Evolutionary Robot Vision for People Tracking Based on Local Clustering

Indra Adji Sulistijono[*,1,2] and Naoyuki Kubota[1,3]

[1]*Department of Mechanical Engineering, Graduate School of Engineering, Tokyo Metropolitan University, 6-6 Asahigaoka, Hino, Tokyo 191-0065, Japan*

[2]*Department of Mechatronics, Electronics Engineering Polytechnic Institute of Surabaya (EEPIS-ITS), Kampus ITS Sukolilo, Surabaya 60111, Indonesia*

[3]*SORST, Japan Science and Technology Agency, Japan*

Abstract: This paper discusses the role of evolutionary computation in visual perception for partner robots. The search of evolutionary computation has many analogies with human visual search. First of all, we discuss the analogies between the evolutionary search and human visual search. Next, we propose the concept of evolutionary robot vision, and a human tracking method based on the evolutionary robot vision. Finally, we show experimental results of the human tracking to discuss the effectiveness of our proposed method.

Keywords: Robot Vision, partner robots, evolutionary computation, clustering.

INTRODUCTION

Visual systems have been discussed from various points of view [1-10]. Furthermore, various types of intelligent methods [11-23] have been applied for image processing in real world applications thanks to the development of cheap and small digital cameras and signal processing boards with low energy consumption. Especially, information extraction by image processing plays the essential role in monitoring systems, surveillance systems, and automatic control. The visual systems including image processing can be divided into passive vision and active vision. In general, the passive vision is used for the focused information extraction toward a specific direction, while the active vision is used for the information extraction by updating the sensing direction and range in the vast area where the range of visual scene is restricted. In fact, the active vision is one of the main methods for the information extraction of an intelligent robot in order to extract perceptual information from the environment.

Various technologies for image processing are required for realizing the robot vision, *e.g.*, color processing, target detection, template matching, shape recognition, motion extraction, and optical flow. We have applied spiking neural networks, cellular neural networks, self-organizing map, and others for human detection, motion extraction, and shape recognition [24-30]. In this paper, we focus on people tracking. The tracking problem of people or objects is significantly harder than that of a single person or object. The people tracking problem includes two problems of people detection problem in each image and a tracking problem of detected people over time. In previous works, people tracking problems have been mainly solved by appearance-based methods. Kalman filter, particle filters, genetic algorithms, particle swarm optimization, and others [11-22] have been applied in appearance-based methods. Furthermore, dynamic model of human movement is also applied to improve the accuracy of people tracking. These methods try to detect the features of human appearance, and to trace them over time, but there are problems on variability of appearance features and computational cost in the real-time people tracking. Therefore, we use only colors corresponding to human face and hair, and propose a local genetic algorithm based on clustering to realize the fast coarse search for people tracking.

This paper is organized as follows. Section II discusses the analogy between human visual search and evolutionary search. Section III and IV explain the detail of the local genetic algorithm based on clustering and people tracking for partner robots. Section V shows preliminary simulation results of the proposed method in a dynamic environment, and experimental results of people tracking.

EVOLUTIONARY ROBOT VISION

Evolutionary computation (EC) is a field of simulating evolution on a computer [13]. Evolutionary optimization methods are fundamentally iterative generation and alternation processes of candidate solutions. The optimization is done by the multi-point search operating on a set of individuals, which is called a population. First, we discuss the role of evolutionary search in dynamic environments. Fig. (**1**) shows the temporal patterns of spatial changes in dynamic environments where the vertical axis indicates the state of environmental conditions represented as a value. If the search speed of EC is faster than the changing speed of the environmental conditions, EC can obtain feasible solutions in the environmental conditions. However, EC should be adaptive to the facing environmental conditions if the environmental changes can be observed. If the environmental change is very slowly and the change is small (Fig. **1a**), the mutation range should be large according to the amount of the environmental change. Basically, this kind of

*Address correspondence to this author at the Department of Mechanical Engineering, Graduate School of Engineering, Tokyo Metropolitan University, 6-6 Asahigaoka, Hino, Tokyo 191-0065, Japan; Tel/Fax: +81-42-585-8441; E-mail: indra-adji@ed.tmu.ac.jp; indra@eepis-its.edu

1874-110X/08 2008 Bentham Science Publishers Ltd.

change can be considered as some noise in a relatively stationary environment. The environment of Fig. (**1b**) includes big changes, but the environmental condition is relatively stationary after a big change. If the big change can be observed, most of candidate solutions should be replaced with randomly generated candidate solutions. The environment of Fig. (**1c**) is changing non-stationary with both features of Fig. (**1a**,**b**). In general, the change of visual images corresponds to the environment of Fig. (**1c**). The visual image of a mobile robot changes according to both the dynamics of environmental changes and the robotic motion. Therefore, we discuss visual perception based on active robot vision and EC from the viewpoint of human visual perception.

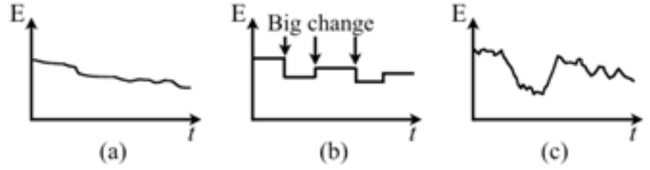
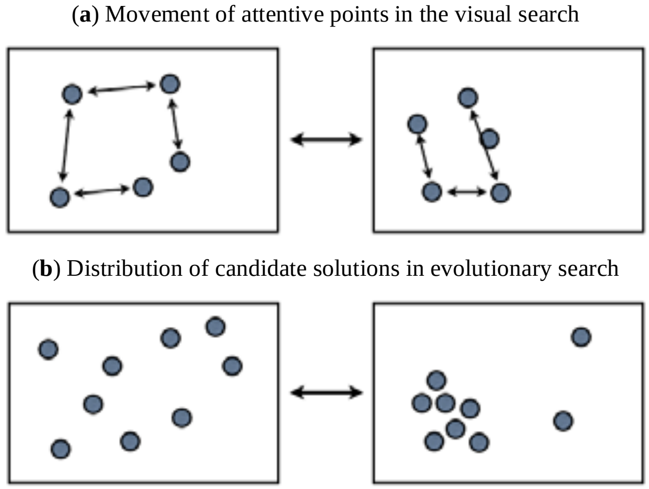


Fig. (1). Patterns of changes in dynamic environments.

To realize a visual perception system for a robot, we should take recent works of psychology into account, especially, sensation, perception, and attention. Sensation is the basic information presented to our sense organs, while perception is organized and involves a process of attaching meaning to sensations. W. James [1] emphasized the importance of selective attention in the following; "*Millions of items of the outward order are present to my senses which never enter into my experience. ... My experience is what I agree to attend to. Only those items which I notice shape my mind*". The research of selective attention leads to the problem of figure-ground organization. Visual perception is organized into a central object called *figure* and its blurred surroundings called *ground*. Our visual system operates in a flexible and adaptive manner to perceive the environment by using bottom-up and top-down processes. Bottom-up processing depends directly on external stimuli, while top-down processing is influenced by expectations, stored knowledge, context, and so on.

Fig. (**2**) shows the comparison between the visual search and evolutionary search. The left and right figures show the distribution of search points in the focused search and distributed search, respectively. The visual search controls the searching area based on the fast movement of attentive point. The region of interest (ROI) for the information extraction is deeply related with the search of geometrical features included in the visual target. The search results are reflected to the next visual search. On the other hand, the evolutionary search controls the searching area based on the selection pressure to candidate solutions. If the selection pressure is high, the candidate solutions are centralized toward the better candidate solutions. Otherwise, candidate solutions are globally distributed in the search space. The next search points in the evolutionary search are generated by crossover and mutation. The search in ROI is mainly performed by mutation and local search, while a new search point for ROI is generated by crossover. The degree of interest is calculated by the fitness value. If the fitness value is high, the focused search should be performed. We apply a steady-state genetic algorithm (SSGA) [16] to realize the continuous and real-time search for the robot vision like human visual perception in a dynamic environment.



(**a**) Movement of attentive points in the visual search

(**b**) Distribution of candidate solutions in evolutionary search

Fig. (2). Comparison of visual search and evolutionary search.

LOCAL GENETIC ALGORITHM BASED ON CLUSTERING

A. Local Genetic Algorithm in Dynamic Environment

The previous section discussed the features of EC in dynamic environment. Next, we discuss a method for the optimization and adaptation in a dynamic environment. Generally, it is very difficult to realize both optimization and adaptation in a real world problem. In order to perform the optimization, it takes much computational time and cost, but the environmental condition might change much. Therefore, the real-time adaptation should be done, but all the population should trace the local minima as much as possible in real-time adaptation, because the current best solution is not guaranteed as the best solution in future. We propose a local genetic algorithm based on clustering (LGAC) as a distributed search method based on local hill-climbing of clustered individuals (Fig. **3**).

Basically, each individual is composed of diploid, i.e., the self-best solution and candidate solution. If the fitness value of the candidate solution is larger than that of the self-best solution, the candidate solution is replaced with the self-best solution. In this way, each individual performs the elitist selection. We use the term of personal best or self-best that inspired from particle swarm optimization (PSO) invented by Eberhart and Kennedy [14,15]. Furthermore, we use the elitist crossover in the following updating rule;

$$\begin{aligned} x_{i,j} \leftarrow\ & x_{i,j} + \alpha_1 r_1 (x^S_{i,j} - x_{i,j}) \\ & + \alpha_2 r_2 (x^S_{L,j} - x_{i,j}) + \alpha_N N(0,1) \end{aligned} \qquad (1)$$

where $x^S_{i,j}$ is the self-best solution; $x^S_{L,j}$ is the locally best solution in a cluster; r_1 and r_2, are uniform random value between 0 and 1.0; $N(0,1)$ is a normal random value with average of 0 and 1.0, and α_1, α_2, and α_N are coefficients. Furthermore, we can use adaptive mutation as follows;

$$\alpha_N = \beta_1 \cdot \frac{f_i - f_{\min}}{f_{\max} - f_{\min}} + \beta_2 \tag{2}$$

where $f_{\max}$ and $f_{\min}$ are the maximal value and minimal value of fitness values in a local cluster or the population; β_1 and β_2, are coefficient and offset, respectively. The proposed method is similar to PSO, but in this paper, we explicitly use a mutation factor in order to trace local minima in a dynamic environment.

We use the *k*-means algorithm [23] as a clustering method. The *k*-means algorithm is one of the most popular iterative descent clustering methods. The inputs to *K*-means algorithm are $(x_{i,1}, x_{i,2}, ..., x_{i,m})$ of the *i*th candidate solution. The number of clusters is *K*. When the reference vector of the *k*th cluster is represented by $r_k = (r_{k,1}, r_{k,2}, ..., r_{k,2})$, the Euclidian distance between the *i*th input vector $u_i = (x_{i,1}, x_{i,2}, ..., x_{i,m})$ and the *k*th reference vector is defined as

$$d_{i,k} = \| u_i, r_k \| \tag{3}$$

Next, the reference vector minimizing the distance $d_{i,k}$ is selected by

$$c_i = \arg\min_k \left\{ \| u_i, r_k \| \right\} \tag{4}$$

where c_i is the cluster number which the *i*th input belongs to.

After selecting the nearest reference vector to each input, the *k*th reference vector is updated by the average of the inputs belonging to the *k*th cluster. If the update is not performed at the clustering process, this updating process is finished. Fig. (**4**) shows an example of the search of the proposed method in a dynamic environment using Gaussian functions. As shown in Fig. (**3**), the number of local clusters should be larger than that of modalities in the dynamic environment. However, the change of fitness landscape is unknown beforehand. Therefore, we prepare the sufficient number of clusters. In this way, the genetic search is mainly conducted within a local cluster.

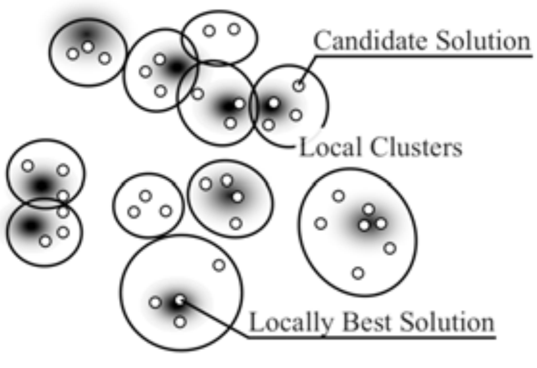
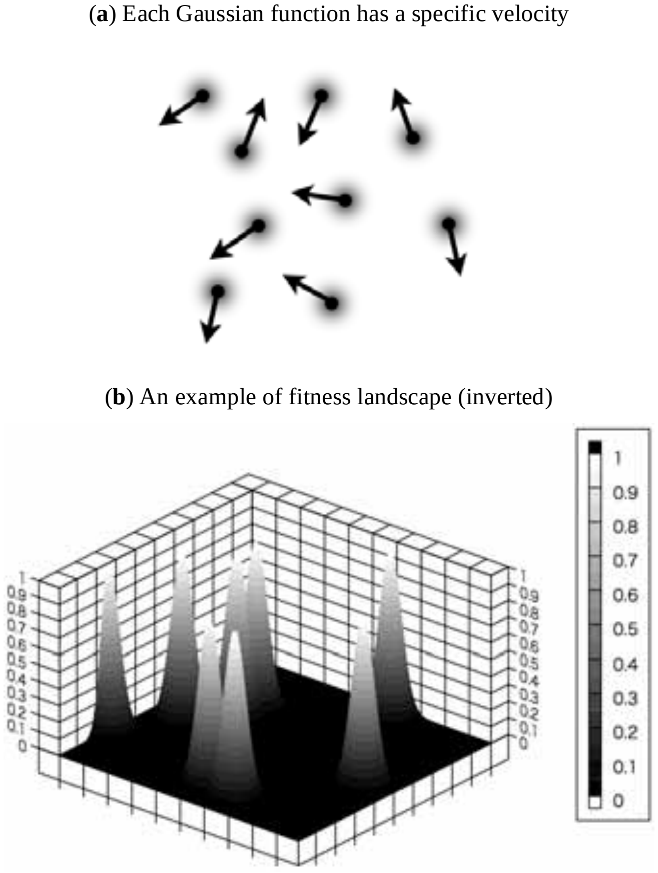


Fig. (3). Search of LGAC in a dynamic environment; fitness landscape is depicted as monochrome gradation.


(**a**) Each Gaussian function has a specific velocity

(**b**) An example of fitness landscape (inverted)

Fig. (4). Patterns of changes in dynamic environments.

B. Preliminary Simulation Results

In order to perform preliminary comparison, we use the following fitness function to be minimized;

$$f_i = 1 - \max_k \exp\left(-\gamma_k \left(\left(y_{k,1} - x_{i,1}\right)^2 + \left(y_{k,2} - x_{i,2}\right)^2 \right)\right) \tag{5}$$

where $(y_{k,1}, y_{k,2})$ is the center of the *k*th Gaussian function; γ_k is the coefficient, $(x_{i,1}, x_{i,2})$ is *i*th candidate solution (Fig. **4**). The position of each central point is updated by using a velocity vector $(v_{k,1}, v_{k,2})$ that is predefined. The reason why we used the above max operator is to keep the minimal value at 0 through the change of dynamical environment. In order to discuss the traceability of the candidate solutions to each local minimum, we use the following distance index;

$$l_k = \min_i \left(\left(y_{k,1} - x_{i,1}\right)^2 + \left(y_{k,2} - x_{i,2}\right)^2 \right) \tag{6}$$

In this paper, we use 9 Gaussian functions. The population size is 100. We show preliminary simulation results by using the following parameters in Table **1**. Case 1 and Case 2 mainly use the self-best solutions; Case 3 and Case 4 mainly use the local best solutions; Case 5 and 6 uses no adaptive mutation and Case 7~10 uses all parameter solutions.

Fig. (**5**) shows snapshots of the search of candidate solutions in Cases 2, 6, and 10 at the final generation of 5000 generations. We obtained similar simulation results on Cased

1, 2, 3, and 4. In Case 2, it is a little bit difficult to trace all Gaussian functions, because the search of candidate solutions strongly depends on the locally best solutions. In Case 6 without adaptive mutation, it's very difficult to trace all Gaussian functions, because the search of candidate solutions strongly depends on the locally best solutions, and all candidate solutions converged into the locally best solutions quickly. On the other hand, in Case 10, all the candidate solutions can trace all the Gaussian functions well.

Table 1. Parameters Used in Simulations

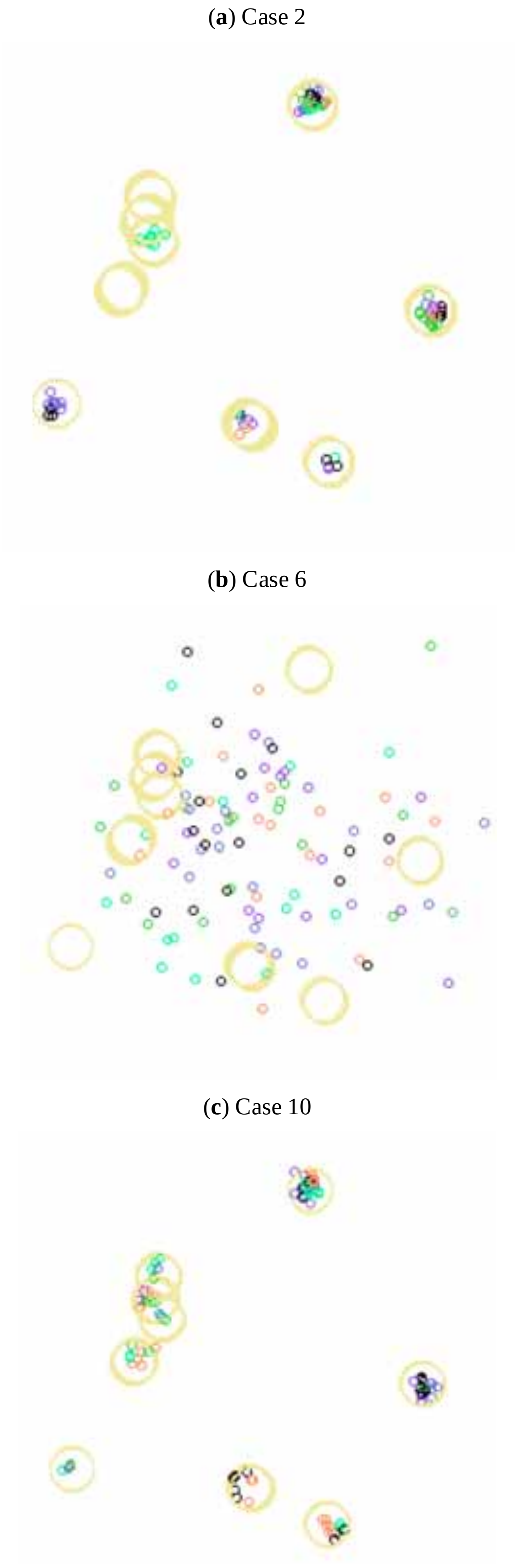
Case	α_1	α_2	α_N	K
Case 1	0.2	0	0.15	20
Case 2	0.2	0	0.15	50
Case 3	0	0.2	0.15	20
Case 4	0	0.2	0.15	50
Case 5	0.2	0.2	0	20
Case 6	0.2	0.2	0	50
Case 7	0.2	0.2	0.15	20
Case 8	0.2	0.2	0.15	30
Case 9	0.2	0.2	0.15	40
Case 10	0.2	0.2	0.15	50

Next, we conducted comparison of performance in detail. Figs. (**6**,**7**) show the simulation results of Cases 1 and 2. Each candidate solution tries to trace Gaussian functions separately and non-cooperatively. The average fitness value is decreasing soon, but the maximal value is still large until the end of generations. In this result, the maximal distance is increasing after 1000 generation. When they are separating after several Gaussian functions are overlapped in the middle, the candidate solutions cannot trace some of them. Here the self-best solution is not the best every generation. As a result, it is difficult to maintain better solutions well. However, the difference between the current solution and self-best solution makes the temporal descent direction. Therefore, the candidate solutions can trace Gaussian functions, but the tracing accuracy is not so good.

Figs. (**8**,**9**) show the simulation results of Cases 3 and 4. In the cases, each individual tries to trace Gaussian functions according to the information of only the locally best solutions, not the self best solution. As a result, the distributions of candidate solutions are larger than Cases 1 and 2. Furthermore, the performance of Cases 3 and 4 depends strongly on the number of clusters. If the number of clusters is small, the average fitness value is improved, but the traceability is decreasing.

Figs. (**10**,**11**) show the simulation results of Cases 5 and 6. Only the use of self-best and locally best solutions cannot guarantee to trace all Gaussian functions although the number of clusters is large. This results show the effectiveness of adaptive mutation in the search in a dynamic environment.

Figs. (**12-15**) show the simulation results of Cases 7 to 10. The traceability is improved as the number of clusters is increasing. In case 10, all the population can trace all Gaussian functions well.

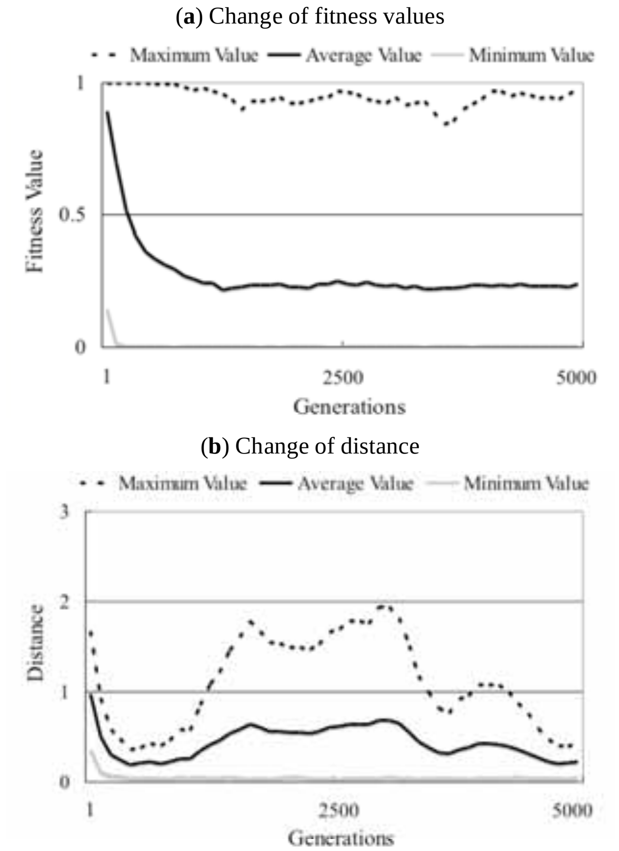


(**a**) Case 2

(**b**) Case 6

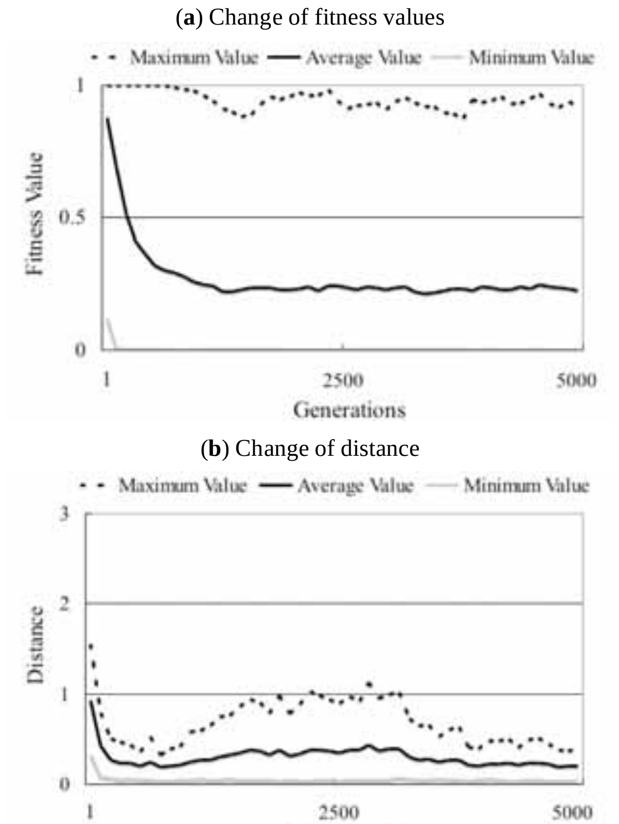
(**c**) Case 10

Fig. (5). The distribution of candidate solutions in a dynamic environment; big yellow circle indicates the position of each Gaussian function.


(**a**) Change of fitness values

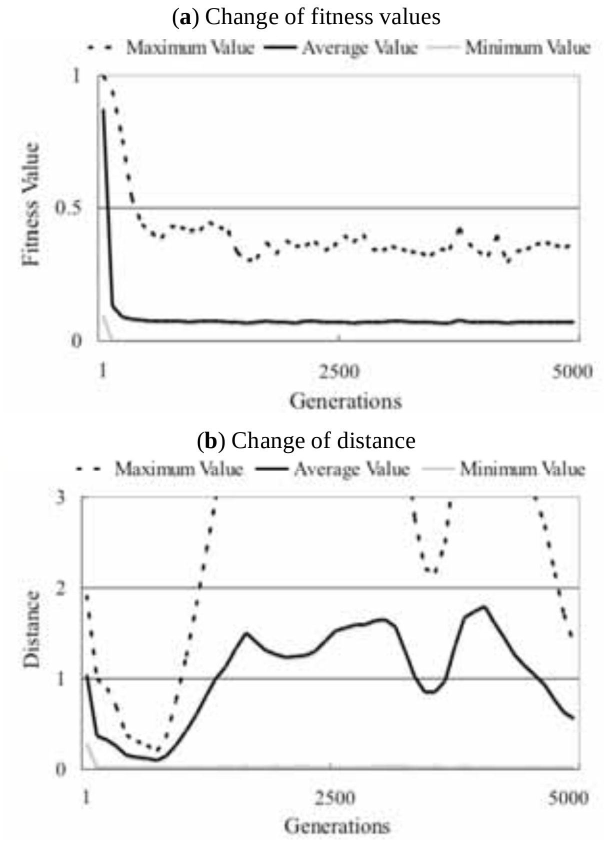
(**b**) Change of distance

Fig. (6). The traceability of candidate solutions in a dynamic environment in Case 1.


(**a**) Change of fitness values

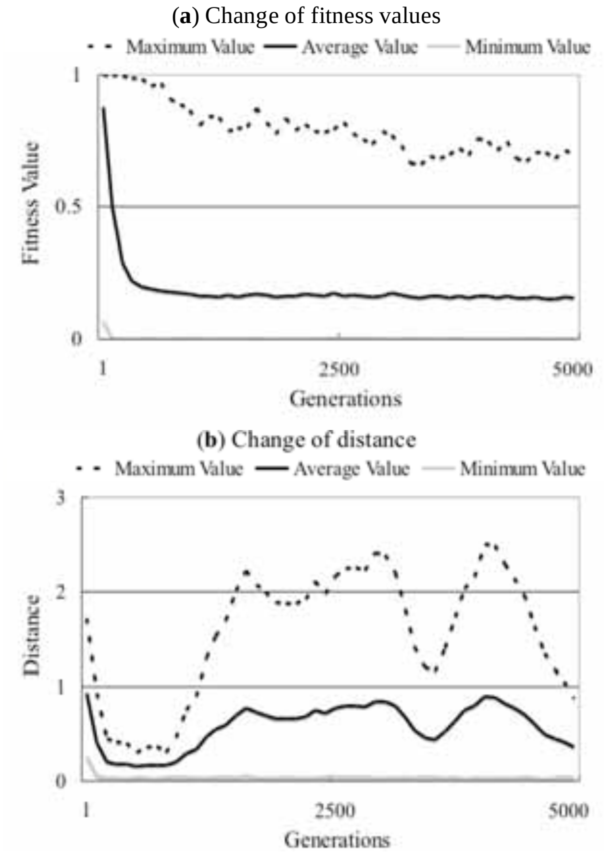
(**b**) Change of distance

Fig. (7). The traceability of candidate solutions in a dynamic environment in Case 2.


(**a**) Change of fitness values

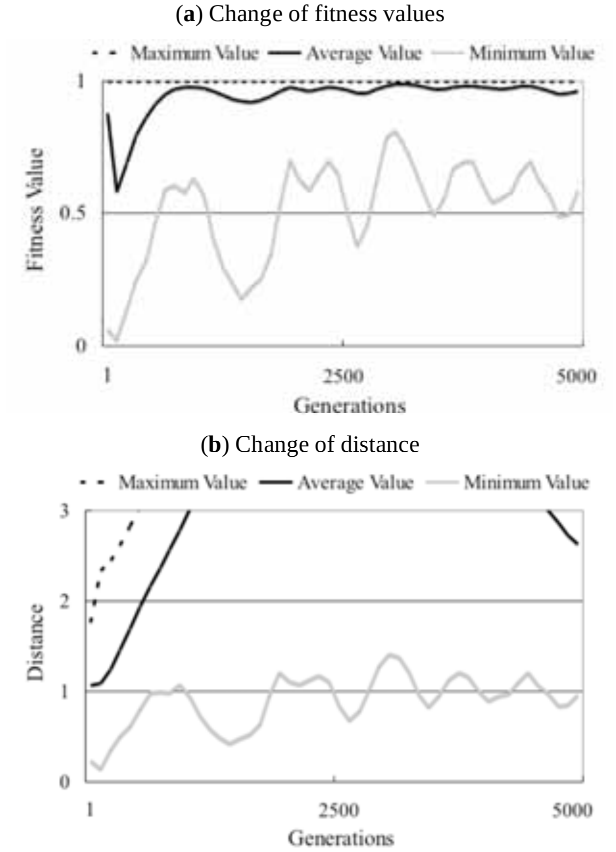
(**b**) Change of distance

Fig. (8). The traceability of candidate solutions in a dynamic environment in Case 3.


(**a**) Change of fitness values

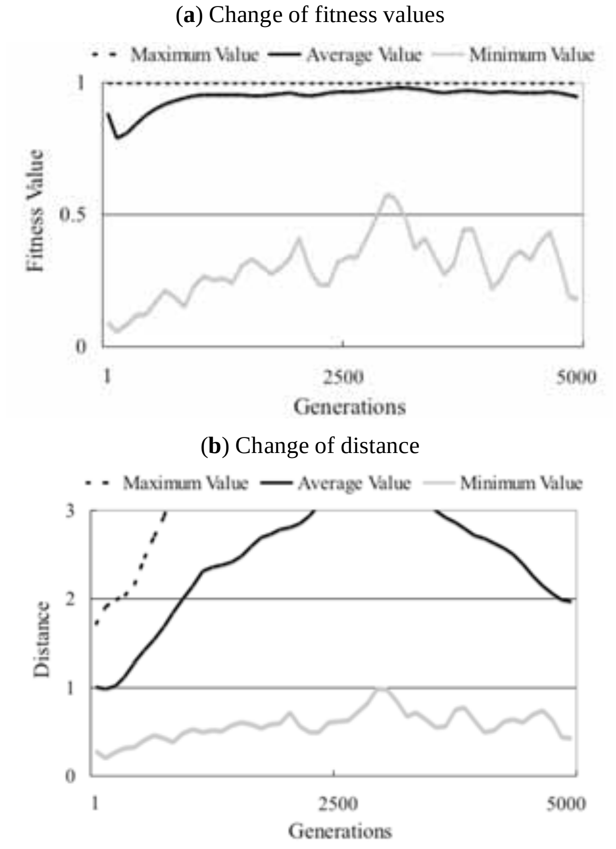
(**b**) Change of distance

Fig. (9). The traceability of candidate solutions in a dynamic environment in Case 4.


(**a**) Change of fitness values

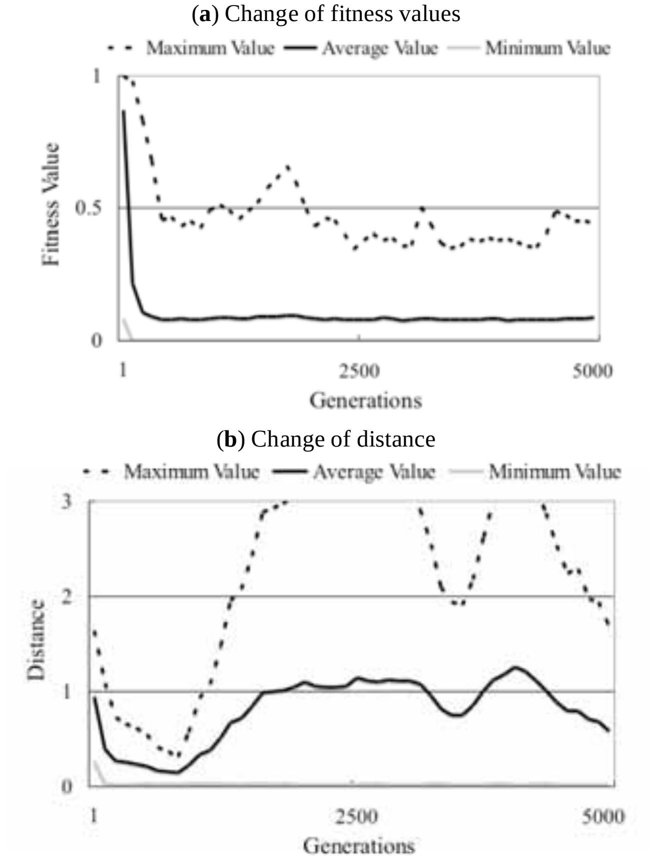
(**b**) Change of distance

Fig. (10). The traceability of candidate solutions in a dynamic environment in Case 5.


(**a**) Change of fitness values

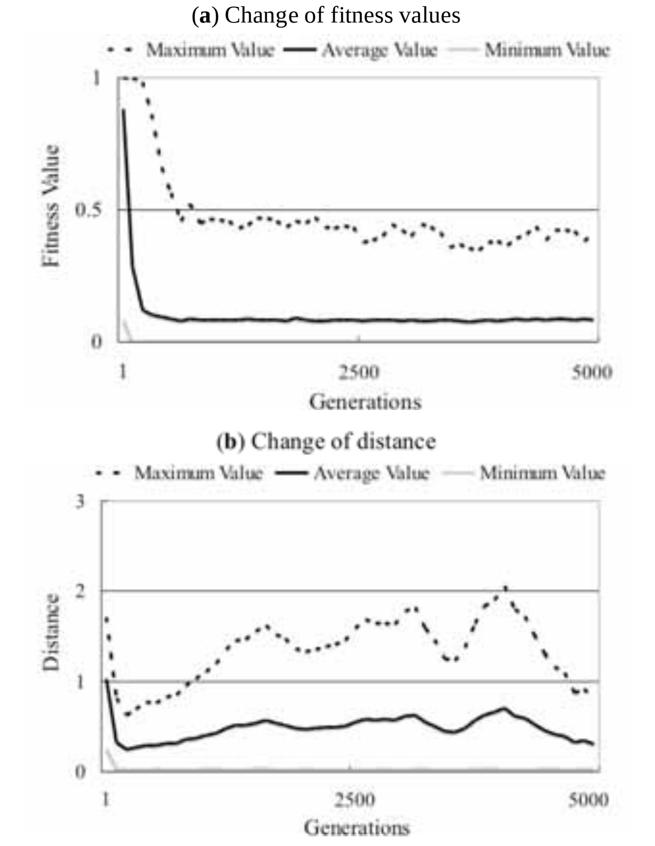
(**b**) Change of distance

Fig. (11). The traceability of candidate solutions in a dynamic environment in Case 6.


(**a**) Change of fitness values

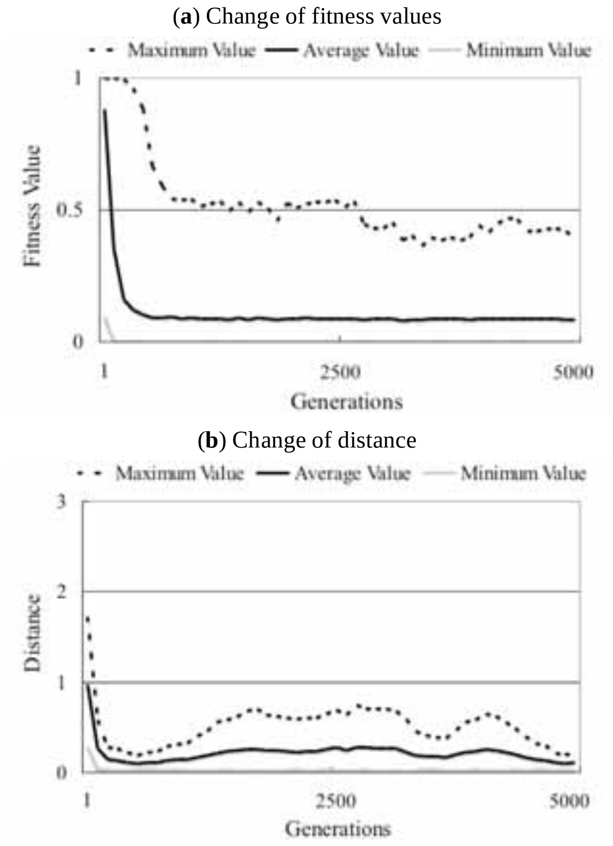
(**b**) Change of distance

Fig. (12). The traceability of candidate solutions in a dynamic environment in Case 7.


(**a**) Change of fitness values

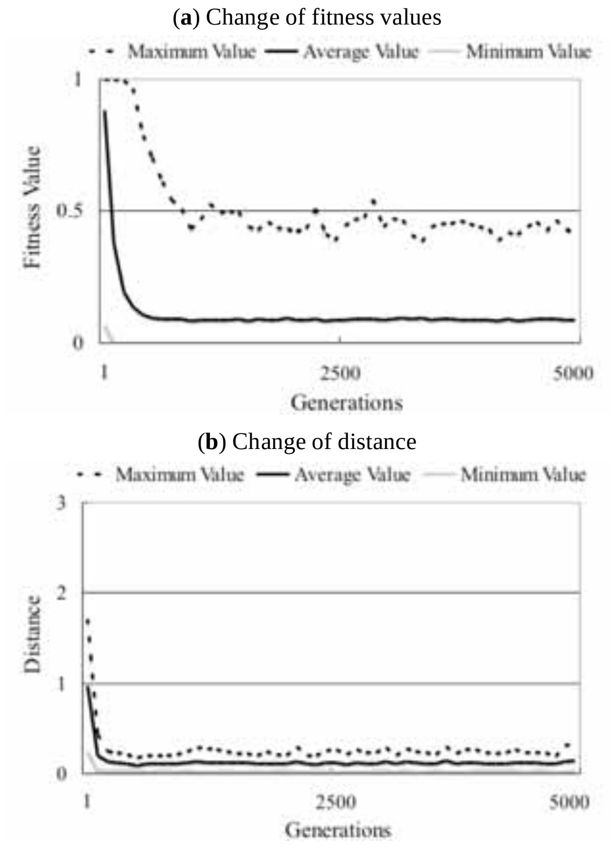
(**b**) Change of distance

Fig. (13). The traceability of candidate solutions in a dynamic environment in Case 8.


(**a**) Change of fitness values

(**b**) Change of distance

Fig. (14). The traceability of candidate solutions in a dynamic environment in Case 9.


(**a**) Change of fitness values

(**b**) Change of distance

Fig. (15). The traceability of candidate solutions in a dynamic environment in Case 10.

Figs. (**6**,**8**,**10**,**12**) show the simulation results where the number of clusters is set at 20, and Figs. (**7**,**9**,**11**,**15**) show the simulation results where the number of clusters is set at 50. In case of 20 clusters, the traceability of Gaussian functions is not good. On the other hand, the traceability of 50 clusters is better than that of 20 clusters. However, as the number of clusters increases, the computational cost also increases. Therefore, the number of clusters should be reduced as much as possible.

To summarize, the self-best and locally best solutions are useful to generate the search directions. The self-best parameter affects better result than locally best parameter with adaptive mutation (Figs. **6-9**) but the result of both the self-best and locally best solutions shows the difficulty to trace all Gaussian functions without the adaptive mutation (Figs. **10**,**11**). And finally, the simulation results using all factors support the effectiveness of the proposed method (Figs. **12-15**).

PEOPLE TRACKING FOR PARTNER ROBOTS

A. Partner Robots

We developed two different types of partner robots; a human-like robot called Hubot and a mobile PC called MOBiMac [24-30] in order to realize the social communication with humans. MOBiMac is composed of two CPUs used for PC and robotic behaviors (Fig. **16**). The robot has two servo motors, four ultrasonic sensors on the sides, four light sensors on the bottom, a microphone, and CCD camera. The basic behaviors of the robots are visual tracking, map building, imitative learning, human classification, gesture recognition, and voice recognition [24-30]. In this paper, we apply the proposed method to realize the people tracking by the partner robots.

B. Human Detection

Human detection is one of the most important tasks for partner robots. Pattern matching has been performed by various methods such as template matching, cellular neural network, neocognitron, and dynamic programming (DP) matching. In general, pattern matching is composed of two steps of target detection and target recognition. The aim of target detection is to extract a target from an image, and the aim of the target recognition is to identify the target from classification candidates. Since the image processing takes much computational time and cost, the full size of image processing to every image is not practical. Therefore, we use the reduced size of image to detect a moving object for the fast human detection.

First, the robot calculates the center of gravity (COG) of the pixels different from the previous image as the differential extraction. The size of image used in the differential extraction is updated according to the previous human detection result. The differential extraction calculates the difference of the number of pixels based on color difference between the previous and current images. If the robot does not move, the COG of the difference represents the location of the moving object. Therefore, the main search area for the human detection can be formed according to the COG for the fast human detection.

The area corresponding to human skin and hair colors is extracted by the proposed method, LGAC, based on template matching. Fig. (**17**) shows a candidate solution of a template used for detecting a target. A template is composed of numerical parameters of $(g_{i,1}, g_{i,2})$, $g_{i,3}$, and $g_{i,4}$. The number of individuals is *G*. Fitness value is calculated by the following equation,

$$F_i = C_{Skin} + C_{Hair} + \eta_1 \cdot C_{Skin} \cdot C_{Hair} - \eta_2 \cdot C_{Other} \tag{7}$$

where C_{Skin}, C_{Hair} and C_{Other} indicate the numbers of pixels of the colors corresponding to human skin, human hair, and other colors, respectively; η_1 and η_2 are the coefficients $(\eta_1, \eta_2 > 0)$.

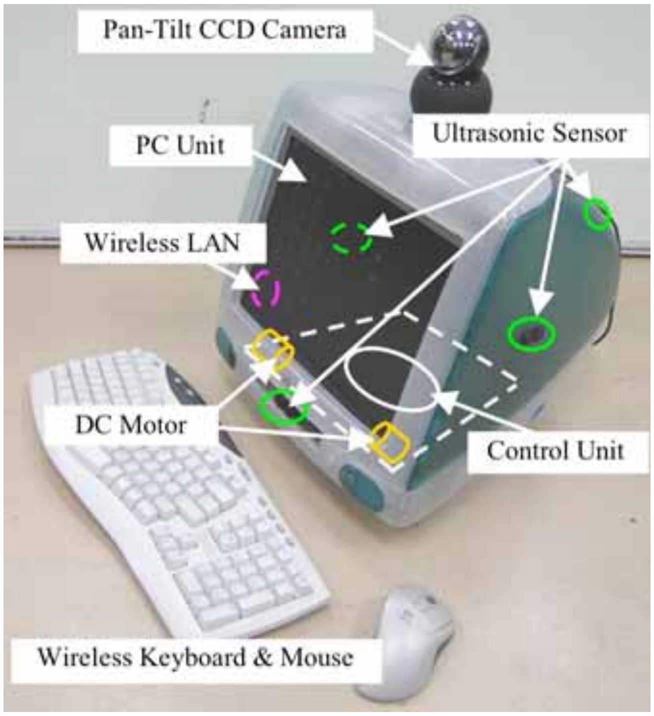


Fig. (16). Hardware architecture of partner robots.

C. Human Tracking

The human tracking is performed based on human candidates generated by LGAC for the human detection. Since LGAC extracts the area of skin colors and hair colors in the human detection, various objects except humans are detected. Therefore, the human tracking is performed according to the time series position of a human candidate $(g_{i,1}, g_{i,2})$. The number of human candidates obtained by LGAC is equal to the number of clusters. Therefore, several human candidates might correspond to a single person in the image (Fig. **18**). The position of a human candidate in the human tracking $(X_{j,1}, X_{j,2})$ is updated by the nearest human candidate position within the tracking range as follows,

$$\begin{cases} X_{j,1} = (1-\lambda)X_{j,1} + \lambda \cdot g_{i,1} \\ X_{j,2} = (1-\lambda)X_{j,2} + \lambda \cdot g_{i,2} \end{cases} \tag{8}$$

where λ is the updating rate. In addition, the width and height of the human candidate for the human tracking $(X_{j,3}, X_{j,4})$ are updated by the size of the detected human $(g_{i,3}, g_{i,4})$,

$$\begin{cases} X_{j,3} = (1-\lambda)X_{j,3} + \lambda \cdot g_{i,3} \\ X_{j,4} = (1-\lambda)X_{j,4} + \lambda \cdot g_{i,4} \end{cases} \tag{9}$$

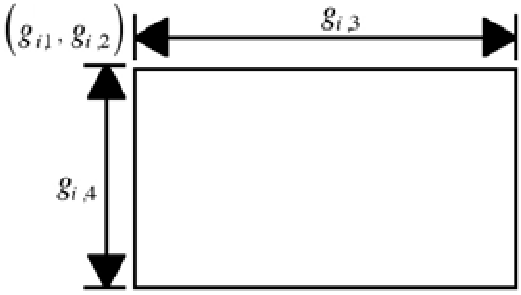


Fig. (17). A candidate solution of a template used for human detection.

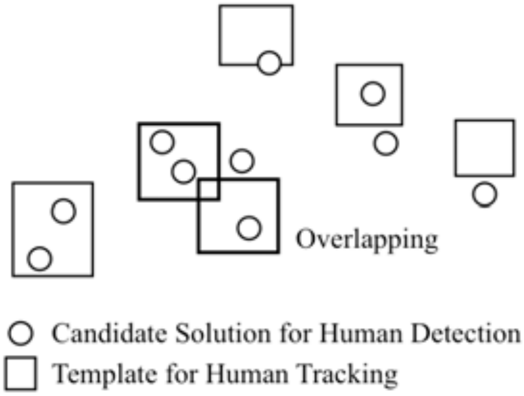
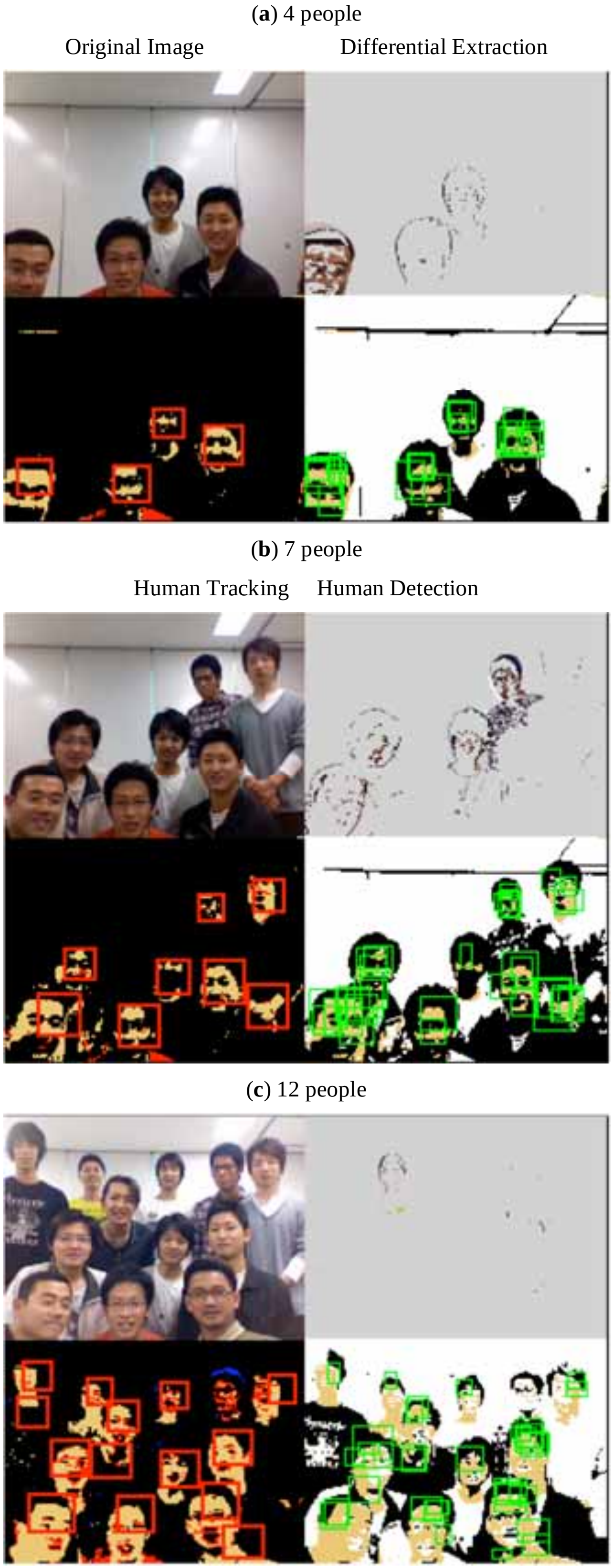

Overlapping
○ Candidate Solution for Human Detection
□ Template for Human Tracking

Fig. (18). Time series position of people tracking.

Furthermore, the time counter is used for the reliability of human tracking. If the human tracking candidate is updated by the nearest human candidate position obtained by LGAC, the time counter is incremented. Otherwise, the time counter is decremented. If the time counter is larger than the predefined threshold, the human count is started. The number of the tracked human is represented at the discrete time *t*. Sometimes, several human candidates are close each other (e.g., overlapping in Fig. **18**), because several human candidates in the detected single human can be generated and overlapped. Therefore, the removal processing is performed when human candidates are coexisting within the tracking range (Fig. **18**).

EXPERIMENTAL RESULTS

This section shows experimental results of people tracking of a partner robot by the proposed method where the maximal number of humans is 12. In the experiment, the number of people is gradually increasing. Fig. (**19**) shows the snapshots in the experimental result of the original image, differential extraction, human tracking results by red boxes, and human detection results by green boxes corresponding to the best solutions in the local clusters.



(**a**) 4 people

(**b**) 7 people

(**c**) 12 people

Fig. (19). Snapshots of the people tracking experimental result.

Fig. (**20**) shows the history of the number of the tracked people. The number of tracked people is 4 in Fig. (**19a**) and this number is correct. The number of tracked people in Fig. (**19b**) is 8, but the actual number of people is 7, because the hand of upper right person is misdetected as a human face. The number of tracked people in Fig. (**19c**) is 14, but the actual number of people is 12, because the neck of upper left and lower right people is misdetected as a human face.

Such a misdetection sometimes occurs, but we can improve this problem by using human facial landmark extraction in our previous research [30]. However, since the aim of the partner robots is not to track all people in the human tracking, but to track the near people for the communication with them. Of course, the precise people tracking is also very important, but the partner robots should perform various functions such as voice recognition and gesture recognition simultaneously. Therefore, the partner robots should perform the human tracking with less computational time and cost.

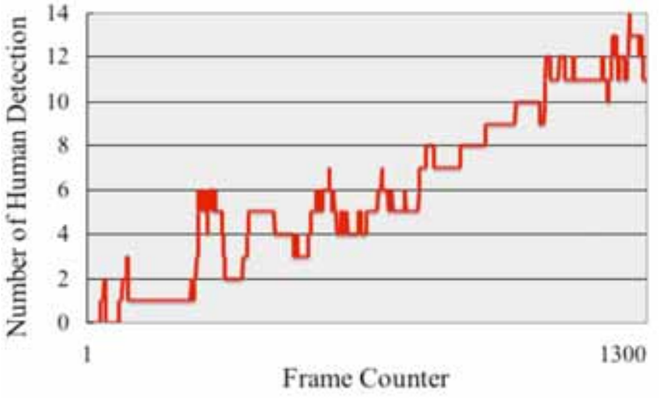


Fig. (20). History of the number of tracking people.

CONCLUSIONS

In this paper, we proposed a local genetic algorithm based on clustering in order to improve the human tracking performance. The proposed method sufficiently performs the tracking of several people. However, the proposed method sometimes misdetects objects similar to skin and hair colors. In order to improve the performance of the human detection, we will incorporate an extraction method of the human facial landmarks into the human detection method [30].

As another future work, we intend to apply the proposed method to the active robot vision by updating the camera angle and position.

REFERENCES

[1] W. James, “Principles of Psychology“, Holt, NewYork, 1890.

[2] J. J. Gibson, “The ecological approach to visual perception”, L. Earlbaum Associates, Publ. 1979.

[3] J. A. Anderson and E. Rosenfeld, “Neurocomputing“, The MIT Press, Cambridge, Massachusetts, USA, 1988.

[4] V. D. Marr and W. H. Freeman, “Vision“, San Francisco, 1982.

[5] K. Fukushima, “Neural Network Model Restoring Partly Occluded Patterns“, *Proc. 7th Knowledge-Based Intelligent Information and Eng. Sys.*, pp. 131-138, 2003.

[6] C. H. Chen, L. F. Pau and P. S. P. Wang, “Handbook of Pattern Recognition & Computer Vision“, World Scientific Publication Co. Pte. Ltd., 1993.

[7] H. H. Bulthoff, S. W. Lee, T. A. Poggio, and C. Wallraven, “Biologically Motivated Computer Vision“, Springer-Verlag, 2002.

[8] A. Mori, S. Uchida, R. Kurazume, R. Taniguchi, T. Hasegawa, and H. Sakoe, "Early Recognition and Prediction of Gestures", Proc. Int. Conf. on Pattern Recognition, Hong Kong, pp. 560-563, 2006.

[9] R. C. Gonzalez and R. E. Woods, Digital Image Processing, Addison Wesley, 1992.

[10] D. A. Forsyth and J. Ponce, *Computer Vision*, A Modern Approach, Prentice Hall, 2003.

[11] D. Goldberg, Genetic Algorithms in Search, “Optimization and Machine Learning“, Massachusetts: Addison Wesley Publishing Company Inc., 1989.

[12] J. H. Holland, “Adaptation in Natural and Artificial Systems“, First MIT Press Ed., Massachusetts: The MIT Press, 1992.

[13] D. Fogel, “Evolutionary Computation“, New York: IEEE Press, 1995.

[14] R. C. Eberhart, J. Kennedy, and Y. Shi, “Swarm Intelligence“, San Francisco: Morgan Kaufmann Publ., 2001.

[15] J. Kennedy and R. Eberhart, “Particle Swarm Optimization” *Proc. IEEE Int. Conf. Neural Netw*., Perth, Australia, pp. 1942-1945, 1995.

[16] G. G. Syswerda, “A Study of Reproduction in Generational and Steady-state Genetic Algorithms“, Genetic Algorithms, San Mateo: Morgan Kaufmann Publ. Inc., 1991

[17] S. Watanabe, T. Hiroyasu and M. Miki, “LCGA: Local Cultivation Genetic Algorithm for Multi-Objective Optimization Problems", *Proc. Genet. Evol. Comp. Conf*., pp. 762-763, 2002.

[18] E. Bonabeau, M. Dorigo and G. Theraulaz, “Swarm Intelligence: From Natural to Artificial Systems“, New York: Oxford University Press, 1999.

[19] M. Dorigo and T. Stutzle, “Ant Colony Optimization“, Bradford Books, The MIT Press, 2004.

[20] J. Branke, "Evolutionary Optimization in Dynamic Environments", Vol. 3 of the Book Series on Genetic Algorithms and Evolutionary Computation, Kluwer Academic Publ. 2001.

[21] J. Huidong, L. K. Sak, and W. M. Leung, “Genetic-guided Model-based Clustering Algorithms" *Proc. 2001 Int. Conf. Art. Intel*., vol. 2, pp. 653-659, 2001.

[22] M. Lozano, F. Herrera, N. Krasnogor, and D. Molina, "Real-Coded Memetic Algorithms with Crossover Hill-Climbing", Evolutionary Computation, The MIT Press, Cambridge, Massachusetts, USA, vol. 12, no. 3, pp. 273-302, 2004.

[23] T. Hastie, R. Tibshirani and J. Friedman, “The Elements of Statistical Learning: Data Mining, Inference, and Prediction“, New York: Springer-Verlag, 2001.

[24] N. Kubota and K. Nishida, "The Role of Spiking Neurons for Visual Perception of A Partner Robot", *Proc. 2006 IEEE World Cong. Comp. Intel*., pp. 530-537, 2006.

[25] N. Kubota and K. Nishida, “Cooperative Perceptual Systems for Partner Robots Based on Sensor Network”, *Int. J. Comp. Sci. and Netw. Security*, Vol. 6, No. 11, pp. 19-28, 2006.

[26] N. Kubota, “Visual Perception and Reproduction for Imitative Learning of A Partner Robot“, *WSEAS Trans. Signal Processing*, vol. 2, no.5, pp. 726-731, 2006.

[27] N. Kubota, “Computational Intelligence for Structured Learning of A Partner Robot Based on Imitation“, *Info. Sci.*, vol. 171, no. 4, pp. 403-429, 2005.

[28] A. Sulistijono and N. Kubota, “Particle Swarm Intelligence Robot Vision for Multiple Human Tracking of A Partner Robot“, *Proc. Soc. Inst. Cont. Eng. Annual Conf.*, pp. 604-608, 2007.

[29] A. Sulistijono and N. Kubota, “Evolutionary Robot Vision and Particle Swarm Optimization for Multiple Human Heads Tracking of A Partner Robot“, *Proc. IEEE Cong. Evolutionary Comp*., pp. 1535-1541, 2007.

[30] N. Kubota, T. Ohinata, S. Wakisaka and H. Liu, “Extraction of Facial Landmarks for a Partner Robot Based on Evolutionary Computation”, Proc. Comp. Intelligence, Robotics and Autonomous Sys., Massey University, Palmerston North, New Zealand, November 28-30, pp. 44-49, 2007.

Received: December 11, 2007 Revised: March 20, 2008 Accepted: April 3, 2008