swept field as quickly as possible

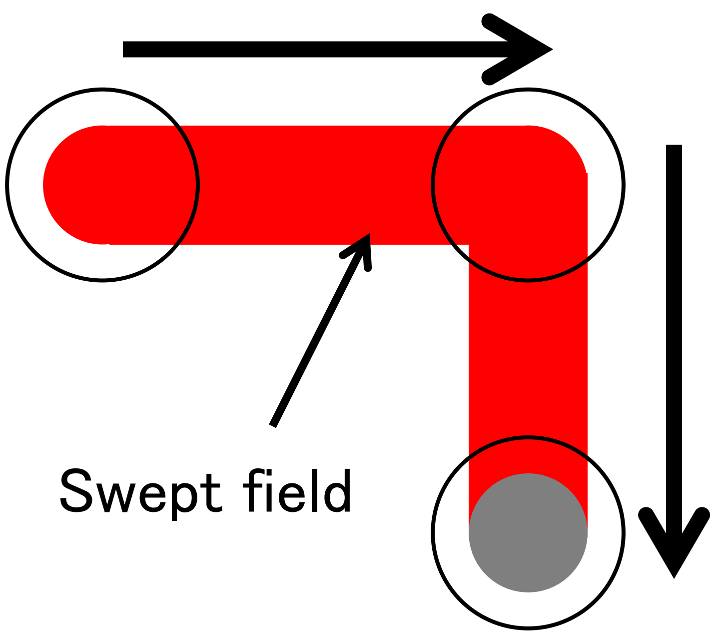
- Hardware performances are constant
  - Agent's moving speed
  - Type, performance and the number of sensors

Working field

- Virtual robot has a *sweeping field* (gray field)
- A moving trajectory of virtual robot's sweeping field is regarded as swept

    (red field)





LCRS

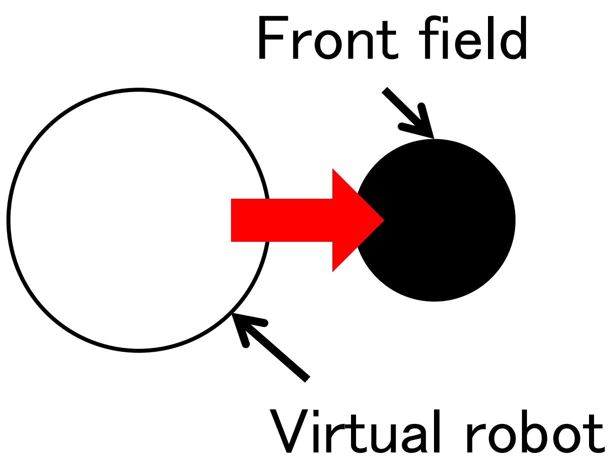
Main components of LCRS

1. Extension of definition of agent's state and setting of reward
   - Specialized for improving sweeping efficiency
2. Macro action
   - To help agent to detect different state
3. Reinforcement Learning
   - Learning method for agent's behavior

Definition of agent's state (1)

- Agent's state is defined as a combination of

1. Distances from the agent to obstacles
2. Binary parameter α_t
   - expressing whether front field is
     - already swept (=1)
     - or not swept (=0)



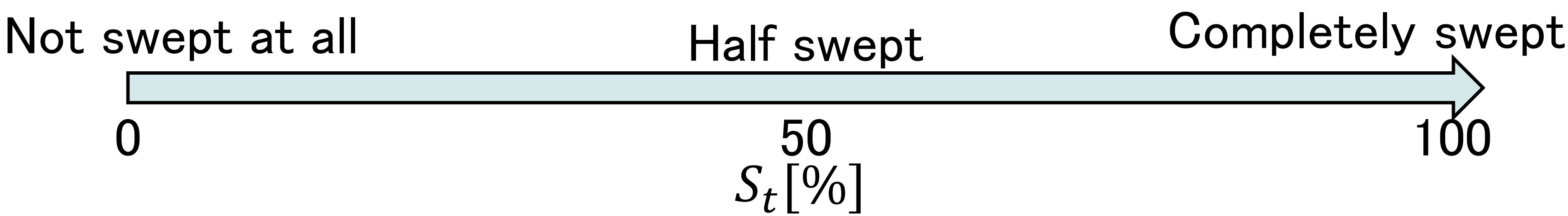
- Agent's front field
  - If the agent go forward, it sweeps it's front field (black field)

Sweeping rate

- We define *sweeping rate* S_t at time t as

$$S_t = \frac{(\text{Area of swept field at } t)}{(\text{Area of working field})} \times 100[\%]$$

- S_t expresses a rate for area of swept field to area of working field



- We define *sweeping rate in agent's front field* S'_t at time as

$$S'_t = \frac{(\text{Area of swept front field at } t)}{(\text{Area of front field})} \times 100[\%]$$

- S'_t expresses the rate for agent's front field
- S'_t is used to determine the value of α_t

Definition of agent's state (2)

- The value of α_t is determined as
  - 1 ($S'_t \geq \theta$)
  - 0 (otherwise)

- Threshold θ has impact on agent's "activeness"
  - The value of θ is small → Rough behavior
    - An agent does not care even if a certain amount of field is not swept yet
  - The value of θ is large → Sensitive behavior
    - An agent considers having to sweep completely if even a bit of its front field is not swept

Setting of reward

- A reward which an agent acquires at time t is

$$r_t = \frac{\Delta S_t - \Delta S_{max}/2}{\Delta S_{max}/2}$$

$\Delta S_t = S_t - S_{t-1}$

ΔS_{max} : maximum possible increment of S_t during a unit time step

- An area of field where the agent swept in a unit time step is normalized to $[-1, 1]$

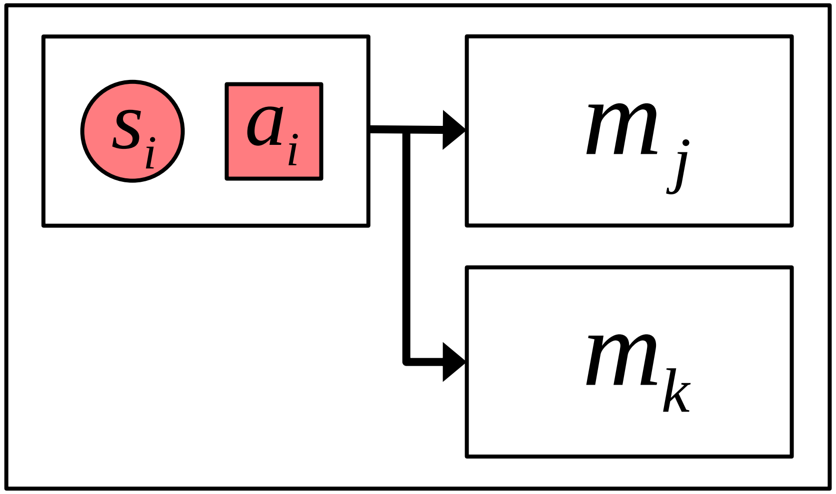
Definition of macro action

- Macro action
  - Helps agent to detect different state
    - It enables agent to have a memory
  - Defined as 2-tuple

$$m = \big((s, a), M\big)$$

  - (s, a) : rule
  - M : a set of macro actions

Macro action m_i

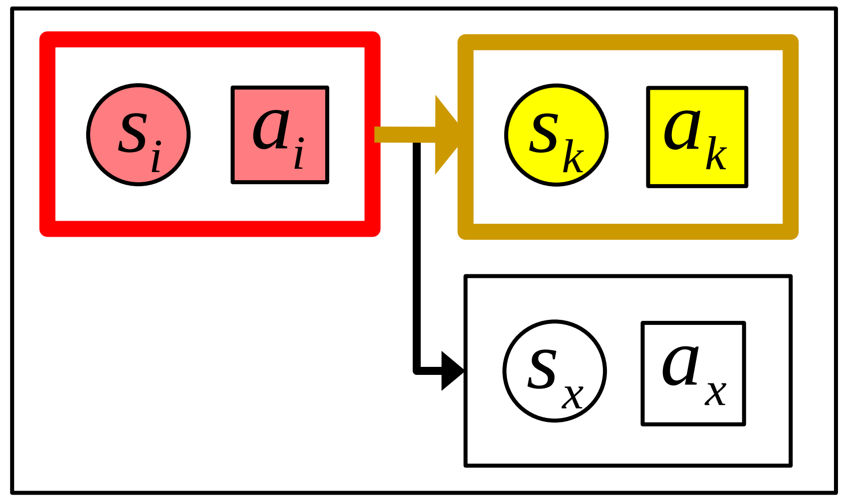


- Examples of macro action
  - $m_i = \big((s_i, a_i), \{m_j, m_k\}\big)$
  - $m_j = \big((s_j, a_j), \emptyset\big)$

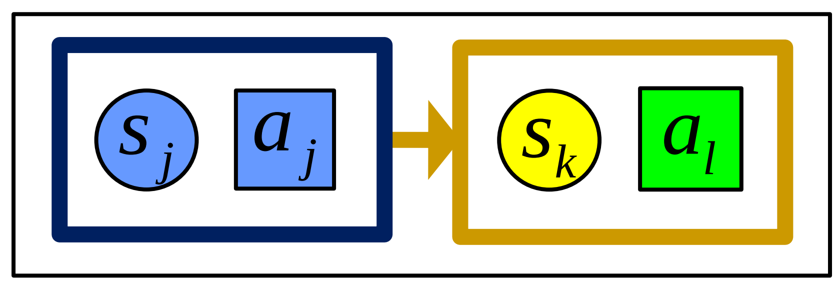
Application of macro action (1)

1. At time step 1
   1. Observe s_i or s_j
   2. Select $m1$ or $m2$
   3. Output a_i or a_j
2. At time step 2
   1. Observe s_k
   2. Output a_k or a_l

Macro action $m1$



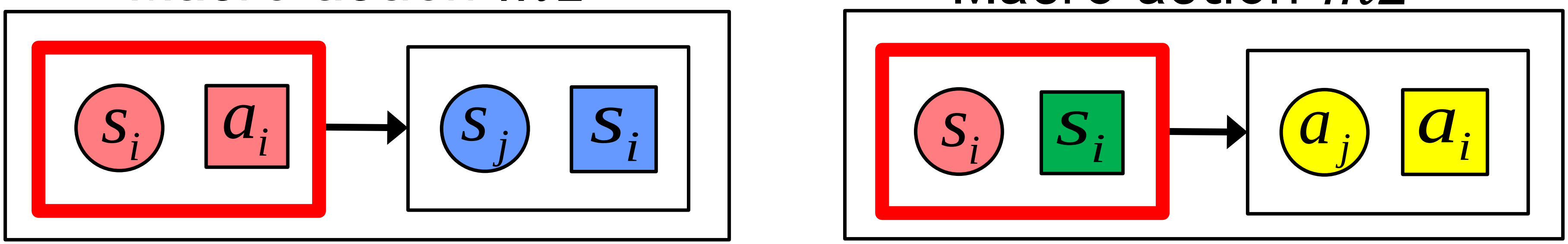
Macro action $m2$



Application of macro action (2)

- Observe s_i
- States of $m1$ and $m2$ are the same
- But, actions of $m1$ and $m2$ are different

Macro action $m1$ Macro action $m2$



- In such situation
- An agent selects macro action stochastically using the value of macro action
- $Q(m)$ indicates the value of macro action m
  - It expresses effectiveness of m

Reinforcement learning

- A kind of machine learning to adapt to an environment based on the concept of dynamic programming
- The learning method in LCRS : sarsa
- $Q(m)$ is updated as follows

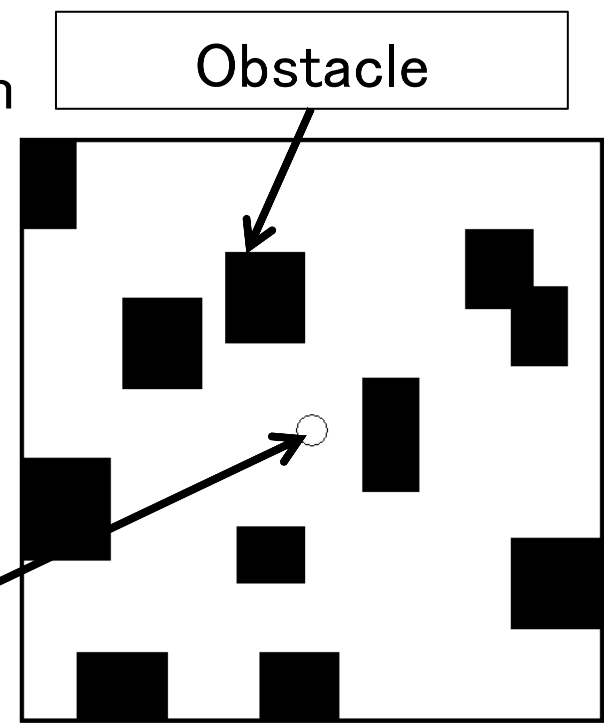
- $Q(m) \leftarrow (1-\alpha)Q(m) + \alpha\left[\sum_{k=1}^{T} \gamma^{k-1} r_{t+k} + \gamma^{T} Q(m')\right]$
- α : Learning rate
- γ : Discount rate
- m : macro action selected at t
- m' : macro action selected at $t+T$

Computer simulations

Evaluating basic characteristics and performance of LCRS

Settings of simulation environment

- Simulation environment
  - Surrounded by perimeter walls
  - it has 11 random-sized obstacles
- In the beginning of sweeping task,
  - an agent is placed at the initial position
  - all swept field is reset to unswept
- When S_t reaches 80%, one episode ends



Settings of agent (virtual robot)

- An agent can detect
  1. Distances from obstacles in 3 directions (front, left and right)
  2. Whether its front field is swept or not (binary parameter α_t)
- The total number of its states = 54
- It takes one of 3 actions
  - Go forward, rotate (45[deg]) left and right

Settings of parameters

Parameters

Parameter	Value	Parameter	Value
α	0.1	C_1	4
γ	0.9	C_2	4
τ	0.5	β_1	1
N	100	β_2	1

- 1 trial is defined as 200 episodes
- In each experiment,
  - We run 10 trials and average the results

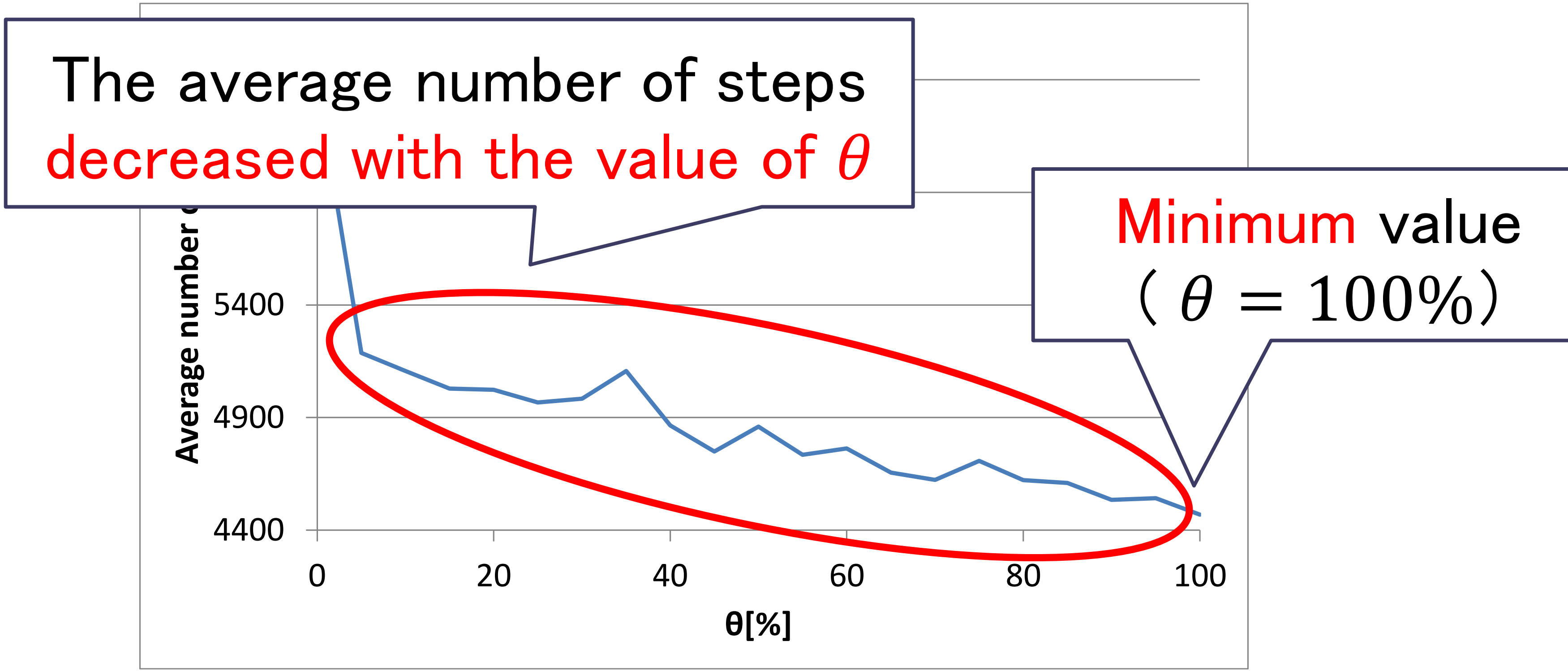
Experiment 1

- We discuss the value of θ
  - With small θ
    - Rough behavior
  - With large θ
    - Sensitive behavior
  - θ is fixed as $0,5,\cdots,100\%$

- We pay attention to episode 100 to 199
  - In which the agent was able to acquire a stable policy

Result and discussions (1)

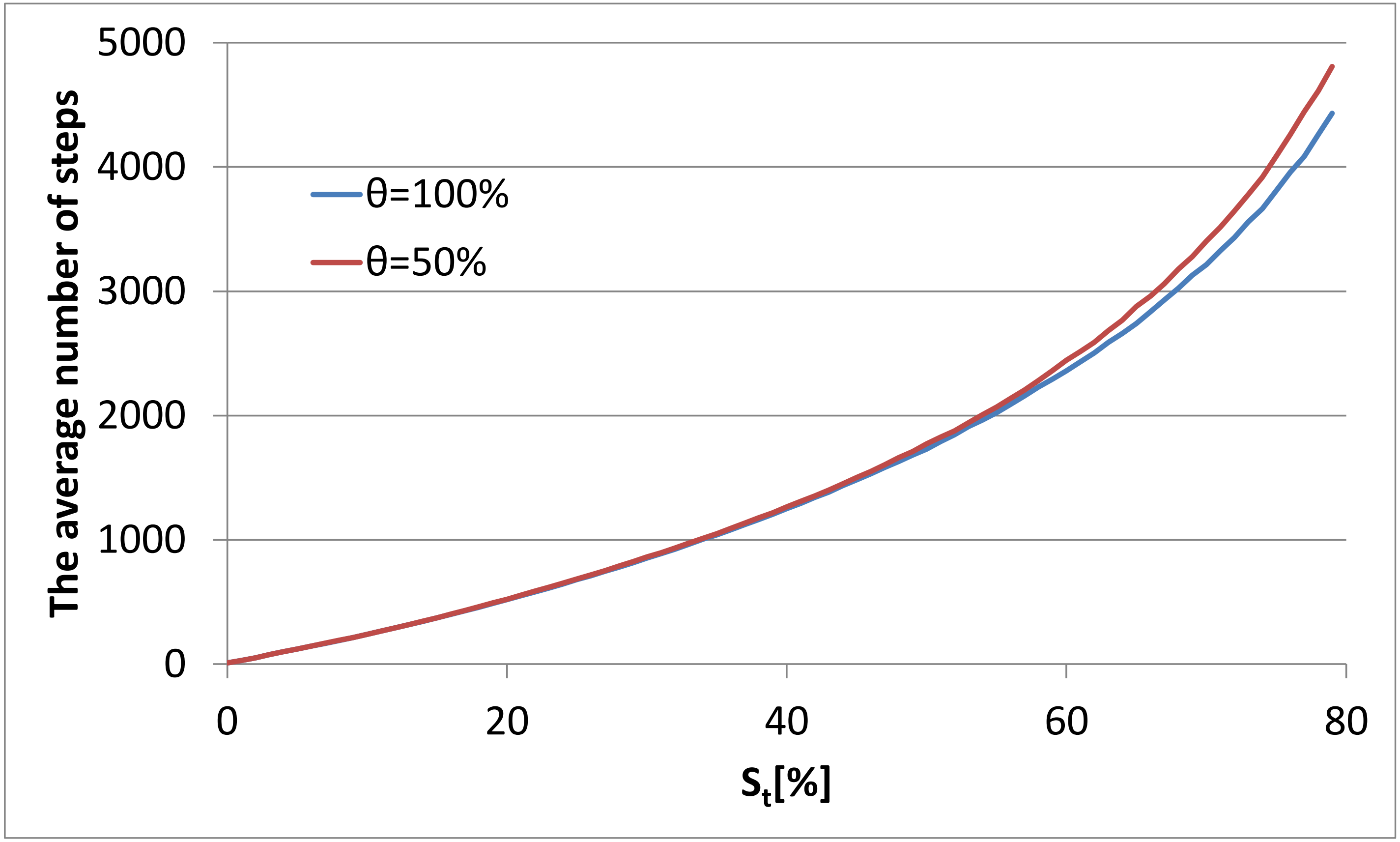
Relationships between the average number of steps for the task completion and θ



We discuss a difference between agent's behavioral characteristics with large and small θ

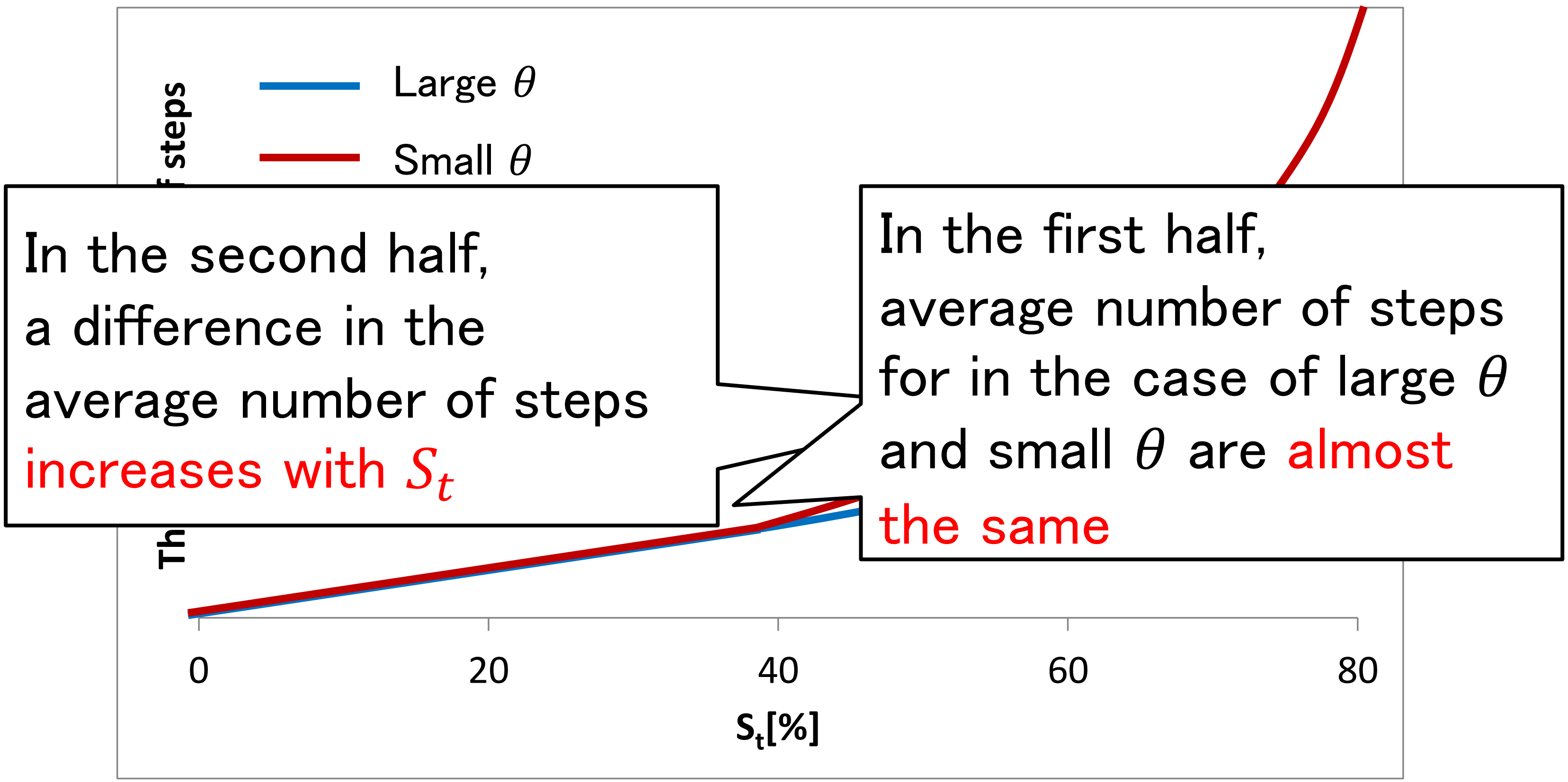
Result and discussions (2)

- The average number of steps for S_t in the case of $\theta = 100\%$ and 50%



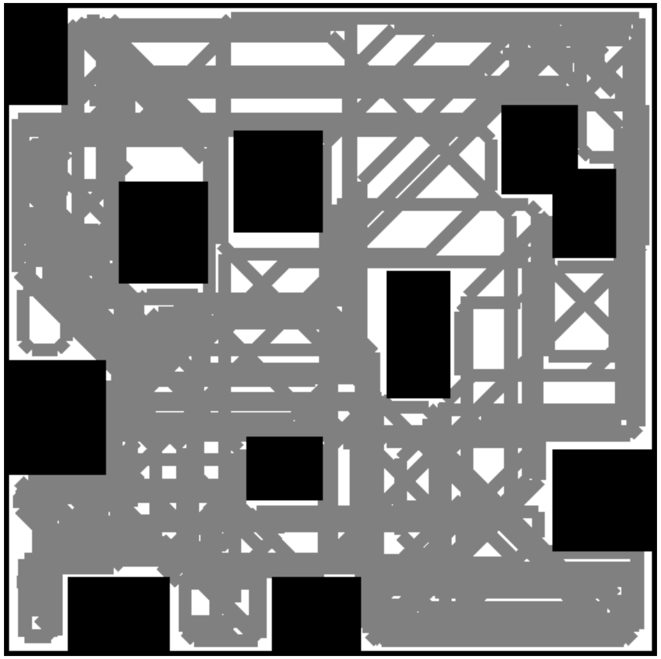
Result and discussions (3)

- The average number of steps for S_t in the case of large θ and small θ



Result and discussions (4)

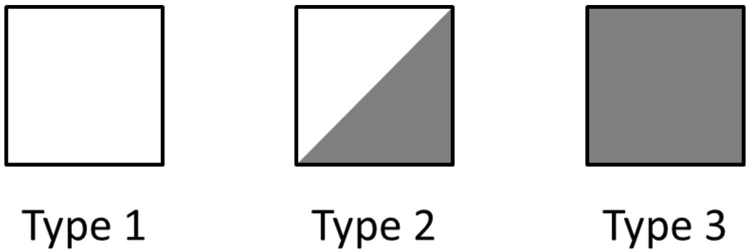
- In the first half,
  - there is no difference in the average number of steps
  - θ does not impact on the sweeping efficiency
- In the second half,
  - the difference is clear
  - This is due to fragmentation and scattering of unswept field



Working field when the task completes in a typical episode

Result and discussions (5)

- Front field can be classified into 3 types



- Type 1 is regarded as unswept
- Type 3 is regarded as swept
- Type 2 is
  - Regarded as unswept (if θ takes large value)
  - Regarded as swept (if θ takes small value)

- The agent with small θ can hardly detect unswept field
- The agent with large θ can have more opportunities to detect fragmented small unswept field

- θ is fixed as 100% hereafter

Experiment 2

- We investigate components having impact on the sweeping efficiency
- We prepare 2 variants
  - sarsa(s) ··· sweeping rate is introduced into definition of state
  - sarsa(m) ··· sarsa with macro action

Sweeping algorithms

Sweeping algorithm	Definition of state	Macro action
sarsa(s)	√	X
sarsa(m)	X	√
sarsa	X	X

√ : same as LCRS X : simplified

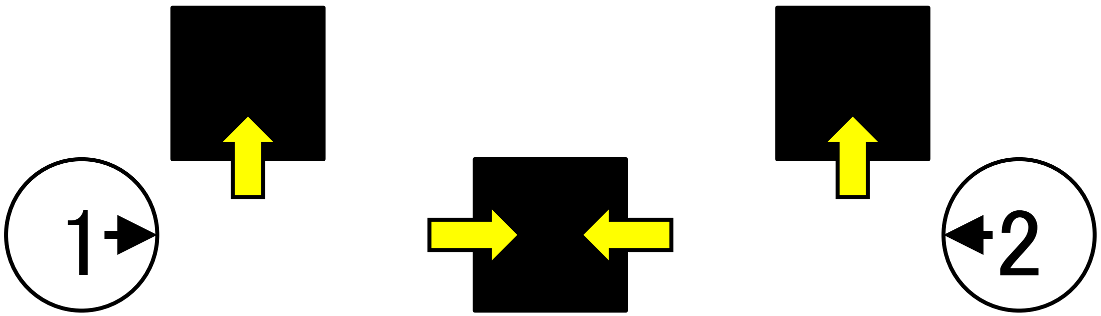
Result and discussions (1)

- Average number of steps for 4 sweeping algorithms

Sweeping algorithm	Average number of steps
LCRS	4640
sarsa(s)	6000
sarsa(m)	6360
sarsa	7320

- We discuss a difference between the behavior of agent with sarsa and sarsa(m)

Result and discussions (2)



time	VR1's state	VR2's state
1	None	None
2	Left	Right
3	Front	Front

- At time step 3
  - In the case of sarsa,
    - 2 agents cannot distinguish each other's state
    - Because current states are seems to be the same (actually different)
  - In the case of sarsa(m) (an agent has a memory),
    - 2 agents can distinguish
    - Because states at time 2 are different from each other

Conclusion

Conclusion

- We proposed a reinforcement learning method LCRS to improve the sweeping efficiency of an agent

- The empirical results indicate
  - LCRS agent behaves effectively

Conclusion

- 3 Future projects
  - Investigation of the impact of settings of environment on sweeping efficiency
  - Multi-agent environment
  - Real-world environment
    - Due to errors in the robot's motor control and sensor noise
    - Agent may not detect whether its front field has already swept
    - Introducing stochastic method into LCRS to overcome this problem