1. INTRODUCTION

Many systems working with humans, e.g., humanoid robot [1], environmental intelligence, and portable information assistant [2], have been studied in recent year. These systems need the autonomous learning function for the adaptations to various users' requests and the changes of environment. One of approaches to the autonomous learning is Reinforcement Learning (RL) [3] and a study on shooting robot is an example of its early applications to practical problems [4]. This study shows that RL can be used for autonomous systems. RL, however, cannot be applied easily to every practical problem since RL needs much time for learning. So other methods instead of RL are necessary for practical applications. One of the methods is Model-based RL [4], which has an inner model of environment and improves the learning efficiency by using that model. That is, if there is the knowledge of a task, then an agent uses it for learning. Although even model-based RL has some problem about the error of an inner model, there are few studies on that.

This study gives careful consideration on the effects of the model error and proposes to solve the problem above a new approach, Model Error based Forward Planning Reinforcement Learning (ME-FPRL) [5]. ME-FPRL controls the amount of learning based on errors of inner models and can make the learning efficiency high. And in this paper, ME-FPRL is applied to the pursuit of a target by a robot camera. The results of this application show that ME-FPRL learns more efficiently than Usual RL and Model-based RL.

2. RL AND MODEL-BASED RL

2.1 Framework of RL and Model-based RL

Fig.1 shows the framework of RL. Using RL based on Monte Carlo Methods and Dynamic Programming, the agent acquires the appropriate actions [3].

Fig.1 Framework of RL

In the RL method, the agent takes an action to environment and then environment changes. Environment gives reward to an agent according to the action. The agent observes the state of environment, and then takes a new action on environment in order to get more reward. The agent learns the appropriate behavior by the both two interactions, with real environment and with the inner model. The learning by the interaction with environment is called direct learning, and by the interaction with the inner model is called indirect learning. If the agent’s model is correct, that is, the same as environment, the agent can get a good performance by indirect learning because the model generates good experiences. On the other hand, if the inner model has some errors, or has some
differences from real environment, the agent can not learn the behavior correctly and the learning efficiency becomes low. It is found that Model-based RL has the problem on the error of the inner model.

![Diagram of Model-based RL](image)

**Fig.2 Framework of Model-based RL**

### 3. ME-FPRL

ME-FPRL is proposed in order to solve the problem about the error in Model-based RL [5]. Fig.3 shows the algorithm of ME-FPRL.

(0) Initialize all variable and \( a \leftarrow policy(s,Q) \)

Do forever:

1. \( s \leftarrow \) current state
2. Execute action \( a \)
   - Observe resultant state \( s' \) and reward \( r \)
3. \( a' \leftarrow policy(s,Q) \)
4. \( Q(s,a) \leftarrow Q(s,a) + \alpha[\hat{r} + \gamma Q(s',a') - Q(s,a)] \)
5. Update \( Model(s,a) \)
6. \( ModelAcy(s,a) \)
   - \((1 - \rho) \cdot ModelAcy(s,a) + \rho \cdot \gamma^{sd(s',i')}\)
7. Repeat \( N_p \) times
   - (7-1) \( \hat{s} \leftarrow s \cdot \hat{a} \leftarrow a \cdot ComAcy \leftarrow 1 \)
   - (7-2) Repeat \( N_i \) times:
     - (7-2-1) \( s', r \leftarrow Model(s,a) \cdot \hat{s}' \leftarrow s' \cdot \hat{a} \leftarrow r \)
     - (7-2-2) \( \hat{a}' \leftarrow policy(s,Q) \)
     - (7-2-3) ComAcy \leftarrow ComAcy \cdot ModelAcy(\hat{s},\hat{a}) \)
     - (7-2-4) \( \hat{s} \leftarrow \hat{s}' \cdot \hat{a} \leftarrow \hat{a} \)
   - (7-3) while stack has some item:
     - (7-3-1) Pop (\( \hat{s},\hat{a},\hat{r},\hat{s}',\hat{a}' \), ComAcy)
     - (7-3-2) \( Q(\hat{s},\hat{a}) \leftarrow Q(\hat{s},\hat{a}) + \gamma^{nd(\hat{s}',\hat{a}')} 
    \)
   - \( ComAcy \cdot \partial h/N_{plan} [\hat{r} + \gamma Q(\hat{s}',\hat{a}') - Q(\hat{s},\hat{a})] \)
8. \( s \leftarrow \hat{s} \cdot a \leftarrow a \)
   - (\( \hat{s},\hat{a},\hat{r},\hat{s}',\hat{a}',ComAcy \) are temporary variable)

**Fig.3 Algorithm of ME-FPRL**

This algorithm is based on Model-Based RL with the trajectory [3]. This algorithm controls the amount of indirect learning by estimating the current error. Therefore, learning is robust against the error [5].

In this algorithm, \( s \) stands for a current state, \( a \) a current selected action, and \( Q \) an action-value function. The state dissimilarity between \( S_g \) and \( S_h \), \( sds(s_g,s_h) \), is defined as the number of necessary steps for giving \( s_g = s_h \). \( \alpha(0 \leq \alpha \leq 1) \) is a step size parameter, \( \gamma(0 \leq \gamma \leq 1) \) is a discount rate and \( \rho \) is a tradeoff parameter. For example, when \( \rho \) is about 1, the latest state dissimilarity \( sds(s',\hat{s}') \) affects \( ModelAcy \) very well, and when \( \rho \) is about 0, the previous state dissimilarity affects \( ModelAcy \). \( N_p \), \( N_i \) and \( h/N_{plan} \) is parameter of indirect learning. \( N_i \) is a length of the trajectory, \( N_p \) is the number of indirect learning, and \( h/N_{plan} \) is a parameter for changing ratio about direct or indirect learning.

### 4. APPLICATION TO PURSUING TARGET TASK

#### 4.1 Continuous State Spaces

In real world, some tasks such as control of robots are very complex and the state space is very large. Therefore, an agent needs much time to learn behavior. This problem is called “curse of dimensionality”. To cope with this, there is a method converting the original state space into the feature space, which has enough size to be able to solve the task. This converting method is called function approximation. One of the methods of applying function approximation is the liner method. In the liner method, action-value function \( Q \) is expressed as Eq. (1).

\[
Q(s,a) = \sum_{i=1}^{I} \theta(i,a)\phi(i,s)
\]

Where \( \theta \) is a parameter vector and \( \theta(i,a) \) is scholar. \( \phi \) is a feature vector and \( \phi(i,s) (0 \leq \phi(i,s) \leq 1) \) is a scholar and \( I \) is the size of feature space. Since \( Q(s,a) \) is a summation of products of \( \theta(i,a) \) and \( \phi(i,s) \), \( Q(s,a) \) is a linear function of \( \phi(i,s) \). So this method is liner. Action-value function is changed by the general gradient-descent update [3]. If feature vector is given by linear approximation methods, the action-value function converges to a local optimum.

In this study, ME-FPRL is extended by the liner method in order to apply this algorithm to some practical tasks. Fig.4 shows the algorithm of ME-FPRL with function approximation. \( \delta \) is temporary variable for TD-Error[3].
4.2 Pursuing Target Task

In the control of a robot system, pursuing target task is a basic problem and many control methods are developed for it. However, a control policy adjusted to a robot cannot be applied to another robot because of different dynamics among robots. That is, many times adjustment of control policy is needed. In order to solve this problem, it is important to make up the differences of each dynamics by learning [6]. In this paper, ME-FPRL is applied to the pursuing target task. A four-legged robot shown in Fig.5 (AIBO ERS-7M3) [7] learns the control policy to pursue a target shown in Fig.6 using its CCD Camera that works pan/tilt directions. The target moves like a pendulum by a.

4.3 Action and Reward

The agent chooses one action out of five actions: “Turn to top” “Turn to bottom”, “Turn to right” “Turn to left” “Stop”. Movement of each action except “Stop” turns its head by 5 degrees.

And the agent gets rewards as follows: 10 points; catching a target near the center of the camera, -10 points; catching a target out of the center of the camera, -20 points; missing a target.

4.4 Learning Agents

In order to confirm the validity of ME-FPRL, the learning efficiencies of the control policy with RL, model-based RL and ME-FPRL are compared with each other.

An agents’ action is selected by Gibbs Sampler [3]. Eq. (2) shows Gibbs distribution, where $\tau$ is a time constant.

\[
\text{Gibbs}(s, a) = \frac{\exp(Q(s, a)/\tau)}{\sum_{b, a} \exp(Q(s, b)/\tau)}
\] (2)

The agents approximate state spaces by the CMAC [3] that is a liner approximation method. In the CMAC, state spaces expressed by some tiles. The number of tile is $\text{Tiling}$. The parameters for the agents are as follows: $\text{Tiling} = 50$, $\alpha = 0.2/\text{Tiling}$, $\gamma = 0.8$, $\rho = 0.3$, $N_p = 8$, $N_f = 3$, $N_{\text{plan}} = N_p/3$, $h = 0.9$, $\tau = 0.4$, $\theta$ and $\text{ModelAcy}$ is initialized by 0.

4.5 Inner Model

The agents with the model-based method have an inner model. In this task, dynamics of the target is constructed by the multi-layered neural networks [8].
Fig. 7 shows the inner model of the target. This model has two neural networks. These networks are switched by the direction of target’s movement. That is, one network deals with a rotation of right and another network deals with a rotation of left. These networks are learned by the error back propagation [8].

4.6 Experiment and Its Result

In this experiment, an episode is defined as follows. One episode finishes when the robot catches a target in the center of a camera for 40 steps or when the robot loses the target completely. The robot performs 100 times tasks. In the first half, 50 times, the target moves to only right side, and in the latter half, 50 times, the target moves to both right and left sides. That is, the movement of the target is changed at the 50th time.

Fig. 8 shows experimental results with RL, Fig. 9 shows those with ME-FPRL and Fig. 10 shows those with model-based RL. It is found that ME-FPRL is faster than RL in the early stage and is more robust than RL.

5. CONCLUSIONS

In this paper, the effective learning algorithm ME-FPRL is proposed and applied to the pursuit of a target. Application results show that the proposed ME-FPRL is more efficient than a RL or Model-based RL. As a result, ME-FPRL is found to be able to apply to practical tasks.

Our future work is constructing more an efficient learning system by using human or other agent’s advice and communication.

REFERENCES

[1] Takakatsu Isozumi: Development of Humanoid Robot, Kawata Giho, Vol. 22, 2003
[2] Shintaro Anzui, Satoshi Hukuda, Masahiro Hamasaki, Ikki Ohmukai, Hideaki Takeda, Takahira Yamaguti: A Recommendation System for Personal Digital Assistance with Ontologies, The 19th Annual Conference of the Japanese Society for Artificial Intelligence, 2005
[3] Richard S. Sutton, Andrew G. Garto: Reinforcement Learning: An Introduction, MIT Press, Cambridge, MA, 1998
[4] M. Asada, S. Noda, S. Tawaratsumida, and K. Hosoda: Vision-Based Behavior Acquisition for a Shooting Robot by Using A Reinforcement Learning, Proc. of IAPR/IEEE Workshop on Visual Behaviors-1994, pp. 112-118, 1994
[5] Yoshiyuki Tajima, Takehisa Onisawa: Model-based Reinforcement Learning with Model Error, International Symposium on Computational Intelligence and Industrial Applications(ISCIIA), pp127-134, Guangzhou, China, 2006
[6] David Muse, Kevin Burn, Stefan Wermtor: Reinforcement Learning for Platform-Independent Visual Robot Control, IEEE World Congress on Computational Intelligence, pp.4766-4773, 2006
[7] AIBO ERS-7M3, http://www.jp.aibo.com
[8] Ryohey Nakano: Basic Theory of Neural Network Information Processing, Surikogakusya, 2005