Acquisition of Visual Motion Guided Behaviors

Minoru Asada and Takayuki Nakamura
Dept. of Mechanical Eng. for Computer-Controlled Machinery,
Osaka University, Suita 565 JAPAN
Phone:+81-6-879-7349, Fax:+81-6-879-7348.
e-mail: nakamura@robotics.ccm.eng.osaka-u.ac.jp

Abstract

Sensor and motor systems are not separable for autonomous agents to accomplish tasks in a dynamic environment. However, almost computer vision researchers have not investigated the relationship between the low level motion representation and the visual information. In this paper, we propose a method by which a mobile robot learns sensorimotor apparatus and target pursuit behavior without any calibrations nor any 3-D reconstructions. Real-time visual tracking routines are used to detect an optical flow of the environment which is correlated with motor commands to obtain the sensorimotor apparatus, and to obtain target pursuit behavior with obstacle avoidance. Target pursuit behavior and obstacle avoiding one are obtained by Q-learning, one of the most widely used reinforcement learning methods. Computer simulation and real experiments are given to show the validity of the method.

1 Introduction

Sensor and motor systems are not separable for autonomous agents to accomplish tasks in a dynamic environment. Horridge [1] advocated that motion is essential for perception in living systems such as bees. In physiological psychology, Held and Hein [2] have shown that self-produced movement with its concurrent visual feedback is necessary for the development of visually-guided behavior. They tested two groups of kittens, one was allowed to vary with its locomotor movements while equivalent stimulation of the other resulted from passive motion. The former could normally perform several visually guided behaviors but the latter failed. These suggest that perception and behavior are tightly coupled in autonomous agents that perform tasks.

In computer vision area, so-called "active vision" has been considered as a representative form of this coupling since Aloimonos proposed it [3] as a method that converts the ill-posed vision problems into the well-posed ones. Hence, many researchers have been using so-called active vision systems in order to reconstruct 3-D information such as depth and shape from a sequence of 2-D images given the motion information of the observer or capability of controlling the observer motion. This implies that people in computer vision expect the motor system to output the ideal motion descriptions in terms of 3-D translation and rotation or to be able to precisely control any motions of its end-effector such as camera or hand.

On the other hand, people in robot control expect the sensor system to output the accurate data such as 3-D locations of feature points or lines extracted from camera images to move its end-effector to the destination or along the desired trajectory with a specified velocity. Since it is misunderstood that the reconstructed 3-D information from 2-D images or encoders of joint angles is the most powerful representation as a general interface between two disciplines [4], perception and motion control have been independently studied. According to Brooks [5], this is just a deliberative approach, and therefore, seems time-consuming, difficult to obtain sufficient results, and brittle. As many researchers replied [6, 7, 8] to a dialogue [4], reconstruction of the 3-D environment is not always necessary nor optimally encoded for the task.

Purposive vision does not consider vision in isolation, but as a part of complex system that interacts in specific ways with world [6]. However, very few have tried to investigate the relationship between motor commands and visual information [9] because people in computer vision have been assuming the motor system as an ideal one and have not cared about it any more (ex. [10, 11]). In order to realize tight coupling between sensor and motor systems, we should consider the relationship between the low level representation of motion (motor commands to actuators) and the visual information, and develop a learning capability to abstract the low level representation into a form suitable for task accomplishment. Pierce and Kuipers [12] proposed a method to learn sensorimotor apparatus of a mobile robot with sonar sensor system. They gathered sonar patterns in terms of each robot action and obtained primitive motions which can produce a sonar pattern caused by every robot action. Since the fluctuations in a sonar pattern of each motion due to environmental factors such as obstacles, walls, and so on are smoothed out by averaging a collection of patterns, the primitive motions seem difficult to be used to detect obstacles in the environment.

In this paper, we investigate the relationship between the motor commands and the visual information due to the motor actions by extending the method [12] to optical flow patterns on the floor, and then apply Q-learning, one of the most widely used reinforcement learning methods, to target pursuit task in a dynamic environment. An agent has a PWS (Power Wheeled Steering) system as a motor system and visual tracking routines [13] which are used to learn sensorimotor apparatus first, and then to learn target pursuit with obstacle detection and avoidance behaviors. Although Aloimonos [6] pointed out two things which determine agent behaviors: the characteristics of the system itself including mechanics and control, and the task it needs to accomplish, one more thing should be added, an "environment" in which an agent performs a task at hand. Here, we assume that the agent does not know the physical meaning of optical flow nor the effect of its motion in the environment. Instead, we provide the environments for the agent to learn behaviors to accomplish these tasks: target pursuit in the environments without and with obstacles.

In the next section, we propose a new method to represent visual scenes called "motion sketch" for a one-eyed mobile robot to learn several behaviors such as obstacle avoidance and target pursuit. Then, we give a method for acquisition of sensorimotor apparatus, and describe a reinforcement learning method to obtain target pursuit behavior avoiding obstacles with computer simulation. Finally, real experimental results are shown.

2 Motion Sketch and Task

Fig.1 shows a basic idea of "motion sketch" which is a method to represent visual scenes for a one-eyed mobile robot to learn several behaviors. The basic components of the motion sketch are visual motion cues detected by several visual tracking routines of which visual behaviors are determined by individual tasks, and are tightly coupled with the motor behaviors which are obtained by Q-learning. The image area to be covered by these tracking routines are specified or automatically detected depending on the individual tasks, and the cooperative behaviors between tracking routines are performed for task accomplishment. The most fundamental task is to obtain the relationship between the visual motions and robot motor commands. To do that, the visual tracking routines are scattered over the whole image and an optical flow due to instantaneous robot motion is detected. In this case, the tracking routines are fixed to the individual image positions. In the task of obstacle detection and avoidance, the candidates for obstacles are first detected by comparing the motion vector with that of non-obstacle (ground plane) region, and then the detected region is tracked by multiple templates each of which tracks the inside of the moving obstacle region. For the target pursuit task, the multiple templates are initialized and every template looks for the target to realize the stable tracking. The motor behavior is a set of motor commands obtained by Q-learning, one of the robot learning methods, based on detected motion cues and given task. The sizes and positions of the target and

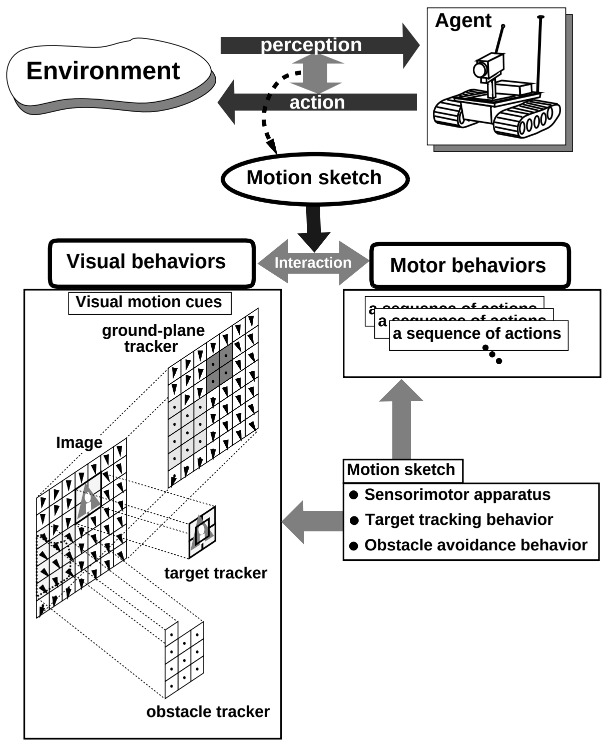


Fig.1: **Motion sketch**

the detected obstacle are used as components of a state vector. Visual and motor behaviors work in parallel in the image and compose a layered architecture. The visual behavior for monitoring robot motion (detecting the optical flow on the ground plane on which the robot lies) is the lowest and might be subsumed in part due to occlusion by other visual and motor behaviors for obstacle detection/avoidance and target pursuits which might occlude each other.

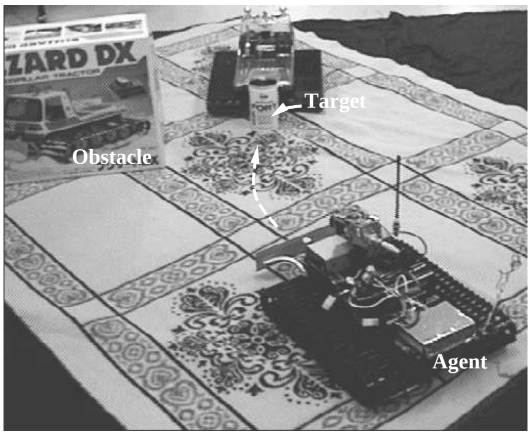


Fig.2: **The task is to pursue a target avoiding obstacles.**

Fig.2 shows an environment in which an agent tries to pursue a target, or to avoid obstacles. The robot can select one action among an action set which controls robot motion, and perceive the changes of the environment due to its motion by using multiple visual tracking routines.

The behavior acquisition scheme consists of the following four stages:

stage 1 Obtaining the fundamental relationship between visual and robot motions by correlating motion commands and flow patterns on the floor with very few obstacles.

stage 2 Learning target pursuit behavior by tracking a target.

stage 3 Detection of obstacles and learning an avoidance behavior.

stage 4 Coordination of the target pursuit and obstacle avoidance behaviors.

3 Robot System

3.1 A configuration of the system

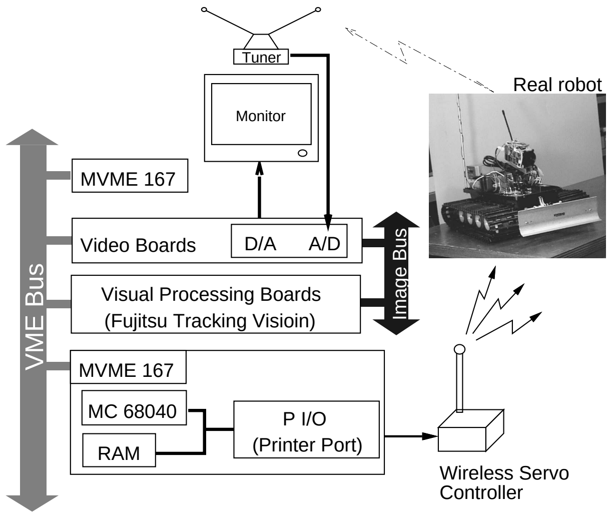


Fig.3: **A configuration of the real system.**

Fig.3 shows a configuration of the real mobile robot system. We have constructed the radio control system of the robot[14]. The image processing and the vehicle control system are operated by VxWorks OS on MVME167(MC68040 CPU) computer which is connected with host Sun workstations via Ether net. The image taken by a TV camera mounted on the robot is transmitted to a UHF receiver and subsampled by scan-line convertor(Sony Corp.). Then, the video signal is sent to a Fujitsu tracking module. The tracking module has a function of block correlation to track some pre-memorized patterns and can detect motion vectors in real time. Then, the tracking module feeds the flow vectors at each regions to the host CPU(MC68040). The host CPU calculates the averaged motion vector field (see Section 3.1 for more detail) and stores them. The host Sun workstation calculates the fundamental relationship bewteen visual motions and motor commands off-line. **Fig.4** shows a picture of the real robot with a TV camera (Sony camera module) and a video transmitter.

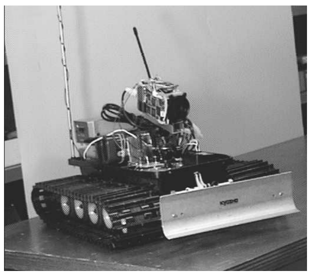


Fig.4: **A picture of the radio-controlled vehicle.**

3.2 PWS system

The robot has a Power Wheeled Steering (hereafter P-WS) system driven by two motors into each of which we can send a motor command, independently. The velocities of translation v and rotation ω of the robot can be represented by two motor commands, more correctly two angular velocities ω_l and ω_r. The following equation shows the relationship between (v, ω) and (ω_r, ω_l) to be sent to the right and left motors.

$$\begin{pmatrix} v \\ \omega \end{pmatrix} = \begin{pmatrix} \frac{R_r}{2} & \frac{R_l}{2} \\ \frac{R_r}{T} & -\frac{R_l}{T} \end{pmatrix} \begin{pmatrix} \omega_r \\ \omega_l \end{pmatrix} \qquad (1)$$

where R_r, R_l, and T denote the radii of the right and left wheels, and the distance between two wheels, respectively.

In our experiment, we quantized $\omega_{l(r)}$ into five levels which correspond to quick forward, slow forward, stop, slow backward, and quick backwrad, respectively. Totally, we have 25 actions. Note that the robot does not even know any physical meanings of these actions.

3.3 Multiple visual tracking routines

To detect changes due to robot motion, we use real-time visual tracking routines which can track about 140 windows (each window consists of 8 by 8 pixcels) in real-time (video rate) by using a motion estimation processor (MEP) [13]. Searching area is 16 by 16 pixcels and MEP output the location of each window where the following matching error (SAD: sum of absolute difference) is minimum.

$$D[i,j] = \sum_{k=0}^{K-1} \sum_{l=0}^{L-1} |R[k,l] - M[i+k, j+l]|,$$
$$i, j : 0 \leq i, j \leq 15,$$

where $R[x,y]$, $M[x,y]$, and $D[x,y]$ denote a reference block, a matching block, and an array of SAD, respectively. The visual tracking routines are used to obtain an optical flow of the floor, to track a target specified by human operator, and to detect and avoid obstacles. We detect a motion vector in an image by applying the block matching process with the reference block ($t = t_i$) and the search window images ($t = t_{i+1}$) continuously. Thus, we can obtain the optical flow vectors in the image any time.

4 Obtaining sensorimotor apparatus

Here, we show how to learn sensorimotor apparatus by correlating sensor information with motion command.

We assume that the robot is given no knowledge of the structure of its sensory system nor of the effects. We extend the method [12] as follows:

- Instead of sonar information (3-D range information), we use an optical flow of the floor which can be obtained by multiple visual tracking routines.
- In order to remove fluctuations of a flow pattern of each action due to the environmental factors, we set up the environment with very few obstacles. In averageing flow patterns, we used the least median of squares method [15] to remove the outliers due to noise or small obstacles.

We can compress the visual motion patterns by the obtained fundamental relationship in order to include the ego-motion information in the internal state space of the agent.

We place $49 (7 \times 7)$ visual tracking routines to detect changes in the whole image. Examples of averaged optical flows from a real environment are shown in **Fig.5**.

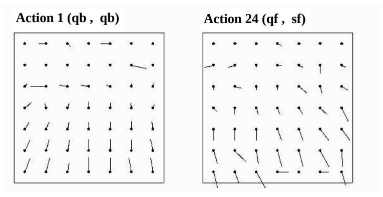


Fig.5: **Examples of averaged optical flows from a real environment.**

Using the set of the averaged optical flows for all actions, we acquire mapping between the optical flows and the actions. This is done by analyzing the space of averaged optical flows that robot is capable of producing. We want to find a basis for this space, i.e., a set of representative motion vector fields from which all the motion vector fields may be produced by a linear combination. We can obtain the representative motion vectors by using Principal Component Analysis that may be performed using a technique called Singular Value Decomposition(hereafter SVD).

The sample values of $\boldsymbol{p}_i$ (the averaged optical flow) corresponding to the actioin $i(\tau_{li}, \tau_{ri})$ are organized as the rows of matrix $\boldsymbol{P}$. There are 25 rows each of which have 98 components (49 vectors per one box). The SVD of $\boldsymbol{P}$ is

$$P_{m \times n} = U_{m \times n} S_{n \times n} E^T_{n \times n}, \qquad (2)$$

where S is a diagonal matrix whose elements are the singular values of $\boldsymbol{P}$ and the rows of $\boldsymbol{E}^T$ are the desired orthonormal basis vectors. Here, $m = 25$, the number of averaged optical flows and $n = 98$, the number of components in each averaged optical flow (two for each local visual tracking routine). $\boldsymbol{U}$ is an orthogonal matrix in terms of the row. The averaged optical flow $\boldsymbol{p}_i$ can be described by a linear combination of the vectors in $\boldsymbol{E}^T$ from the quation (2), using the K principal components:

$$p_i \approx \sum_{k=1}^{K} u_{ik} s_k e_k^T \tag{3}$$

Thus, we have found a basis set (the row vectors of $\boldsymbol{E}^T$) for the space of averaged optical flow. In fact, we obtain 26 principal components by caluculating SVD for the $\boldsymbol{P}$ which consist of 25 flow patterns. **Fig.6** shows the singular values with respect to the principal components.

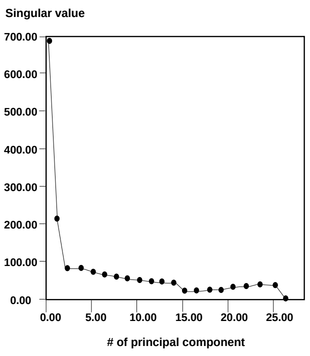


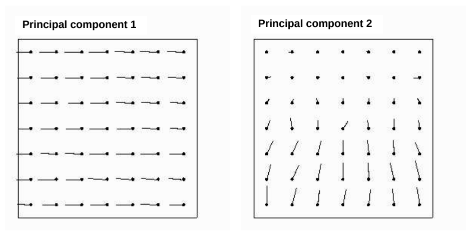
Fig.6: **Singular values of principal components.**

We select two important basis vectors among them which have larger singular value than others. The averaged optical flow may be approximated by throwing away all but the important basis vectors. Thus, for example, vector $\boldsymbol{p}_i$ may be approximated by

$$p_i \approx u_{i1} s_1 e_1^T + u_{i2} s_2 e_2^T$$

if we keep only the first two components. The first two principal components obtained in the real environment are shown in **Fig.7**. Evidently, the first (a) corresponds to a pure rotation and the second (b) to a pure backward motion.

Next step is to make a relation between the possible actions $\boldsymbol{p}_i$ by representing each of them in terms of the coeficient $a_k^i = u_{ik} s_k$ in the action space which consists of two principal components. The relation



(a) (b)

Fig.7: **The first two principal components.**

between possible actions of the real robot are shown in **Fig.8**, where the number indicates the number of the action i $\quad (i = 1 \sim 25)$.

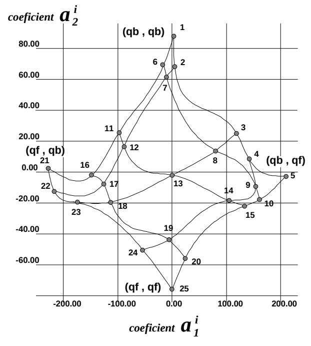


Fig.8: **The relation between the possible actions of the real robot.**

5 Behavior acquisition based on visual motion cues

5.1 Basics of Reinforcement Learning

Reinforcement learning agents improve their performance on tasks using reward and punishment received from their environment. They are distiguished from supervised learning agents in that they have no "teacher" that tells the agent the correct response to a situation when an agent responds poorly. An agent's only feedback indicating its performance on the task at hand is a scalar reward value. One step Q-learning[16] has attracted much attention as an implementation of reinforcement learning because it is derived from dynamic programing[17]. The following is a simple version of the 1-step Q-learning algorithm we used here.

Initialization: $Q \leftarrow$ a set of initial values for the action-value function (e.g., all zeros).
Repeat forever:

1. $s \in \mathbf{S} \leftarrow$ the current state
2. Select an action $a \in \mathbf{A}$ that is usually consistent with the policy f but occasionally an alternate.
3. Execute action a, and let s' and r be the next state and the reward received, respectively.
4. Update $Q(s,a)$:

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha(r + \gamma \max_{a' \in A} Q(s', a')). \tag{4}$$

5. Update the policy f:

$$f(s) \leftarrow a \quad \text{such that} \quad Q(s,a) = \max_{b \in \boldsymbol{A}} Q(s,b) \tag{5}$$

5.2 Target tracking behavior acquisition

We use visual tracking routines in order to pursue a target specified by an human operator and obtain the information about the target in the image such as its position and size which are used in the Q-learning algorithm for acquisition of target pursuit behavior.

5.2.1 State of visual tracking routine

Our visual tracking routine has the following visual functions.

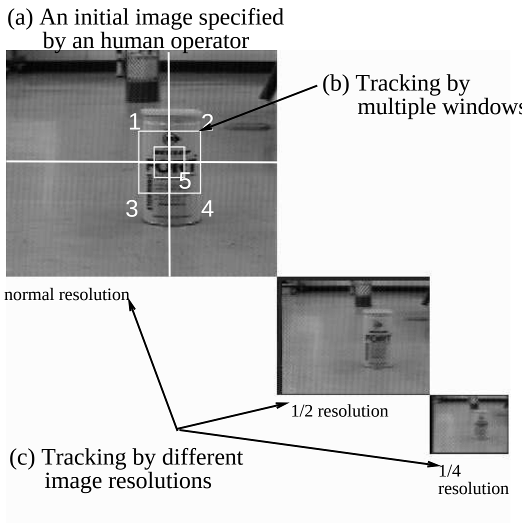

(a) An initial image specified by an human operator
(b) Tracking by multiple windows
1
2
3
4
5
normal resolution
1/2 resolution
1/4 resolution
(c) Tracking by different image resolutions

Fig.9: **Visual functions of tracking routine.**

1. Tracking by multiple windows with different image resolutions: A target is tracked by an object tracker which consists of 5 visual tracking routines fixed together as shown in **Fig.9**. Even if the pattern of the target is fluctuated by occlusion or the vibration of the robot body, the object tracker can continue to track target by the effect of multiple visual tracking routines. Further, we prepare three kinds of resolutions(a normal, a half and a quarter resolution). Even if the the pattern of the target become large or small, the object tracker can continue to track by changing the image resolution and the search area for the block matching. **Fig.9** shows an inital image specified by an human operator (a), tracking by multiple windows (b), and tracking by different image resolutions (c), respectively.
2. When the target detection fails, search-whole-image routine is called in order to detect the target again.

We define the state of the target in the image based on the target position and the target size (three levels) obtained by a visual tracking routine.

5.2.2 State and action spaces in Q-learning

In order to apply the Q-learning scheme to target pursuit task, we define a number of sets and parameters. The state of the target $\boldsymbol{S}$ in the image is quantized into 9 sub-states, combinations of three positions (left, center, and right) and three sizes (large (near), medium, and small (far)). Similarly, changes in terms of target position and size in the image are quantized into 9 sub-states, combination of three states in terms of position changes (move left, no move and move right) and three states in terms of size changes (enlarge, no change, shrink). We add two lost situations (target is lost into the left side or the right side) in the state space. Futhermore, we add the action (totally 25 actions) just taken on observing the current situation into the state space. Totally, we have 92×25 states in the set $\boldsymbol{S}$. We have 25 actions in the action set $\boldsymbol{A}$. We assign a reward value 1 when the robot touched the target or 0 otherwise. A discounting factor γ is used to control to what degree rewards in the distant future affect the total value of a policy. In our case, we set the value a slightly less than 1 ($\gamma = 0.9$).

5.2.3 Target tracking with no obstacles

We performed the computer simulation with the following specifications (the unit is an arbitrary-scaled length). The target is a ball of which diameter is 6. The robot is 16 wide and 20 long. The camera is mounted on the robot and looks toward the floor (15 degree tilt). Its visual angle is 30 degrees. These and other parameters such as friction between the floor and the crawler and bounding factor between the robot and the ball are chosen to simulate the real world. The target is moving randomly. **Fig.10** shows the pursuit behavior aquired by Q-learning. The robot initially looked around because the target was not observed from its initial position. Once the robot found the target, the robot ran to the target quickly. Finally, the robot reached the target.

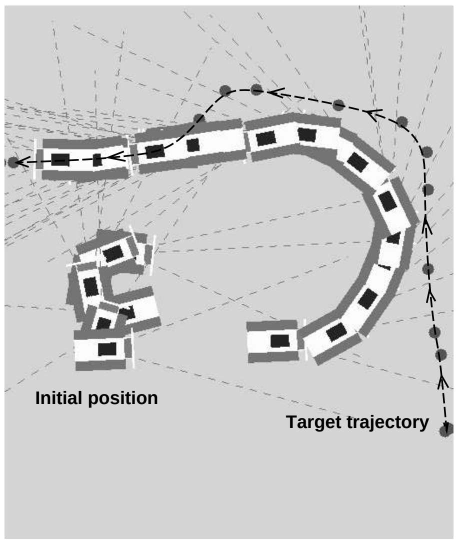


Fig.10: The target pursuit behavior in simulation.

5.3 Acquisition of obstacles avoidance behavior

(a) Detection and tracking of obstacles by flow differences

We know the flow pattern $\boldsymbol{p}_i$ corresponding to the action i in the environment without any obstacles. Therefore, it makes motion segmentation easily. Motion segmentation is done by comparing the flow pattern $\boldsymbol{p}_i$ with the flow pattern $\boldsymbol{p}_i^{obs}$ which is obtained in the environment with obstacles.

The area in the $\boldsymbol{p}_i^{obs}$ which differs from the same area in the $\boldsymbol{p}_i$ is detected as the one in which the obstacle candidates are projected. This information (position and size in the image) is used to obtain the obstacle tracking behavior. After obtacle detection,

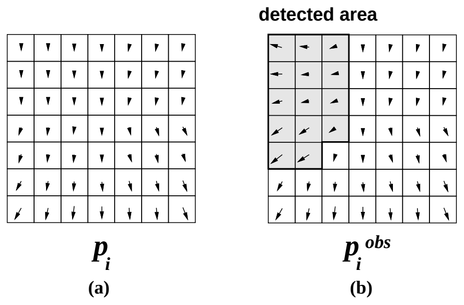


Fig.11: Obstacle Detection.

the visual tracking routines are set up at the position where the obstacle candidates are detected and the region is tracked until the region disappears from the image.

Fig.12 show the result of obstacle detection in the real environment, where (a) displays the environment in which the robot detected candidates for obstacles (see (b)). The circles in the image indicate the obstacle candidate regions.

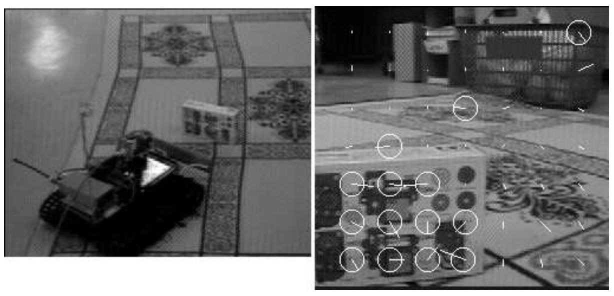
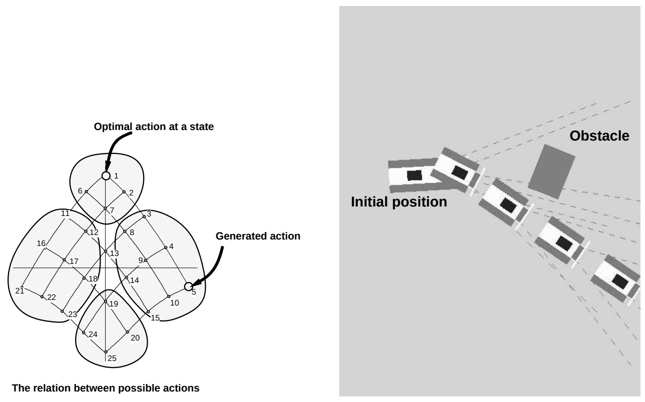


Fig.12: A picture of the environment with an obstacle and obstacle detection.

(b) Learning an obstacle avoidance behavior

Learning an obstacle avoidance consists of two stages. First, the obstacle tracking behavior is learned by the same manner as in learning the target pursuit behavior. Next, the obstacle avoidance behavior is generated by using the relation between the possible actions and the obstacle tracking behavior as follows: (1) the relationship between the possible actions is divided into four categories by clustering the action space represented by the coefficients (a_1^i, a_2^i) (See **Fig.13**(a)), (2) the obstacle tracking behavior is mapped on the relationship, and the category C^twhich includes the obstacle tracking action is found, (3) the obstacle avoidance actions are selected among the categories except for C^t. More correctly, the obstacle avoidance action is obtained by finding the action having the smallest action-value function with respect to the obstacle tracking behavior among the categories except for C^t. **Fig.13**(b) shows the obstacle avoidance behavior acquired by the proposed scheme in computer simulation.


Optimal action at a state
Generated action
The relation between possible actions
Obstacle
Initial position

(a) Generation of obstacle avoidance behavior

(b) Obstacle avoidance behavior in simulation.

Fig.13: **Obstacle avoidance behavior**

5.4 Coordination of obstacle avoidance and target tracking behaviors

We consider coordinations in which the previously learned behaviors are combined: switching action value functions according to the situation. We used the subsumption architecture [5] to combine previously learned behaviors. The switching condition is to select target pursuit behavior unless only the obstacle can be observed very largely. **Fig.14** shows the results of the coordinated behavior acquired by our method.

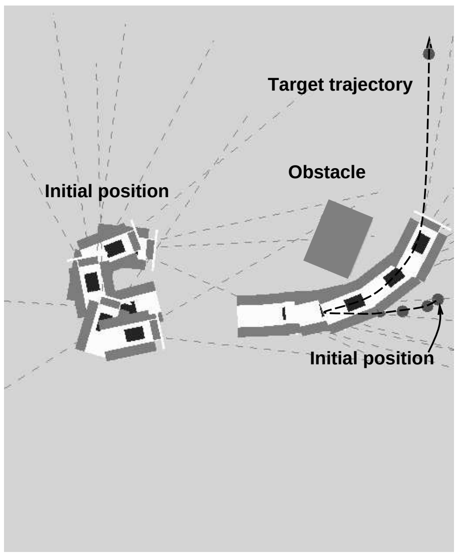


Fig.14: **Coordinated behavior in simulatoin.**

6 Future Work

As one of the method for representing the visual scenes for situated agents to behave adequately against the external world, we proposed "motion sketch" which is independent of scene components and tightly coupled with motor commands. Now, we are planning to develop a new program which tightly connects MaxVideo 200 and Fujitsu Tracking module to speed up and add other higher level visual functions.

References

[1] G. A. Horridge. "The evolution of visual processing and the construction of seeing systems". In *Proc. of Royal Soc. London B230*, pages 279–292, 1987.

[2] R. Held and A. Hein. "Movement-produced stimulation in the development of visually guided behaviors". *Journal of Comparative and Physiological Psycology*, 56:5:872–876, 1963.

[3] Y. Aloimonos, I. Weiss, and A. Bandyopadhyay. "Active vision". In *Proc. of first ICCV*, pages 35–54, 1987.

[4] M. Tarr and M. Black. "Dialogue: A computational and evolutionary persepctive on the role of representation in vision". *CVGIP: Image Understanding*, 60:1:65–73, 1994.

[5] R. A. Brooks. "A robust layered control system for a mobile robot". *IEEE J. Robotics and Automation*, RA-2:14–23, 1986.

[6] Y. Aloimonos. "Reply: What i have learned". *CVGIP: Image Understanding*, 60:1:74–85, 1994.

[7] G. sandini and E. Grosso. "Reply: Why purposive vision". *CVGIP: Image Understanding*, 60:1:109–112, 1994.

[8] S. Edelman. “Reply: Representatin without reconstruction”. *CVGIP: Image Understanding*, 60:1:92–94, 1994.

[9] G. Sandini. “Vision during action”. In Y. Aloimonos, editor, *Active Perception*, chapter 4. Lawrence Erlbaum Associates, Publishers, 1993.

[10] Y. Aloimonos, E. Rivlin, and L. Huang. “Designing visual systems: Pruposive navigation”. In Y. Aloimonos, editor, *Active Perception*, chapter 2. Lawrence Erlbaum Associates, Publishers, 1993.

[11] C. Fermüller. “Navigatinal preliminaries”. In Y. Aloimonos, editor, *Active Perception*, chapter 3. Lawrence Erlbaum Associates, Publishers, 1993.

[12] D. Pierce and B. Kuipers. “Learning to explore and build maps’. In *Proc. of AAAI’94*, pages 1264–1271, 1994.

[13] H. Inoue, T. Tachikawa, and M. Inaba. “Robot vision system with a correlation chip for real-time tracking, optical flow and depth map generation”. In *Proc. IEEE Int’l Conf. on Robotics and Automation*, pages 1621–1626, 1992.

[14] M. Asada, S. Noda, S. Tawaratsumida, and K. Hosoda. “Vision-based behavior acquisition for a shooting robot by using a reinforcement learning”. In *Proc. of IAPR / IEEE Workshop on Visual Behaviors-1994*, pages 112–118, 1994.

[15] P. Meer, D. Mintz, A. Rosenfeld, and D. Y. Kim. “Robust regression methods for computer vision: A review”. *IJCV*, 6:1:59–70, 1990.

[16] C. J. C. H. Watkins. *Learning from delayed rewards”*. PhD thesis, King’s College, University of Cambridge, May 1989.

[17] R. Bellman. *Dynamic Programming.* Princeton University Press, Princeton, NJ, 1957.