Stereo Sketch: Stereo Vision-Based Target Reaching Behavior Acquisition with Occlusion Detection and Avoidance

Takayuki Nakamura and Minoru Asada
Dept. of Mech. Eng. for Computer-Controlled Machinery
Osaka University, 2-1, Yamadaoka, Suita, Osaka 565, Japan
nakamura@robotics.ccm.eng.osaka-u.ac.jp

Abstract

*In this paper, we proposed a method by which a stereo vision-based mobile robot learns to reach a target by detecting and avoiding occlusions. We call the internal representation that describes the learned behavior "***stereo sketch***." First, an input scene is segmented into homogeneous regions by the enhanced ISODATA algorithm with MDL principle in terms of image coordinates and disparity information obtained from the fast stereo matcher based on the coarse-to-fine control method. Then, in terms of the segmented regions including the target area and their occlusion status identified during the stereo and motion disparity estimation process, we construct a state space for a reinforcement learning method to obtain target reaching behavior. As a result, the robot can avoid obstacles without explicitly describing them. We give the computer simulation results and real robot implementation to show the validity of our method.*

1 Introduction

Realization of autonomous agents that organize their own internal structure in order to take actions for their goal achievement is the ultimate goal of Robotics and AI. That is, the autonomous agents have to learn through the interaction with the environment via their perception and actions. There have been a variety of approaches to analyze the relationship between perception and action.

In computer vision area, so-called "purposive active vision paradigm" [1, 2, 3] has been considered as a representative form of this coupling since Aloimonos [4] proposed it as a method that converts ill-posed vision problems into well-posed ones. Purposive vision does not consider vision in isolation but as a part of complex system that interacts with world in specific ways [1]. However, many researchers have been using so-called active vision systems in order to reconstruct 3-D information from a sequence of 2-D images given the motion information of the observer or capability of controlling the observer motion. Furthermore, very few have tried to investigate the relationship between motor commands and visual information. Sandini et al. [5] built a robot which tried to balance the flow seen from two cameras facing laterally. The robot motion are determined by the control law designed by the programmer. Therefore, their robot cannot improve the performance of its behavior through the interaction between the robot and its environment. Kuniyoshi et al. [6] constructed an active tracking system with vergence controlled stereo cameras supported by the Extended Zero Disparity Filter in order to realize robust stereo tracking and target pursuit behaviors of the robot. However, their robot requires another image processing module in order to avoid obstacles. Recently, Huber and Kortenkamp [7] used a real-time stereo vision system [8] to pursue moving agents while still performing obstacle avoidance. Since their stereo matching is based on edges extracted by a Laplacian-Gaussian filter, they have the following drawbacks: The system likely looses the target because tracking module is easily attracted by higher texture areas than the current target one, and therefore they cannot cope with any occlusions of the target area by other objects. Since the system try to keep the target in 3-D space at a fixed distance, they cannot cope with changes in scale of the target image (a group of edges) due to motions of the target and/or the robot, nor reach the target.

In robot learning area, many researchers have only shown computer simulations, and only a few real robot applications which are simple and less dynamic [9, 10] are reported. In these works, proximity sensors such as bumper and sonar are used. Therefore, their tasks are limited to local, reflexive, and simple ones. The use of vision in the reinforcement learning is very rare due to its high costs of sensing and processing.

In this paper, we propose a stereo vision-based behavior learning method. As a real-time stereo matching method, we use a block correlation of image intensity [11], by which the capability of stereo and motion disparity estimation are much improved than only edge-based approach. By adopting the coarse-to-fine control of stereo [12, 13] and motion disparity estimation techniques, we can cope with changes in scale due to target and/or the robot motions. Further, we apply a reinforcement learning method to obtain a reaching behavior for the environmental adaptation using a well-defined state space consisting of occlusion status identified during the stereo and motion disparity estimation process.

The remainder of this article is structured as fol-

lows: In the next section, we describe basic ideas of *stereo sketch*. Then, we give explanations of visual behaviors, the method of learning, and state space definition. Finally, we give computer simulations, real robot implementation results, and concluding remarks.

2 Stereo Sketch

The interaction between an agent and its environment can be seen as a cyclical process in which the environment generates an input (perception) to the agent and the agent generates an output (action) to the environment. If such an interaction can be formalized, the agent would be expected to carry out actions that are appropriate to individual situations. "Motion sketch" we have proposed in [14] is one of such formalizations of interactions by which a one-eyed vision-based learning agent with real-time visual tracking routines behaves adequately against its environment to accomplish a variety of tasks.

Here, we add one more camera on the robot and realize a real-time stereo-vision system with the same tracking routines.

We prepare image processing procedures for estimating stereo disparity and for region segmentation and tracking based on the estimated disparity map as **visual behaviors**. The stereo and motion disparity information not only improves the tracking performance but also provides useful information about occlusion and disocclusion in terms of which we construct a state space.

We prepare a set of motor commands which are organized into a sequence of motor commands through the learning process. We call such sequences **motor behaviors**. We assume that the robot initially has no knowledge about physical meaning of individual motor commands.

The robot obtains the relationship between the situation identified by visual behaviors and the motor behaviors applying a reinforcement learning algorithm, and acquires the target reaching behavior. As a result, the robot realizes obstacle avoidance without explicitly describing them. We call an internal representation obtained by this behavior acquisition method **stereo sketch**. Figure 1 shows the procedure of the proposed method by which **stereo sketch** is obtained as the representation of the tight coupling between **visual behaviors** and **motor behaviors**.

3 Visual Behaviors

. As visual behaviors, we prepare the following image processing procedures using a real-time visual tracker by a simple block correlation based on SAD (Summation of Absolute Difference) [11]:

1. Obtaining a disparity map based on a coarse-to-fine stereo matching procedure from a pair of stereo images.
2. Decomposing an input scene into some region fragments by using the enhanced ISODATA algorithm with MDL principle in terms of image location and disparity.

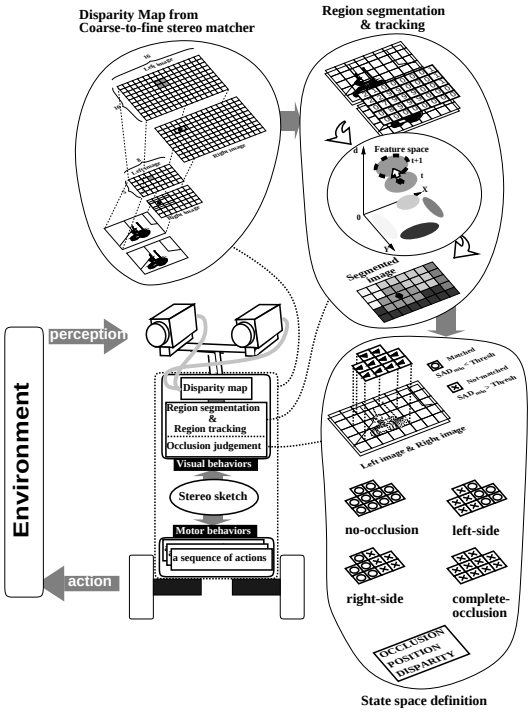


Figure 1: **Stereo sketch**

3. Tracking the area including the target (specified by the human operator or by selecting the region closer to the robot) and its neighbor regions between consecutive frames and identifying the occlusion status of the target region based on stereo and motion disparity information.

3.1 Stereo Matching

A coarse-to-fine stereo matching method [12, 13] is implemented based on block correlation with SAD. In the first stage, each of a coarse image pair is tessellated into 8×5 grids, each grid consists of 16×16 pixels and the search area is 64×24 pixels to cope with rough stereo camera calibration. In the second stage, each of a fine image pair is tessellated into 16×10 grids, each grid consists of 16×16 pixels and the center of the search area (32×24 pixels) for each grid is located at the position where the stereo correspondence in the coarse matching stage is found.

Figure 2 shows the stereo matching result where three pairs of the left and right images are stacked in three rows, respectively. Due to the hardware limitation, we are currently using the middle (coarse image: 128×120 pixels) and large (fine image: 256×240 pixels) scaled image pairs. The final matching result is shown at the right-bottom as a disparity map.

3.2 Region Segmentation and Tracking

In order to reach the target area, the robot always needs to identify the area to be tracked which might be

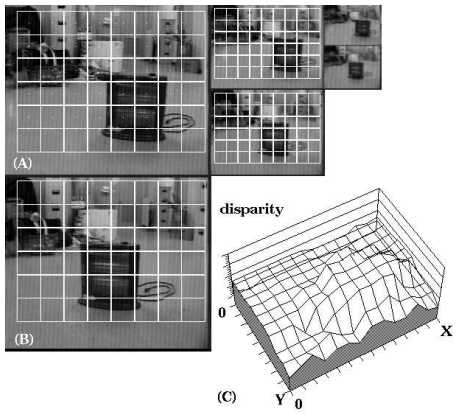


Figure 2: A coarse-to-fine stereo matching

partially or completely occluded by other objects. To cope with this problem, we prepare region segmentation and tracking routines as higher visual behaviors.

3.2.1 Region segmentation

In order to segment the input scene, we form a three-dimensional feature space $\boldsymbol{F}$ consisting of $image\_location(x, y)$, and $disparity(d)$. Every grid in the input scene is grouped into homogeneous regions using a modified ISODATA clustering algorithm. Originally, ISODATA clustering is an iterative clustering algorithm based on heuristics of splitting or merging [15]. Instead of heuristic rules, we search for the compact description grounded by the Minimum Description Length (MDL) principle [16]. See [17] for more detailed procedure of our clustering algorithm.

3.2.2 Region tracking

Based on the motion information and the result of region segmentation, the target region is tracked. Motion vectors are estimated by multiple real-time visual tracking at all 16 $\times$ 10 grids. Our region tracking algorithm is as follows:

1. Let $\boldsymbol{\mu}_j(t_i)$ be a mean vector for the region j at time t_i in the feature space $\boldsymbol{F}$. $\hat{\boldsymbol{\mu}}_j(t_{i+1})$ indicates a linear prediction at t_{i+1}.

2. Using $\hat{\boldsymbol{\mu}}_j(t_{i+1})$ as an initial cluster, we apply the clustering algorithm described in 3.2.1 and then obtain $\boldsymbol{\mu}_j(t_{i+1})$.

    - If the target area is split into two clusters or merged with other cluster, the model of the target area is updated to describe these situations.
    - If the target area is completely occluded, the location is updated by the linear prediction.

On detecting motion vectors in the target region, occlusion status can be detected simultaneously. If the situation that the target area in part are mismatched and its neighbor is correctly matched is observed, the state is judged as an "occlusion state." The judgment of mismatch or match is done according to the minimum SAD value obtained in the block matching process.

4 Behavior Learning

4.1 Basics of Reinforcement Learning

Reinforcement learning agents improve their performances on tasks using rewards and punishments received from their environment. One step Q-learning [18] has attracted much attention as an implementation of reinforcement learning because it is derived from dynamic programming [19]. Here, we briefly review the basics of the Q-learning [20].

We assume that the robot can discriminate the set $\boldsymbol{S}$ of distinct world states, and can take one from the set $\boldsymbol{A}$ of actions on the world. The world is modeled as a Markovian process, making stochastic transitions based on its current state and the action taken by the robot. Let $T(s, a, s')$ be the probability that the world will transit to the next state s' from the current state-action pair (s, a). For each state-action pair (s, a), the *reward* $r(s, a)$ is defined.

Given definitions of the transition probabilities and the reward distribution, we can solve the optimal policy, using methods from dynamic programming [19]. A more interesting case occurs when we wish to simultaneously learn the dynamics of the world and construct the policy. Watkin's Q-learning algorithm gives us an elegant method for doing this. Let $Q^*(s, a)$ be the expected return or *action-value function* for taking action a in a situation s and continuing thereafter with the optimal policy. It can be recursively defined as:

$$Q^*(s,a) = r(s,a) + \gamma \sum_{s' \in S} T(s,a,s') \max_{a' \in A} Q^*(s',a').$$

Because we do not know T and r initially, we construct incremental estimates of the Q values on line. Starting with $Q(s, a)$ at any value (usually 0), every time an action is taken, update the Q value as follows:

$$Q(s,a) \Leftarrow (1-\alpha)Q(s,a) + \alpha \left[r(s,a) + \gamma \max_{a' \in A} Q(s',a') \right],$$

where α is a leaning rate (between 0 and 1) and γ is the discounting factor which controls to what degree rewards in the distant future affect the total value of a policy (between 0 and 1).

4.2 State Space Construction

State space should include necessary and sufficient information to achieve the given goal while it should be compact because Q-learning time can be expected exponential in the size of the state space [21]. Focusing on the target reaching behavior, we construct a state space in terms of occlusion status of the target area and its neighbor obtained by visual behaviors described in 3.

Here, we assume that all obstacles in the environment are convex and exist on the floor, therefore the top of the target cannot be occluded (the bottom

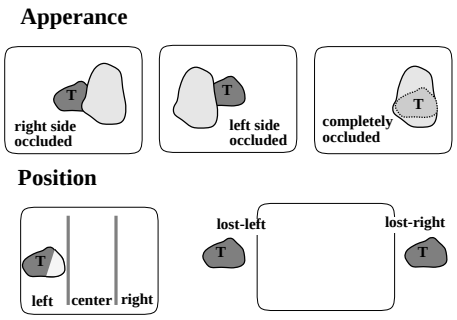


Figure 3: State space construction

might be occluded, but this situation can be categorized into one of three occlusion status). Occlusion status is defined as combinations of target states in the left and right images. For each image, we define the situation as follows: The target state is a triplet of appearance: occlusion (the left side, the right side, completely occluded, no occlusion, or disappearance), position (left, center, or right), and disparity (far, middle, or near). In case of disappearance, we prepare two situations (lost-into-the-right or lost-into-the-left) (see Figure 3). Totally, we have 1327 states in the state space $\boldsymbol{S}$.

One feature of our state space is that it does not include explicit description of obstacles. Instead, the occlusion status of the target area tells indirectly the status of obstacles.

Although the state space seems complete, it suffers from "perceptual aliasing problem" [22] due to the limit of perception. It is generally defined as "a problem caused by multiple projections of different actual situations into one observed state." The multiple projections make it very difficult for a robot to take an optimal action. To find such states (called "hidden states"), we estimate the state transition probabilities $P_{ij}(a)$ by using the MLE (Maximum Likelihood Estimation) that is denoted as follows:

$$P_{ij}(a) = \frac{n_{ij}^a}{\sum_{j \in \boldsymbol{S}} n_{ij}^a},$$

where, n_{ij}^a indicates the number of observations where the robot takes an action a at state i and as a result the state transits to state j. After memorizing the history of these transitions to some extent during the learning process, we estimate the state transition probabilities $P_{ij}(a)$. If the state transition probability density function has multiple peaks, the state is a candidate for a hidden state. By allocating one memory to each candidate of the hidden states, the candidate state can be discriminated as a hidden state [23]. After finding hidden states and adding them to the state space, we apply Q-learning again to determine the adequate actions in the hidden states.

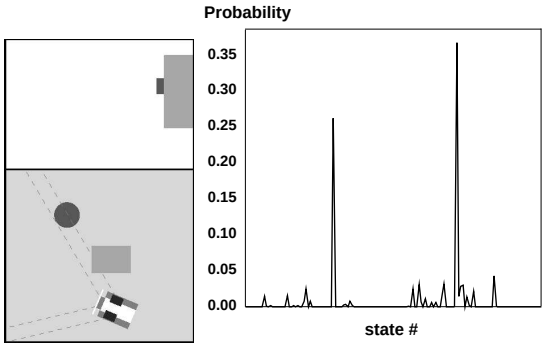
5 Experimental Results

The experiment consists of two parts: first, learning the optimal policy f through the computer simulation, then applying the learned policy to a real situation.

5.1 Simulations

In applying Q-learning to our task, we have to define an action space. Our robot can select an action to be taken in the current state of the environment. The robot moves around using a PWS (Power Wheeled Steering) system with two independent motors. Since we can send the motor control command to each of the two motors separately, we construct the action set in terms of two motor commands ω_l and ω_r, each of which has 3 sub-actions, forward, stop, and backward. All together, we have 9 actions.

Due to the peculiarity of visual information, that is, a small change near the observer results in a large change in the image and a large change far from the observer may result in a small change in the image, one action does not always correspond to one state transition. We called this the **"state-action deviation problem"** [24]. To avoid this problem, we reconstruct the action space as follows. Each action defined above is regarded as an action primitive. The robot continues to take one action primitive until the current state changes. This sequence of the action primitives is called an action.



(a) A hidden state

(b) A state transition probability

Figure 4: **An example of hidden states**

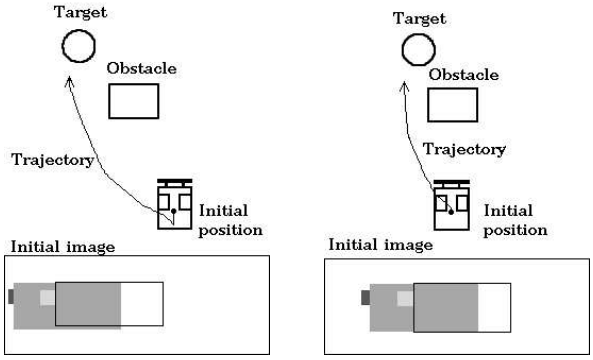
Figure 4 shows an example of hidden state and its state transition probability density function estimated by MLE. The top of Figure 4(a) indicates the right image where the target is observed (right side is occluded) while the target is not observed in the left image (not shown). Since the disparity information is not available, the position of the target cannot be determined. This means that in the left image, the target is completely occluded or disappeared. Two peaks in Figure 4 (b) correspond to these two situations. As a result, 12 hidden states were found.

The goal state is shown in Table 1. We give a reward 1 when the robot achieves the goal state, otherwise 0. When the robot makes a collision with an obstacle, we do not give any negative rewards but reset the robot position because the negative rewards make many local maxima of Q-values. Although the convergence time might spend a lot, reset and 0 reward indirectly suggest the negative situations

Table 1: **A goal state**

image	appearance	position	disparity
left	*no occlusion*	*right*	*near*
right	*no occlusion*	*left*	*near*

Figure 5 shows examples of the target reaching behavior. The bottom of Figures 5 (a) and (b) show input stereo images taken at initial position, respectively, where two images are overlapped into one. As shown in Figure 5 (a), the robot acquired such a behavior that the robot moves backward until the robot can see the target clearly (*action for vision*) then, avoids the obstacle, and finally reaches the target (*vision for/during action*). It seems that the robot plans the viewpoint from which the robot can avoid occlusion and moves. Note that, it's not a result of planning but one of learning. The robot learned to control the position and direction of its own visual sensor actively and to improve the quality of information obtained by varying the position and the orientation of viewpoint so as to expand the observation space.

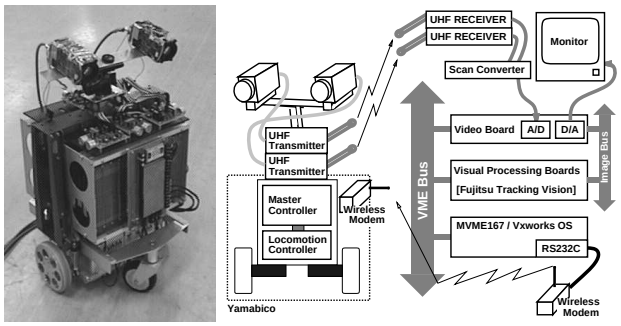


(a) Example 1 (b) Example 2

Figure 5: **Reaching behavior**

5.2 Real Robot Experiments

Figure 6 shows our real robot system. The parallel stereo cameras are set on a mobile platform (Yamabico) controlled by MVME167/VxWorks OS through RS232C. The base line, the tilt angle, the camera height and the visual angles of both cameras are about 17cm, 10 degrees, 60cm and 60 degrees, respectively. The maximum vehicle speed is about 60cm/s. The images taken by stereo cameras mounted on the robot are transmitted to UHF receivers and subsampled by the scan-line converter (Sony Corp.). Then, the video signal is sent to a Fujitsu tracking module. One visual tracking board has a capability of video rate tracking of about 140 windows. We are now using four boards, but the procedures of coarse-to-fine stereo matching, region clustering with MDL, and tracking are too much for one CPU board, and now it takes about 160ms for one cycle (about $6Hz$).



(a) A robot (b) A system architecture

Figure 6: **Our robot and system**

Figure 7 shows a sequence of stereo images taken during the target reaching behavior in which the robot succeeded in reaching the target avoiding the obstacle. Along with these figures, a sequence of region tracking images are shown where the proposed region segmentation and tracking method can cope with changes of target size in image, and successfully track the target region (the region like a mesh in the Figure 7). A larger rectangle and a cluster of smaller ones indicates the target area and occluding ones, respectively. Table 2 shows the result of state discrimination for the scene shown in Figure 7. In Table 2, state step that the robot discriminated, target states in both left and right images (Occlusion status: **N**o **O**cclusion or **L**eft side **O**cclusion, Position: **L**eft, **C**enter, Disparity: **N**ear, **M**edium), and control commands to right and left motors (**F**orward , **S**top, **B**ackward) are shown. The images at state steps marked "*" are shown in Figure 7. The numbers in () shows the sampling steps. Although a state discrimination at a state step may fail, the robot succeeded in achieving the given task because the errors do not occur continuously and the image processing is done by utilizing the information extracted from consecutive frames.

At state step 2, the robot takes a backward action. This shows the "*action for vision*" described in section 5.1 because the backward action is useful for expanding the field of view.

6 Discussion and Future Works

Our robot sometimes fails. The first problem is due to the fluorescent lamp from the ceiling. It makes texture(shadow and brighter place) on the floor although we assumue no textures on the floor. The robot pursues or avoids it, because the robot perceives that it is the target or the obstacle. To cope with this problem, we plan to use other information such as color and the image pattern of the target region.

Although it takes long time to converge, the learning method can find a sequence of feasible actions for the robot to take. If we assign a reward function according to the Euclidean distance to the goal to speed up the learning, we would suffer from local maxima

Table 2: **State-Action data in a real environment**

state step	state		action	
	left	right	L	R
1*(1-11)	(NO,C,M)	(NO,C,M)	F	F
2*(12-14)	(LO,C,M)	(NO,C,M)	S	B
3(15-16)	(LO,C,M)	(NO,C,M)	F	B
4(17)	(LO,C,M)	(NO,L,M)	F	B
5*(18)	(LO,L,M)	(NO,L,M)	F	F
6*(19-21)	(NO,L,M)	(NO,L,M)	S	F
7*(22-28)	(NO,C,M)	(NO,L,M)	F	F
8(29-38)	(NO,C,N)	(NO,L,N)	F	F
9(39)	(NO,C,N)	(NO,C,N)	S	F
10*(40-44)	(NO,C,N)	(NO,L,N)	F	F

of Q-values because the Euclidean distance measure cannot always reflect the length of the action sequence because of the non-holonomic property of the mobile robot. The learning method does not need to care about these issues.

Now, we are planning to extend the method to the environment where the target and/(or) the obstacle are/(is) moving.

References

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

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

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

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

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

[6] Y. Kuniyoshi. "A compact mobile robot with binocular tracking vision". *Journal of the Robotics Society of Japan*, 13:3:343–346, 1995.

[7] E. Huber and D. Kortenkamp. Using stereo vision to pursue moving agent with a mobile robot. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, pages 2340–2346, 1995.

[8] K. Nishihara. "Practical real-time imaging stereo matcher". *Optical Engineering*, 23-5, 1984.

[9] P. Maes and R. A. Brooks. "Learning to coordinate behaviors". In *Proc. of AAAI-90*, pages 796–802, 1990.

[10] J. H. Connel and S. Mahadevan. "Rapid task learning for real robot". In J. H. Connel and S. Mahadevan, editors, *Robot Learning*, chapter 5. Kluwer Academic Publishers, 1993.

[11] 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.

[12] D. Marr and T. Poggio. "A computatinal theory of human stereo vision". In *Proc. of Royal Soc. London B204*, pages 301–338, 1979.

[13] W. E. L. Grimson. "A computatinal theory of visual surface interpolation". In *Proc. of Royal Soc. London B298*, pages 395–427, 1982.

[14] Takayuki Nakamura and Minoru Asada. "Motion sketch: Acquisition of visual motion guided behaviors". In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 126–132, 1995.

[15] G. H. Ball and D. J. Hall. "ISODATA, a novel method of data analysis and pattern classification". *Stanford Research Institute*, AD-699616, 1965.

[16] J. Rissanen. *Stochastic Complexity in Statistical Inquiry*. World Scientific, 1989.

[17] T. Nakamura. *Behavior Acquisition for Vision-Based Mobile Robots"*. PhD thesis, Dept. of Mech. Eng. for Computer-Controlled Machinery Osaka University, January 1996.

[18] C. J. C. H. Watkins. *Learning from delayed rewards"*. PhD thesis, King's College, University of Cambridge, May 1989.

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

[20] L. P. Kaelbling. "Learning to achieve goals". In *Proc. of IJCAI-93*, pages 1094–1098, 1993.

[21] S. D. Whitehead. "A complexity analysis of cooperative mechanisms in reinforcement learning". In *Proc. AAAI-91*, pages 607–613, 1991.

[22] S. D. Whitehead and D. H. Ballar. "Active perception and reinforcement learning". In *Proc. of Workshop on Machine Learning-1990*, pages 179–188, 1990.

[23] R.A. McCallum. "Instance-based utile distinctions for reinforcement learning with hidden state". In *Proc. of the 12th Int. Conf. on Machine Learning*, pages 387–395, 1995.

[24] M. Asada, S. Noda, S. Tawaratsumida, and K. Hosoda. Vision-based reinforcement learning for purposive behavior acquisition. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, pages 146–153, 1995.

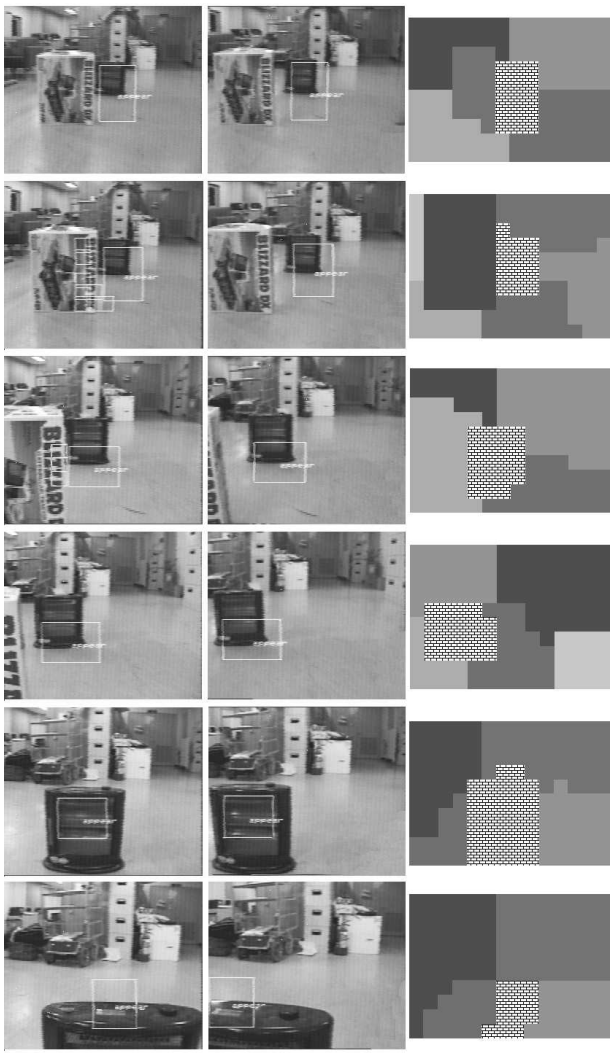


Figure 7: **Reaching behavior by the real robot in the environment with the obstacle**