Proceedings of the 1996 IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems

Emergent Behavior Based Sensor Fusion for Robot Navigation System

Yasushi Nakauchi

Dept. of Mechanical Engineering
National Defense Academy
Yokosuka, 239, Japan

Yasuchika Mori

Dept. of Mechanical Engineering
National Defense Academy
Yokosuka, 239, Japan

1 Introduction

In order to utilize mobile robots in service fields, such as in hospitals, in offices and in houses, the robots should equip the functions for coexisting with human. The surrounding environments of such robots are shared by both robots and human. Thus, there is a danger that robots and human may physically collide with each other. Therefore, the robots must not injure the human. Human may also influence the shared environments. For example, when a robot is moving, human may move chairs or may put carton boxes on the floor. It causes the unexpected obstacles for robots. In general, obstacles are classified into two categories. One is the static obstacles and the other is dynamic ones. The static obstacles are the ones that are difficult to move and occupy the space eternally, such as walls, desks and bookshelves. On the other hand, the dynamic obstacles are the ones that may be moved and temporally occupy the space, such as human, other robots and chairs. Therefore, the robots should be able to avoid both of static and dynamic obstacles. Furthermore, if the robots stop their functions without being able to achieve given tasks, they won't be of any use. So it is required for the robots to be able to keep functioning until they achieve given tasks. To do so, the robots should have several methods to achieve given tasks and retry several times until they achieve the object. The required functions for human coexistent robots mentioned above are summarized as table 1.

Table 1: Required functions for human coexistent robots

a) A robot must not injure human.
b) A robot must adapt to unknown environments.
c) A robot must keep functioning.

To realize such a robot, in our laboratory, we are developing human like mobile robot named WM (Work Mate). WM is equipped with dual arms, mobile robot base and CCD stereo vision camera system as shown in figure 1. WM is also equipped with multiple sensors to detect environmental information. They are touch sensors, infrared sensors (IRSs), ultrasonic sensors (USSs) and CCD cameras. To utilize WM as a human coexistent robot, we need a suitable architecture to integrate these sensors.

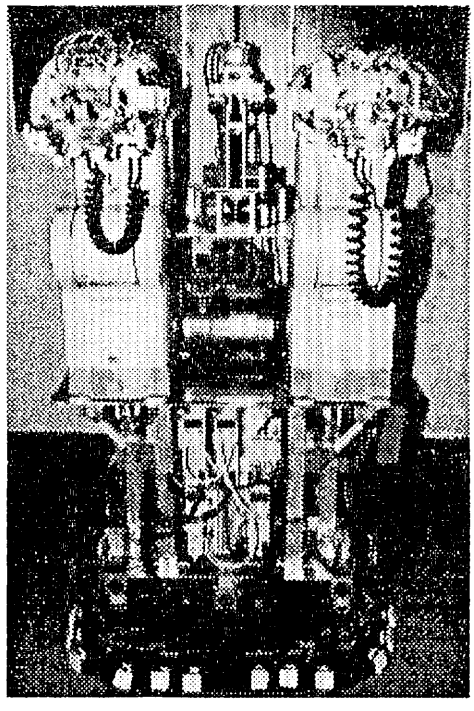

Figure 1: The out look of WM (Work Mate)

In this paper, a new architecture for constructing intelligent robots named EBBA (Emergent Behavior Based Architecture) is proposed. EBBA is constructed by behavior modules. In the framework of EBBA, the emergent behavior of the system can be controlled by a planner (one of a behavior modules). And sensor information are integrated at the level of behavior modules.

The rest of this paper is organized as follows. In section 2, we discuss the related works and the required functions for human coexistent robot architecture. Then in section 3, we propose EBBA. In section 4, we explain

0-7803-3700-X/96 $5.00

the robot navigation system based on EBBA, then in section 5, the implementation on WM is illustrated. And in section 6, the evaluation of the implemented system in various situation is showed. Finally, we conclude in section 7.

2 Mobile Robot Navigation System

The central issues of mobile robot navigation system are environmental map constructions and planning systems [1, 2]. But these approaches are insufficient to develop a robot utilized in human coexistent environment. In the real world, unexpected objects may exist. Ohya et al. proposed the dynamic map construction system by using ultra sonic sensors [3]. On the other hand, Pan et. al proposed moving obstacle avoidance algorithm [4]. But these methods take time to generate maps or are difficult to recognize arbitrary moving obstacles' positions and directions by using current technology [5]. Thus, by using this approach, it is difficult to keep the responsiveness of a system in the real world.

On the other hand, Brooks proposed behavior based architecture SSA [6]. This approach enables the system to adapt the dynamic environments. But in the framework of SSA, they are not allow to use planners, therefore, it is not suitable for achieving objective tasks.

Arkin proposed the hybrid system of SSA and a planner [7]. We also have developed mobile the robot navigation system by using SSA and a planner [8]. These systems realized both adaptive behaviors and objective behaviors. But when we want to develop more complicated system, we have to prepare behavior modules for each function. Thus, the number of behavior modules will be immense. And this cause the difficulty for a planner to manage behavior modules.

Assume that there are two behavior modules A and B (here after we call BM_A and BM_B respectively) and BM_B is invoked in succession to BM_A. BM_A is the behavior module that stops a robot's wheel and then goes backward a little if something touches on sensors. And BM_B is the behavior module that moves a robot 50cm towards the 90 degree inclined direction where something touched (see table 2). Thus the resulting emergent behavior is detouring avoidance as shown in figure 2.

Assume we changed some parameters of BM_B, so that it moves a robot 50cm towards the 45 degree inclined direction where something touched. Then the resulting emergent behavior is slipping through avoidance as shown in figure 2.

Furthermore, assume we changed some parameters of BM_A and BM_B as follows. BM_A moves a robot forward in stead of backward after it stops the robot. And BM_B moves a robot 50cm towards the direction where something touched. Then the resulting emergent behavior is the pushing avoidance as shown in figure 2.

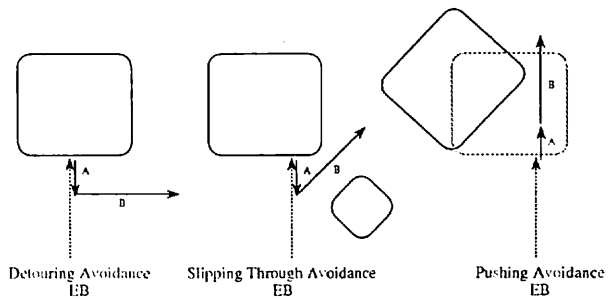

Figure 2: The variety of emergent behaviors by different set of parameters

In this way, by just changing parameters of behavior modules, we can realize many kinds of emergent behaviors. Therefore, if the planner can modify these parameters explicitly, it will be able to control many kinds of emergent behaviors without adding behavior modules. But in the framework of SSA and the previous works, theses parameters are not explicitly expressed and utilized. In next section, we propose EBBA, in which a planner can utilize these parameters explicitly and control emergent behaviors of a system.

3 EBBA

EBBA is designed to realize required functions for human coexistent robots as shown in table 1. At first, to realize a robot that doesn't injure human, a system must be sensor based architecture by which obstacles and human are detected without failure. Seconderly, to adapt to unknown environments, the system should respond to the changes of the environment in realtime. Therefore, the system must be constructed by a set of concurrently executable behavior modules. Thirdly, in order the system to keep functioning, it should have several methods to achieve given tasks. Also these methods must be controllable by a planner, so that it can emerge behaviors along with the situation.

These required functions of EBBA is summarized as shown in table 3. We have designed EBBA based on these requirements.

3.1 Behavior Module of EBBA

EBBA is constructed by behavior modules as shown in figure 3. Each behavior module is concurrently executable entity which has its own object state (OS) and internal states (ISs) as internal memory. An OS is the state that represent the current object of a behavior module. ISs are the parameters that effect the action of a behavior module. Each behavior module refers not only sensor inputs but also its own and other modules' OSs

Table 2: Emergent behaviors derived by same behavior modules

	Emergent Behavior 1	Emergent Behavior 2	Emergent Behavior 3
Behavior Module A	If something touchs on sensors, it stops and goes backward a little.	If something touchs on sensors, it stops and goes backward a little.	If something touchs on sensors, it stops and goes forward a little.
Behavior Module B	It moves 50cm towards the 90 degree inclined direction where something touched.	It moves 50cm towards the 45 degree inclined direction where something touched.	It moves 50cm towards the direction where something touched.
Emergent Behavior	Detouring Avoidance	Slipping Through Avoidance	Pushing Avoidance

Table 3: The basic ideas for designing EBBA

i)	Sensor based architecture
ii)	Constructed by a set of concurrently executable behavior modules
iii)	To have multiple methods to achieve given tasks by utilizing behavior modules
iv)	A planner can control emergent behaviors

and ISs, then decides its actions. As the resulting actions, each behavior module can modify its own OS and ISs, can modify other behavior modules' OSs, and can output orders to actuators.

Even if the same information is detected from sensors, the behavior of each module can be different depending on its OS and ISs. These actions derived by the interaction with environments are viewed as emergent behaviors. In order to control emergent behaviors of a system, a planner modifies other behavior modules' OSs.

3.2 System Control Architecture of EBBA

A typical system control architecture based on EBBA is the one as shown in figure 4. Within the framework of EBBA, a planner is also realized as a behavior module. A planner behavior module is different from others in the way it inputs other module's OSs as its sensor inputs. Thus, a planner can monitor the system's emergent behavior. If it is necessary, a planner can control the system's emergent behavior by just modifying other behavior modules' OSs.

Furthermore, with EBBA, we can construct meta-planners that control planners. A planner also has its own OS and ISs as shown in figure 3. Therefore, meta-planners are realized by just connecting their sensor inputs to other planners' OSs and ISs. This architecture is suitable for constructing more complicated and larger scaled system. For example, we can construct planners for each of robot hand, mobile robot base and eye system, then we can integrate them by using a meta-planner that controls these planners. In this way, we can expand a system very easily.

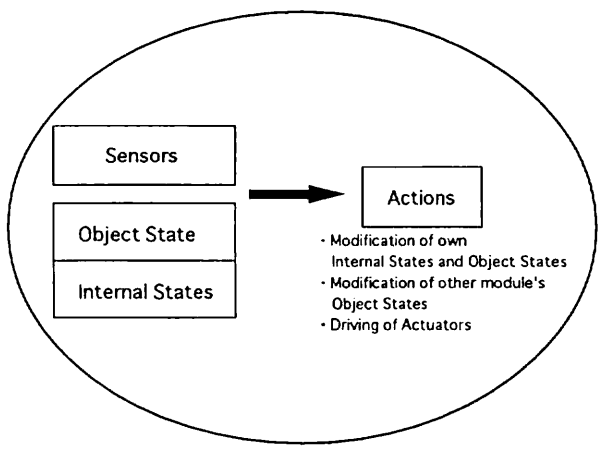

Figure 3: A behavior module of EBBA

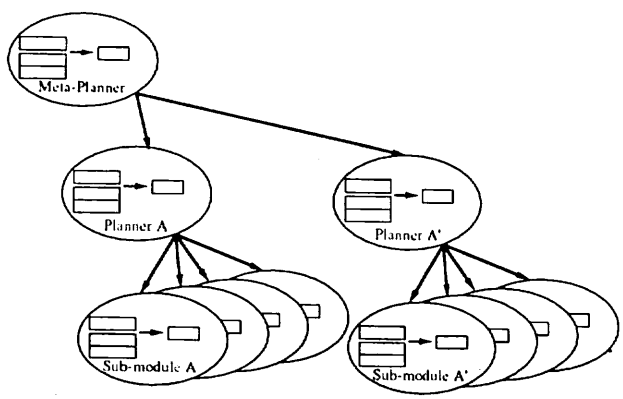

Figure 4: A typical system control architecture based on EBBA

A behavior module we have described in 3.1 is similar to the one in SSA, in the way it acts by referring sensor

inputs. But it is different in EBBA as to that the actions of each behavior module differ by its OS. Thus, a planner can control emergent behavior of the system by modifying the lower level behavior modules' OSs.

The system architecture of EBBA is rather simple. A system designer prepares necessary behavior modules, then connect reference lines to other modules' OSs and ISs from one module to another. If there are multiple modification lines go into one module, the priorities of the lines can be set. These priorities are depicted as subsuming line in figure 3.

4 System Configuration

4.1 Mobile Robot Navigation System Based on EBBA

We have designed mobile robot navigation system based on EBBA as shown in figure 5. The OSs that each behavior module may take are shown in table 4. The arrows in figure 5 denote the read/write lines to other modules' OSs and ISs. Each behavior module is illustrated as a layer. In the lower level (level 0, 1, 2), we arranged basic modules that must be realized to avoid system failure. On the other hand, in the heigher level (level 5, 6), we arranged the modules to achieve given goals. These heigher level modules refer and control the lower level modules' OSs and ISs. The behaviors of each module are as follows.

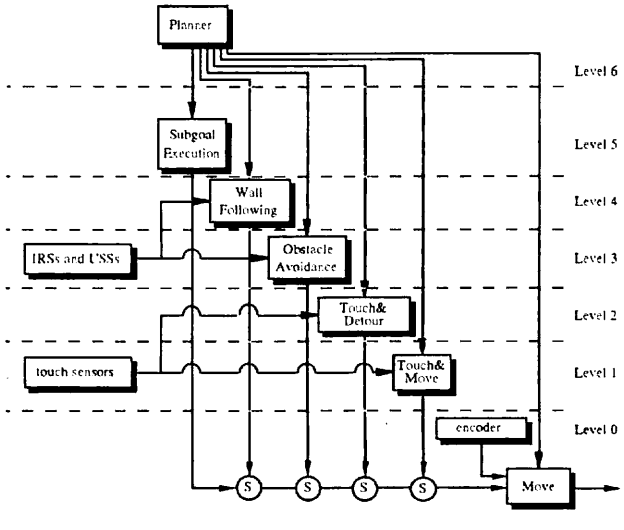

Figure 5: Control architecture of mobile robot navigation system

Level 0: Move

This behavior module drives wheel actuators by referring motor's encoders. This module has four kinds of OSs, one for each action a robot can take. It also has a current position data as IS, so that other modules can refer.

Level 1: Touch & Move

This behavior module (BM_A in table 2) refers touch sensors. If something has detected by the touch sensors and its OS is "stop & move backward", it stops a robot and then avoids contacting situation by moving backward 10cm. On the other hand, if its OS is "stop & move forward", it moves a robot forward (towards obstacles) 10cm and then move backward 5cm. By moving backward 5cm, the module can recognize if the obstacle can be pushed or not. And this information is expressed in its ISs so that other module can refer.

Level 2: Touch & Detour

This behavior module (BM_B in table 2) is invoked in succession to level 1. It refers inputs from touch sensors and its own OS. Depending on the OS, it moves a robot 50cm towards 90 degree inclined direction where something has touched, moves a robot 50cm towards 45 degree inclined direction where something has touched, or moves a robot 50cm forward.

By the combination of level 1 and 2, the emergent behaviors shown in figure 2 are emerged.

Level 3: Obstacle Avoidance

This behavior module avoids obstacle by using IRSs and USSs. This module is activated when something is detected at the front of a robot. If the OS is "detour", it moves a robot towards 90 degree inclined direction where something has detected. On the other hand, if the OS is " slipping through", it moves a robot towards 45 degree inclined direction where something has detected. The robot keeps moving until it detects nothing on sensors.

Level 4: Wall Following

This behavior module lets a robot follow walls by using IRSs and USSs. This module is activated when something is detected at the side of a robot. If the OS is "turn to parallel to wall", it turns a robot so that a side of the robot is parallel to the wall. On the other hand, if the OS is "move by following wall", it moves a robot forward by keeping a constant distance from the wall. In order to follow a wall, the former behavior is invoked first, then the latter one next. This module changes its own OS by itself so that the wall following behavior is emerged.

Level 5: Subgoal Execution

This behavior module is the one to achieve a subgoal. Subgoals are expressed as a small segment of goals, which can be realized by simple actions. If the OS is "turn towards subgoal", it turns a robot

to face a subgoal position. On the other hand, if the OS is "move towards subgoal", it moves a robot forward at the distance from the current position to the current subgoal position.

Level 6: Planner

The planner has a static environmental map and makes plans by decomposing a goal to subgoals. Then it modifies the OS of the subgoal execution behavior module to make an order.

The planner has links to all other behavior modules so that it can monitor how well the robot is achieving the current subgoal or avoiding obstacles. By referring these situations, the planner changes emergent behaviors of a robot by modifying other modules' OSs. After the planner achieves a goal, it modifys its OS as "idle".

Table 4: Object state of each behavior module

Level	Behavior Modules	OS (Object State)
0	Move	move forward
		move backward
		left turn
		right turn
1	Touch & Move	stop & move backward
		stop & move forward
2	Touch & Detour	move towards 90 degree inclined direction
		move towards 45 degree inclined direction
		move forward
3	Obstacle Avoidance	detour
		slipping through
4	Wall Following	turn to parallel to wall
		move by following wall
5	Subgoal Execution	turn towards subgoal
		move towards subgoal
6	Planner	goal
		idle

4.2 Sensor Based Emergent Behavior

Assume that when a human walks in a half dark room where carton boxes are on the floor. If the human can see the boxes he/she walks around them before he/she hits them. Or if the human can see the path is narrowed by the boxes, he/she passes through them without hitting them. But if the room is too dark to recognize them, he/she slowly walks toward the direction where he/she wants to go. And if the boxes touch on his/her legs, he/she may avoid them by detouring side way. Or he/she may try to slipping through them by walking diagonal direction. Or if there are too many boxes and it seems to take time by detouring them, he/she may kick them to make a path through. We can realize these behaviors which a human may take by using the architecture described in 4.1.

The former avoidance behaviors are realized by level 3 and 4 as follows. If a box in front of a robot is detected by IRs and USSs, the obstacle avoidance behavior module is activated. Then the robot moves around the box by changing directions. And if a box is detected at the side of a robot, the wall following behavior module is activated. Then the robot moves along with the box.

On the other hand, the latter three kinds of avoidance behaviors are realized by level 1 and 2 as follows. These behaviors are emerged by setting each module's OS as shown in table 5. The planner watches these situations, by referring these modules' OSs ans ISs. When the robot is avoiding the box by using the detouring avoidance emergent behavior and it is taking a time, the planner can recognize it by reading number of retries from the IS of the touch & move behavior module. In that case, the planner can change emergent behavior to the pushing through avoidance.

Table 5: The emergent behaviors by the different set of object states

Emergent Behavior	OS (Object State)	
	level 1	level 2
Detouring Avoidance	stop & move backward	move towards 90 degree inclined direction
Slipping Through Avoidance	stop & move backward	move towards 45 degree inclined direction
Pushing Through Avoidance	stop & move forward	move forward

5 Implementation on WM

We have implemented the mobile robot navigation system on WM. The hardware structure of WM's mobile robot base is as shown in figure 6.

We have employed TRC Labmate as the mobile robot base. Labmate has four free wheels and two DC servo motors. Each servo motor has an encoder that can measure the relative position. We have arranged 24 touch sensors (each of three are grouped by daisy chaining), 8 IRSs and 8 USSs as shown in figure 7. As a main computer, we used VME CPU board AVME-130 on which the realtime OS VxWorks runs. And via AVME-350 serial I/F board and a parallel port of AVME-130, the all sensors and the servo motors are connected. We also used SUN SS5 for developing and downloading programs.

We have implemented the system control architecture described in section 4 as follows. Each behavior mod-

ule of EBBA is written as a function of C program. And each function is implemented as a concurrently executable task of VxWorks. Therefore, by using the realtime scheduling scheme of VxWorks, we insured the sampling time of sensors.

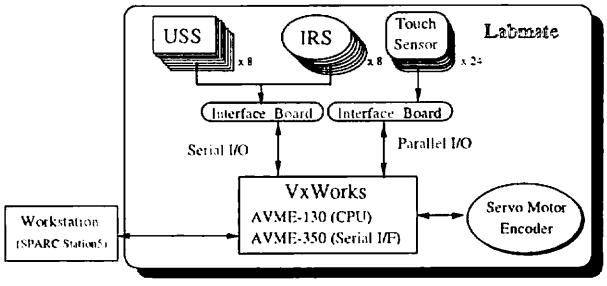

Figure 6: Hardware structure of the mobile robot base

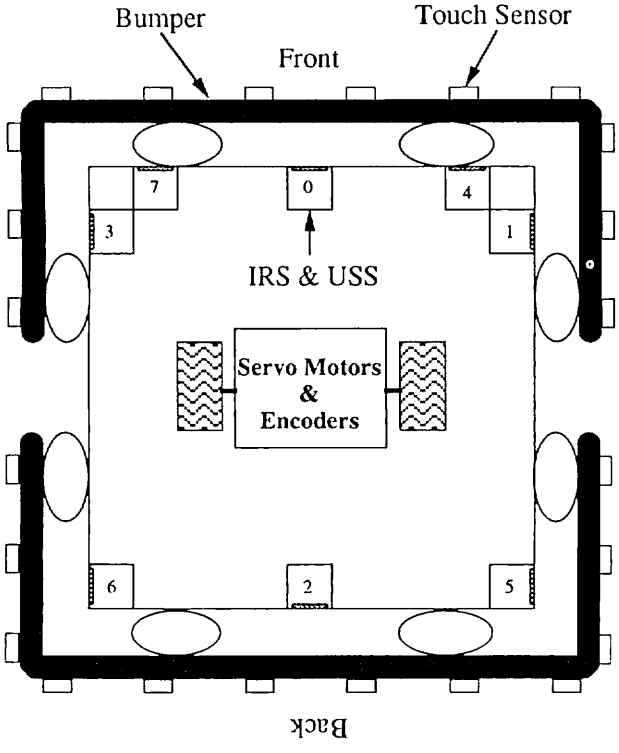

Figure 7: Sensors' arrangement of the mobile robot base

6 Evaluation

6.1 Experiment

To confirm the efficiency of EBBA, we have evaluated the system in the various situations. In the following experimental results, the planner knows environmental map beforehand and is given a goal position. In figures, the start position of the robot is the origin (0, 0) and the goal position is 3000mm forward to the start position (0, 3000). The rectangles in the figures denote obstacles. On the other hand, the dotted rectangles denote the original position of the obstacles which have pushed aside.

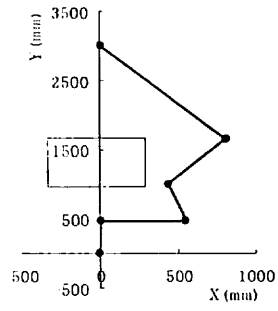

Figure 8: Obstacle avoidance emergent behavior

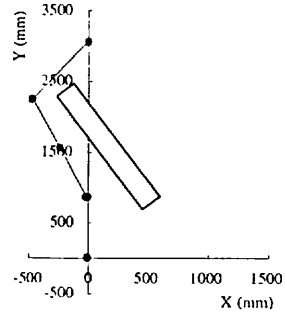

Figure 9: Wall following emergent behavior

6.1.1 Obstacle Avoidance Emergent Behavior

In order to evaluate the obstacle avoidance emergent behavior derived by level 3, we have put a box on the way to the goal. In this case, the planner initialized level 3's OS as "turn to parallel to wall" beforehand. The experimental result is shown in figure 8.

As shown in figure, when the robot detected the obstacle by IRSs, it turned the robot 90 degree inclined direction and moved forward until to move the place where nothing has detected on sensors. Then the subgoal execution behavior module was activated and it turned the robot towards the goal. But it detected the obstacle again. By iterating these behaviors again, finally the robot succeeded to reach the goal.

6.1.2 Wall Following Emergent Behavior

In order to evaluate the wall following emergent behavior derived by level 4, we have put a box on the way to the goal as it is inclined to the robot. The planner initialized level 4's OS as "turn to parallel to wall" beforehand. The experimental result is shown in figure 9.

As shown in figure, when the robot detected the obstacle by IRSs, it turned the robot so that the side of the robot faced to the obstacle. Then it changes its own OS to "move by following wall". Thus it moved the robot forward by keeping the distance from the obstacle. The robot stopped at the place where nothing has detected on sensors. Then the subgoal execution behavior module was activated and it turned the robot towards the goal. And finally the robot succeeded to reach the goal.

6.1.3 Detouring Avoidance Emergent Behavior

In order to evaluate the detouring avoidance emergent behavior derived by level 1 and 2, we have put a box

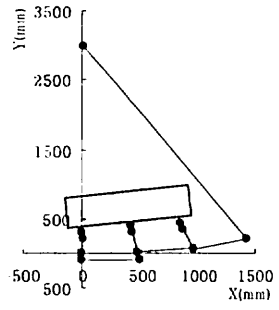

Figure 10: Detouring avoidance emergent behavior

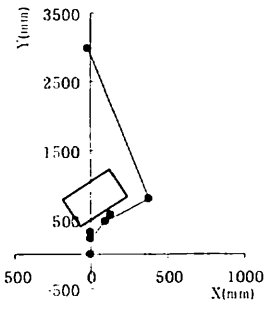

Figure 11: Slipping through avoidance emergent behavior

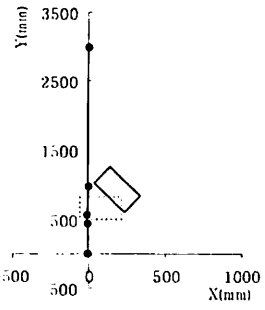

Figure 12: Pushing avoidance emergent behavior

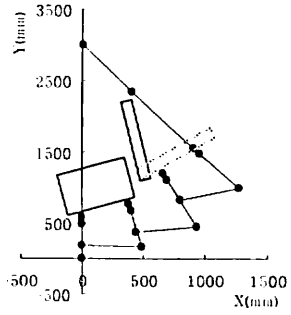

Figure 13: Experiment in more complicated situation

on the way to the goal. This box is lower height than previous ones, so it can not be detected by IRSs and USSs (we used this type of box hereafter). And we have forced the planner to set OSs of level 1 and 2 as "stop & move backward" and "move towards 90 degree inclined direction" respectively. The experimental result is shown in figure 10.

As shown in figure, when the robot touched the obstacle, it backs the robot 10cm. Then it moved the robot 50cm towards 90 degree inclined direction where the box has detected. After that, the subgoal execution behavior module has activated. The module turned and moved the robot towards the goal position, but the robot touched the same obstacle again. By iterating this behavior three times, finally the robot detoured the obstacle and succeeded to reach the goal.

6.1.4 Slipping Through Avoidance Emergent Behavior

In order to evaluate the slipping through avoidance emergent behavior derived by level 1 and 2, we have put a box on the way to the goal. And we have forced the planner to set OSs of level 1 and 2 as "stop & move backward" and "move towards 45 degree inclined direction" respectively. The experimental result is shown in figure 11.

As shown in figure, when the robot touched the obstacle, it backs the robot 10cm. Then it moved the robot 50cm towards 45 degree inclined direction where the box has detected. After that, the goal execution behavior module is activated. The module turned and moved the robot towards goal position, but it touched the same obstacle again. By iterating this again, finally the robot slipping through the obstacle and succeeded to reach the goal.

6.1.5 Pushing Avoidance Emergent Behavior

In order to evaluate the pushing avoidance emergent behavior derived by level 1 and 2, we have put a box on the way to the goal. And we have forced the planner to set OSs of level 1 and 2 as "stop & move forward" and "move forward" respectively. The experimental result is shown in figure 12.

As shown in figure, when the robot touched the obstacle, it moved the robot forwards 10cm, then moved back 5cm. The box was moved and this was recognized by touch sensors. So it moved forward the robot 50cm. Thus, the box was pushed aside and then the subgoal execution task has activated. It moved the robot towards the goal position, and finally the robot succeeded to reach the goal.

6.1.6 Experiment in More Complicated Situation

In order to evaluate the efficiency by controlling emergent behaviors derived by level 1 and 2, we have examined the system in more complicated situation. In this case, level 1 and 2's OSs are initialized as "stop & move backward" and "move towards 90 degree inclined direction" respectively. The experimental result is shown in figure 13.

As shown in figure, when the robot touched the obstacle, the detouring avoidance behavior was emerged. The robot tried this behavior three times. The planner watched this situation by referring the counter IS of level 2, then decided to change the emergent behavior to the pushing avoidance. Thus, the fourth time the robot touched the obstacle, it pushed aside the obstacle. Then, the subgoal execution behavior module has activated and succeeded to reach the goal.

6.2 Discussion

We also have examined the WM where humans are standing in front of the robot. When WM detected the human by IRSs or USSs, it detoured the human as well as shown in figure 8. But since the human's legs are rather slim, it sometimes failed to detect them by IRSs or USSs. In those cases, it detected humans by touch sensors, then it stopped and moved backward to avoid

human. We also tried to pushing WM by touch sensors, then WM continuously moved backward until the human quits it.

From the experimental results, we have confirmed that the mobile robot navigation system based on EBBA can adapt to the human coexisting environment by utilizing three kinds of sensors. Also we have confirmed that the planner can improve the efficiency of the system by selecting and controlling the behavior modules.

In some cases, it is too difficult to get to the subgoal position because of too many obstacles, or the subgoal position is occupyed by obstacles. The planner can recognize these situations by referring lower levels' modules. Then the planner also can cancel the current subgoal and set the current subgoal to the next one. In this way, we can construct flexible system. This is one of the major features of EBBA.

In the framework of EBBA, the flows of the sensor information are viewed as follows. The patterns of information detected by sensors are associated with the actions in each behavior module. And each behavior module represents raw and abstracted sensor information as its ISs. Also each behavior module represents its internal states (i.e. retrying number, activation status) as the ISs. Thus, the other module can refer them, if necessary. Especially, a planner refers those information for deciding its actions. And the planner represents more abstracted information of environments as its ISs. Finally, the meta-planner refers these abstracted informations for controlling planners.

This flow of information abstruction seems similar to the one taken in traditional AI. But it is different in the way it realizes both behavior based actions and sensor fusioning at the same time. We summarize the characteristics of EBBA in follows.

Reliability
Fundamental functions required for a robot are realized as lower level of behavior modules.

Parallel Architecture
Each behavior module can be easily implemented on a parallel hardware architecture.

Robustness
A system can have redundant behavior modules.

Extendability
A system can be easily expanded by adding behavior modules.

Responsibility
A planner is also realized as a sensor based light weight process.

Reusability
By using the same set of behavior modules, many kinds of emergent behaviors are realized.

Manageability
A planner can easily control a system's emergent behavior as well as other behavior modules.

7 Conclusion

In this paper, we proposed EBBA the new architecture that fusions heterogeneous sensor information at the level of behavior modules. And we have designed and implemented the mobile robot navigation system based on EBBA. By experiment, we have confirmed that the system based on EBBA satisfies the required functions for human coexistent robots.

We are currently developing the dual arm robot system and the stereo vision system of WM by using EBBA. We are planning to integrate them as one service robot in near future. At that time the real worth of EBBA will be examined.

References

[1] R. Alami and T. Simeon, "Planning robust motion strategies for a mobile robot", *Proc. IEEE ICRA*, Vol.2, pp.1312–1318, 1994.

[2] A.D. Luca and G. Oriolo, "Local Incremental Planning for Nonholonomic Mobile Robots", *Proc. IEEE ICRA*, Vol.1, pp.104–110, 1994.

[3] A. Ohya, Y. Nagashima and S. Yuta, "Exploring Unknown Environment and Map Construction Using Ultrasonic Sensing of Normal Direction of Walls", *IEEE ICRA*, pp.485–492, 1994.

[4] T.J. Pan and R.C. Luo, "A Feasible Collision-Detection Algorithm for Mobile Robot Planning with Moving Obstacles", *IECON'91*, pp.1011–1016, 1991.

[5] P.K. Allen et al., "Autometed Tracking and Grasping of a Moving Object with a Robotic Hand-Eye System", *IEEE Trans. on Robotics and Automation*, Vol.9, No.2, pp.152–165, 1993.

[6] R.A. Brooks, "A Layered Intelligent Control System for a Mobile Robot", *IEEE Journal Robotica and Automation*, Vol.RA-2, pp.14-23. 1986.

[7] R.C. Arkin, "Integrating Behavioral, Perceptual and World Knowledge in Reactive Navigation", *In P. Maes eds. Designing Autonomous Agents*, pp.105–122, MIT Press, 1990.

[8] Y. Nishida, Y. Nakauchi and Y. Mori, "Development of autonoumous mobile robot by using SSA", *IEEJ*, IIC-95-56, pp.51–60, 1995 (In Japanese).