1998 International Computer Symposium
Workshop on Artificial Intelligence
December 17-19, 1998, N.C.K.U., Tainan, Taiwan, R.O.C.

AN AGENT-BASED INTELLIGENT INTERFACE FOR MOBILE ROBOT CONTROL *

Alan Liu
aliu@ee.ccu.edu.tw

Yih-Wen Lee
m85034@ee10-2.ee.ccu.edu.tw

Jia-Houng Shyu
g8542027@ccunix.ccu.edu.tw

Department of Electrical Engineering
National Chung Cheng University
160 San-Hsin, Min-Hsiung, 621, Taiwan

Abstract

The system introduced in this paper is an application of designing intelligent human-computer interface agents for a mobile robot system. The goal of our intelligent interface system and framework is to offer a different interface which is tailored for each user. By an intelligent interface, we focus on the features which can assist the user, learn some habits or actions from the user and do some jobs which the user takes, and even fit the user's preferences. Our multi-agent structure of the intelligent interface agent consists of ten small agents cooperating to process the user requests. We also present some of the intelligent strategies for constructing our intelligent interface agent.

1 Introduction

Intelligent agents may employ some degree of learning of adaptation to improve the quality of its assistance over time. They can be put together and do complex tasks or work alone for simple tasks. Our goal in this paper is to introduce a multi-agent architecture for building an intelligent interface system. An IHCI (Intelligent Human-Computer Interface) can be aided by an intelligent interface agent, which consists of different small agents. An ordinary user interface has its basic functions to control its system, but by introducing intelligent agents, it becomes more functional in assisting users because its intelligent strategies are capable of helping users to do their jobs.

The applications for IHCI agents include intelligent tutoring systems [1], calendar management system [2], mail agent system [3] etc. We have chosen a robot control system for our application. Many researches in robot control interface concentrate on stability of control and goal achievement. They neglect the control interface is designed for users. We take the user-centered design into our design structure. This IHCI has the control stability, goal achievement, and user-aided functions. We propose a framework and a method for designing an IHCI agent.

From our IHCI agent method and framework, the agent can offer a different interface which is tailored for each user. It can turn the passive interface into a more active interface. Using the techniques of AI enables the learning ability of interface. By this approach, the interface becomes not only a communication media but also an advisor, teacher, director, and agent. Advantages include increasing the ability of manipulating by itself, helping users dealing tasks rapidly and effectively, providing an active interface, and accepting more variable instructions. These advantages will conduce to users for operating the interface more handily. The individual agents make themselves responsible for their works. Any agent can work independently and concentrate on its tasks.

In the next section, we also take a look at the proposed intelligent interface agent architecture and relations with applications. Section 3 proposes the intelligent strategies and agents cooperation, and explains our work and propose design methods. A sample application is given in Section 4. Finally, we suggest our future works and conclusion.

2 Intelligent Interface Agent

An IHCI becomes a complicated system when adding intelligence and adaptation to it. The application cannot expect what the user's next action or what degree he understands and familiarizes this system. We propose an agent architecture to solve the complex system designing and adding intelligence and adaptation in it. Using agents with different ability offers a multi-agent environment to process many problems and improves the efficiency of performing tasks.

*This research was supported by the National Science Council under grant NSC 87-2212-E-194-003.

2.1 Proposed Multi-agent Structure

Our multi-agent structure of intelligent interface agent consists of Interface supervisory agent, Active dialog agent, Interaction monitoring agent, Advice agent, User model agent, Self-Scheduling agent, Error detection agent, Explanation agent, Instruction processing agent and Application control agent. The system structure is shown in Fig. 1. The intelligent interface is designed separately from the application and the user interacts with the agent which substitutes the jobs of the communication and control to application for autonomous takeover by itself.

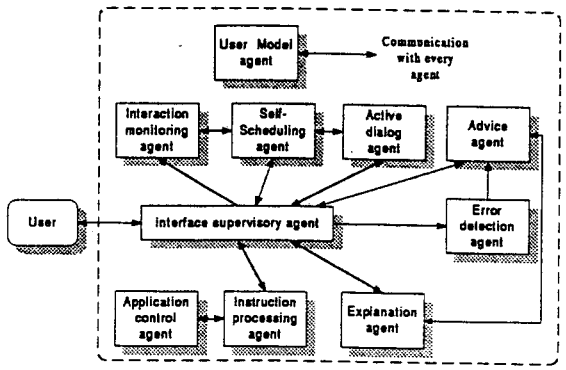

Figure 1: Intelligent user interface agent architecture

The interface supervisory agent manages all other agents' events. Our multi-agent system also uses the user model for adapting to users. The self-scheduling agent schedules some tasks which should be needed for users and executes them automatically. The active dialog agent chooses the dialogues representation for users and deals some dialogues from users inputs. Interaction monitoring agent gathers all information when users interact with agents. The advice agent gives users some hints for any aspect. These agents work with the user model agent that manages the user model. The error detection agent observes the mistakes that users make and corrects it or takes it to the advice agent. The explanation agent serves users when they require the help, query some questions or the advice agent requests for some accurate hints. The instruction processing agent deals with users' instructions or some actions which are associated to application. The application control agent deals with application, and it is like a gateway to application. It takes the messages or the commands from instruction processing agent and notifies the application for proper actions. Agents communicate with the user and they have collaboration and communication with each other. These construct our multi-agent intelligent user interface architecture.

The interface supervisory agent is the chief of all agents. Users interact with it for their goals, purposes, and even suspicions. It will take the jobs or tasks and assign to agents which can take it and complete it. Agents can also activate the other agents for achieve some goals or they can request the interface supervisory agent for some requirement to supplement the missing information or functions which are the condition of finishing the goal. For example, the error detection agent can activate the advice agent when it detects the user's mistakes. And the advice agent sends a request to the explanation agent, asking for proper answer or explanation. The advice agent sends back data to determinate the better advice according to the user model agent. The actions between agents are not only passive but active. The agent can request others for some events, some suggestions or some information. It can decide to accept or reject information which is from others and find the best way to achieve the user's goal. All agents have the ability of learning and adaptation. Every agent can work according to the user model which contains user's information observed from interactions or inferences and also its own knowledge base.

Most agents in the system have a learning ability. The agent can get information from the user and integrate it to some useful knowledge. Over a period time, agents will have rich information about dealing the related tasks or we can say that they have the experience in processing these jobs. Each agent provides its information for other agents to use in the task processing. The collaborative relation builds on the information sharing agent community. Learning is an adaptive process for the user. For users, the adaptive system may reduce the user's work or their time because the jobs are taken by the system. It also makes the user to feel that the system is easy to use. Manipulating the system is a hard task for some people who is not familiar with using computer or the system. The learning or adaptive property increases the usability of the system, and the conveniences that it brings are ineffaceable.

2.2 Single Agent Component

The single agent could be thought as a small version of the whole multi-agent system. It can work independently without other's help or collaboration with others. We can add an extra agent for extra functions for the multi-agent community without affecting the agents' facilities. In order to make the whole system concurrent and consistent, our single agent structure follows some common properties. Their structures and

behaviors must consort with each other. The protocols between the agents have to be clearly stated.

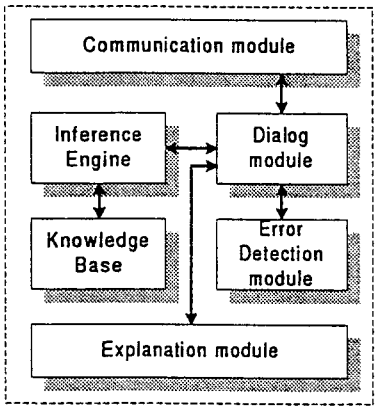

Figure 2: Basic component of a single agent

Our basic agent component has these parts: the communication module and the knowledge-based system which includes the inference engine, dialog module, explanation module and error detection module. These modules are the basic items of the single agent shown in Fig. 2. Every agent may have specific modules for extra functional requirements besides these basic modules. The common agent that consists of the basic modules has the basic abilities of finishing the tasks that users assign or other agents require. Some tasks are about the domain knowledge, error detection inside the agent, explanation for the agent's domain and communication with other agents. The knowledge base contains something about this domain knowledge. By using the inference, the agent can retrieve and maintain the knowledge base. The communication module is used to collaborate with other agents and to send and receive the data or messages. The protocol of the agent communication is made through the communication module. The dialog module receives the messages from the communication module and processes it. The explanation module explains the tasks that are inside its own domain when it processes. It is also activated by the request of user or other agents. The error detection module finds out the errors occurred in this agent and tries to recover or report to the required agent.

3 Strategies and Agents Cooperation

The intelligent strategies used in our system include a knowledge base, a user model, and a dynamic assistance. We use a rule base with forward chaining to implement our knowledge base system. The user model is another topic of our intelligent strategies. Building the user model will be helpful for adapting the user when the user uses the interface [4]. It also reduces the time for interaction with the user if the user has used the system before. The user model makes the intelligent system easy to manipulate whatever who the user is. The system will record the information about users and make the optimal environment for the user using it.

The system also provides an assistance feature for becoming more active and friendly that can help the user to accomplish their works quickly and effectively. The dynamic actions, advisory functions and automatic scheduling processes enable the system some intelligent behaviors. We present the methods of designing the intelligent interface agent by the intelligent strategies and researching some problems in the following sections.

3.1 Active Dialog System and Advisor

We design our strategy according to [5] and make the the advice and help be adapting to the specific users. The dialog in intelligent interface agent is active, not the same as the traditional passive dialog that uses the question-and-answered style. It can propose some question actively or give the user suggestions unconditionally. The characteristics of the active dialog are using some predictions or guesses in the system to happen the interaction with users. When the user manipulates the system, it will monitor his actions and check them. The user will be queried or suggested when the system detects something strange. The advice agent plays a role of the advisor who provides the on-line help and hints to the user who is a novice or a beginner. The user can accept or reject the agents' suggestions,

3.1.1 Active interaction

The agent learns knowledge from the user through the dialog. Different from the common active dialog system, our active dialog method has the properties of adaptive, dynamic and non-monotonic. The adaptive property is that the dialog will be presented differently for different users. This will depend on the information of the user model. The same problem could be presented by several ways and the adaptive dialog will find the optimal styles for the specific user. For the novice or beginner, a problem can be divided into several parts in order to make the user understand easily. But for the advanced user, it may tell him directly because his ability on this domain is very strong. Basically, our dialog will not separate the dialog intentionally at the beginning. It must realize (learn) these characteristics on users and information through a period of observational time.

The dynamic property is that no matter the system receives information of the user or not, it is not necessary idle when the user stops using system temporarily. The user might be thinking, might have left his seat, or might stop working and forget to quit the job. Dynamic (active) learning (dialog) is processing at any possible time. Thus, the predictions or guesses are important. We consider these facilities into our machine learning method. The anticipation and deductive reasoning make the system respond more actively. More learning methods produce more opportunities of information gathering and knowledge evolution. The system updates the knowledge base when it observes the new information.

3.1.2 Advising feature

The advice module gives some hints to the user. The help function in the common system is on-line help which is checked by the user manually. The on-line help usually provides a search environment for users. It is a convenient electrical book but the user must know what he wants and how to search. The improvement to the on-line help is the automatic error detection. Many intelligent interface systems embed this function [6]. It combines on-line monitoring process to enable the error detection and help action. This automatic help is convenient for the novice that uses the system. Users can directly communicate with computer using dialog and require the information they want. In requiring help information, it is more convenient than on-line help but if the system meets a passive user, it cannot do anything.

Our advice module is designed to take responsibility of a tutor and to assist users using the system. We take the consideration of the active hints production function into the advice module that makes system assisting users dynamically. For different kinds of users, we have different advisory strategies that depend on the user model. The active hints function provides passive users a gateway to guide them to use the system. If the user is a beginner, he can use the system by respecting the instructions of the advisor. Whenever he is using the system or stops to think how to do something, our advisor gives some suggestions for next step. For a passive user, active communication with users is a good method to assist users solving their problem. The suggestion that is proposed by the system is another choice to do their tasks. Although the suggestions are not necessary the best way or the preferable way to solving the problem, the user can refer to them or consider them as alternatives.

The user can overwrite or deny the suggestion that is proposed by the system. Too many suggestions may also disturb the user. The system can control the active advice occurrence by the user model but we cannot say exactly whether the user will like it or not. The advice can be disabled by the user when the user needs to think composedly. The system can get information from the user's responses and his opinions that can improve the performance of advice.

3.2 Automatic Scheduling Module

Our intelligent interface agent also plays the role of a secretary to the user. It offers the scheduling function for users. The common scheduling system is manual. The user must manage his schedules by himself, and the system will execute them automatically. Combining the active advice module, the system can propose some suggestions for users to choose. From learning users' preferences, the system will know how to schedule the tasks that can fit users' demand. Our system is responsible for scheduling and executing the routines that reduce the user's load. The system may give suggestions or offer automatic scheduling.

Scheduling suggestion is similar to the advice module that proposes the suggestions for the user to refer. It suggests various schedules for users to choose. According to some experiences from the users, the system can use the knowledge to permute some cases of scheduling. These cases will be shown to the user and give him some references about them. At initialization, the system does not know what schedule the user will like. It first assumes some suggestions as hints. The user can accept or deny it by his opinion.

The fully automatic scheduling is that the user gives the system full permission to schedule and let the system execute whatever it has decided to do. Different to the scheduling suggestion, it is autonomous ability in scheduling and executing. Its arranging methods are the same as the scheduling suggestion, but it has a confidence rate for arranging tasks that are used to decide whether the schedule can be executed without permissions. If the confidence rate is not enough to activate the automatic execution of the schedule, the system needs to get the permission from the user to have the privilege of implementing it. Those jobs of the automatic executing schedule are not run quietly. In order to make the user know that some jobs are executed by the automatic scheduling module, the system will tell the user while these jobs are running. This step is necessary because the user has the highest power to do their jobs, so that the user may cancel them. The automatic executing schedule can be canceled by the user before it completed it. The system also learns from these events that manipulate by users.

3.3 Agents Cooperation

An agent that receives a job will process it and require other agents' assistance when it needs the information outside its domain. They can call for assistance between the related domain agents. Another way is that agents communicate through the interface supervisory agent because the supervisory agent manages the task assignment. The supervisory agent not only plans the tasks from the user but also balances the coordination between agents. The supervisory collaboration is the main method of agent cooperation in our structure. It processes the major planning control and task assignment. The coordinative collaboration is used when an individual agent needs other agents' information or assistance while processing the sub-goals.

3.3.1 Supervisory collaboration

The supervisory collaboration manages the integrative events. It is supervised by the interface supervisory agent. The agent communicates with users directly and all users' demands will reach here at first. It will plan the goal and decide which agents are needed in finishing tasks. Agents that are assigned by the supervisor send the report of processing tasks to the supervisor and the supervisor knows the jobs are finished or not. All agents besides the interface supervisory agent cannot communicate with the user directly. The user can only tell his goal to the interface supervisory agent. It is like a project leader that handles the events and tasks of the whole agent group. And it is also a gateway from the user to agents. When an agent needs other agents' information, it must send a request to the interface supervisory agent for asking other agents' cooperation. The supervisor will forward the request to the target agent. After the target agent completes its work, it replies the message to the supervisor and the supervisor returns the message to the source agent. The source agent receives the message and knows its request has completed successfully.

The planning is done by the interface supervisory agent. The task planing module in the interface supervisory agent processes the requests of users and agents. This agent contains many types of planning style and information related to other agents' domain. It has the abilities of identifying types of tasks, classifying and arranging them. We have the functional classification interpreter inside the agent. It identifies the incoming tasks from their functional requirements. The supervisor assigns these tasks to related agents according to the identification of requirement. If the user's intention is not clear, the default assignment will be the active dialog agent because it can do some interactions and the work of the knowledge acquisition that can realize the user's intention. The supervisor needs to analyze the information or message from the user and the agent.

Generally, to identify the task that which agents should take is not difficulty. The interface supervisory agent utilizes the knowledge and rules stored in the knowledge base to plan the steps of achieving the goal and identify which agent should be activated. The planning table is created by the interface supervisory agent according to the planning interpreter There are many planning tables in the interface supervisory agent at the same time. The agent must manage the execution of these tables. Every event in the planning table can be set with priorities. It must keep the tasks executing consistently. Arrangement of the plan is important that can increase the efficiency of execution.

3.3.2 Coordinative collaboration

The coordinative collaboration is the characteristic of independence of agents [7, 8]. Every agent can call the related agents for requests no matter it needs the information or assistance. Different from the supervisory collaboration, agents can exchange the information privately. This can establish the strong link between agents and exhibits the agents' abilities. Generally, the supervisory collaboration processes and plans the large tasks like the user's task. The coordinative collaboration processes the small tasks and requirements. The task assigned by the supervisor may need other agents' help to accomplish. For example, if the user issues an erroneous commands, the system can observe it by the error detection agent. The error detection agent sends a request to the advice agent for giving advice to tell the user error occurs and how to correct it. The advice agent receives the message from the error detection agent and it needs to retrieve the information of a "how-to" solution from the explanation agent, so it sends a request to require the information from the explanation agent. Finally, the advice agent sends the result to the interface supervisory agent and it presents to the user.

3.3.3 Agent communication

The communication between agents has some common protocol. When one agent wants to transfer information to another, the form of data must be agreed by both sides. Any agent requires or replies some information should send a message from its communication module. The message has the formal structure that is easy to identify. The communication module decodes the receiving message and encodes the message that wants to be sent. Typically, the information flow has two kinds: sending and receiving. They are similar

to each other. The difference is the source and destination are just opposite. If a message transfers from agent A to agent B, the source agent is A and the destination is B. In contrast, the source agent may be B and the destination may be A. We define the formal frame for communication between agent. The frame is shown in Fig. 3.

Transmission Frame

Destination Agent	Source Agent	Requirement type	Data

Figure 3: Transmission frame structure

The transmission frame records the destination agent, source agent, requirement type and data. The destination agent records the agent where the message is sent to. The source agent records the agent where the message is from. This tag provides a return path for the receiving agent when it wants to reply the execution report. The requirement type has many different values. They are information, task and report, etc. The information requirement type indicates that the objective of the transmission frame is requiring or sending information. The data tag contains the information that wants to be transmitted. The task requirement type indicates that the objective of the transmission frame is delegating some tasks to finish. The data tag records the related data and information for doing tasks.

4 Sample Application

The intelligent wheelchair control system [9], our application for IHCI agent framework, consists of three layers: the control layer, the planning layer, and the interface layer (Fig. 4). Our system is to make the interface layer more intelligent and user-friendly toward the user. The patients using our interface may have different handicaps. Our agent is designed for helping those people to go to where they want to go.

Because the three layers are integrated, we also design the simulation environment to substitute the other two layers for our experiment. In order to design our intelligent interface agent, we propose constructing the knowledge base, rule-based inference engine, and the user model to implement the system. The user interface uses a graphic (visual) window that is friendly for users.

The objective of IHCI of our intelligent wheelchair offers patients a convenient environment and services such as moving around, reaching a specific location and assisting the user in controlling the wheelchair.

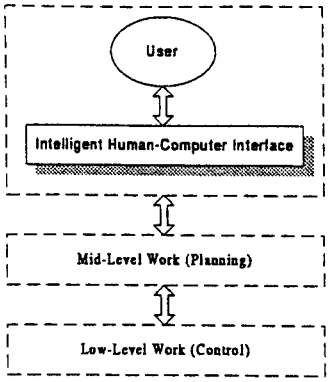

Figure 4: Intelligent wheelchair control system

Thus, the system must posses the autonomy and active property that helps the user to accomplish their goals. Adaptation, active processing, advice assistance, and automatic scheduling make users to reduce many hard and complex works while manipulating the wheelchair. The autonomy property also provides the heavily handicapped to control the wheelchair by themselves. The system must posses these abilities that assist patients:

- Realizing the patient where he wants to go and send a command to the wheelchair,
- Doing automatic scheduling of some tasks for the patient,
- Giving some hints or advice to the patient,
- Observing the patient's habits and adapting the patient by them,
- Having learning abilities for new expressions and tasks,
- Dealing emergency condition (i.e. unexpected accidents)

4.1 System Architecture

The components of the interface architecture are shown in Fig. 5. The input/output devices we consider are LCD, VR-glasses, VR-glove, and Microphone. However, in the initial test environment, we adopt the basic computer devices such as a mouse and a keyboard.

The system completes the user's request by agents cooperation. It is designed by multi-thread program to achieve the independence of agents, efficiency and autonomy. Although it cannot be accessed by others directly, many agents still want to refer the user model

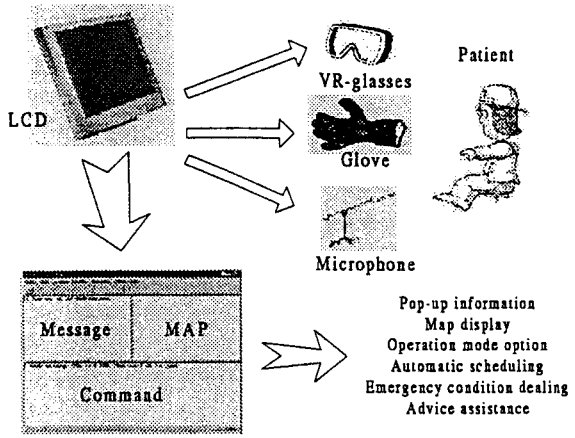

Figure 5: System overview

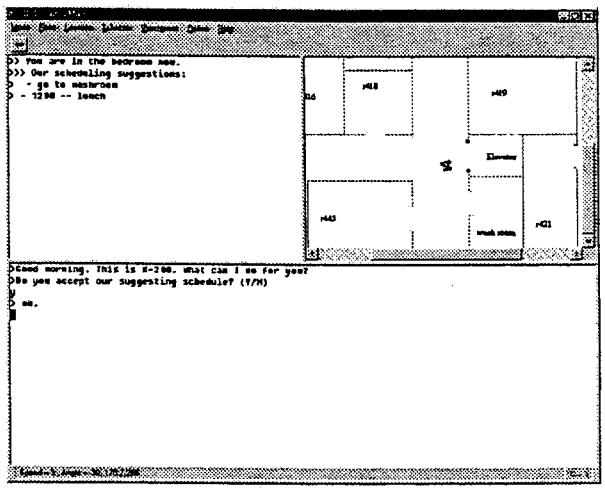

Figure 6: System interface

for their performances. We design the user model agent that can offer the call for the user model data to other agents. This means all agents can share and use the common user model and reduce the redundancy. The communications among agents depend on the protocol they agree.

The steps and processes of manipulating intelligent interface system are described as follows. First, the user needs to input the identification number (ID) to login the system. If he is a new user, it will ask some questions for initialization. After it prepares information for the user, the user can operate the interface, and the system also starts monitoring the operation. If the user is active, he sends the instruction to the interface and the instruction processing agent will take it and pass the data to the system. The advice agent can propose some advice for user to do something if the user does not know how to start. The self-scheduling agent arranges plans or schedules according to observing interactions and the user model. Interaction monitoring agent extracts information about the user's characteristics and sends to the user model agent to maintain the user model. The error detection agent and the advice agent can cooperate to guide or teach the user to recover the error in manipulating interface. The explanation agent gives the answer when the user asks about the system. The active dialog agent creates dialogs from interactions. Although the user does not do anything after login the system, the interface system is still running and responding actively. Giving advice and proposing schedule are executed automatically. A screen dump from our prototype system is shown in Fig. 6.

4.2 System Functions

The operation modes are manual, assistance and automatic. The manual mode is the primary function that uses the joystick as the control equipment, and the intelligent agent will not be active at this mode. The assistance mode provides the location selection, speed assignment, directions assignment, and some other forms of instructions. The patient can use these instructions to control the wheelchair accurately. For a heavily handicapped patient, the automatic mode is an intelligent control mode that can reduce the user's instructions. It offers flexible instructions, automatic scheduling, advice and hints, and some abstraction instructions. The system also provides the emergency handling because some patients are unstable. Some functions from the assistance mode and the automatic mode are described below.

Assistance mode By using the simple instruction, the wheelchair can take the user where he wants to go. The functions in assistance mode are explained below:

- Clickable map display — The map of the floor will be shown on a sub-window with the room numbers and other information for locations. The user can use the cursor to select where he wants to go.
- Message sub-window — The message sub-window is used to display the messages that the system wants to let the user know.
- Dialog sub-window — The window shows dialog to the user for interactions and accepts input instructions by the user. Sometimes it will propose

some questions to the user to acquire information about the user and realize the user's requirements.

- Place selection — The user can select a specific place where he wants to go by using the cursor to select places in map, by issuing the place in the dialog window, or by selecting from the menu.
- Detail operations — If a place is not a certain location (e.g. 2 meters from the door), the user can use the map clicking or directly input the distance that he wants to go (e.g. forward 10 meters). The speed can be also set, and the user can stop the wheelchair anytime he wants.
- Semantic instructions — Semantic instructions are not formal commands and they can be defined by users or learned from users.
- Basic on-line help — The system offers the on-line help for using system.

Automatic mode The automatic mode is active mode of the interface control system. In this mode, the interface agent will depend on the experiences and knowledge learned previously to make some proposals and decisions for the user. This mode provides functions as follows:

- Automatic scheduling — The system creates a schedule according to the characteristics of the user, which is learned by past experiences. The manual scheduling can be also made by the user to over-write the schedule proposed by the system.
- Advice proposal — The advice proposal can guide users to use the system. For ordinary users, it also has the active help and teaching functions.
- Personal instructions definition — When the system receives the new instruction that is never used before, it will start the interaction and require more information about the new instruction. The learning process is on-line.
- Emergency processing — The function has two settings. One is "call the doctor," which notifies the emergency center (when a wireless communication device is set up). The other one is the "setting place", which will make the wheelchair go to the specific places when the emergency occurs.

5 Conclusion

The agent cooperation has an important topic. When many agents work together, it must work with each other in harmony. Our structure is mainly a supervisory structure with a simple cooperation function. By increasing the ability of the interface supervisory agent, we can solve the conflict resolution problem and other agent cooperation problems. Besides these, our future works include combining some I/O devices (e.g. voice input, glove, and touch monitor, etc.) and technologies (mentioned above) into the intelligent interface agent to make more adaptive, intelligent, and friendly system.

References

[1] R. W. Chu, et al., "Using the operator function model and OFMspert as the basis for an intelligent tutoring system: Towards a tutor/aid paradigm for operators of supervisory control systems," *IEEE Transactions on System, Man, and Cybernetics*, vol. 25, pp. 1054–1075, July 1995.

[2] T. Mitchell, et al., "Experience with a learning personal assistant," *Communication of the ACM*, vol. 37, pp. 81–91, July 1994.

[3] T. R. Payne and P. Edwards, "Interface agents that learn: An investigation of learning issues in a mail agent interface," *Applied Artificial Intelligence*, vol. 11, no. 1, pp. 1–32, 1997.

[4] E. Averbukh, el al., "User-model based design of adaptive human-computer interfaces," in *IEEE Proceedings of International Conference on System, Man, and Cybernetics*, pp. 1693–1697, 1997.

[5] K. Hook, "Adaptation to the user's task," Tech. Rep. ISRN SICS-R-95-08-SE, Swedish Institute of CS, 1995.

[6] D. N. Chin, "KNOME: Modeling what the user knows in UC," in *User Models in Dialog Systems* (A. Kobsa and W. Wahlster, eds.), pp. 74–107, Springer-Verlag, 1989.

[7] K. Sycara, A. Pannu, M. Williamson, and D. Zeng, "Distributed intelligent agents," *IEEE Expert*, vol. 11, pp. 36–46, Dec. 1996.

[8] Y. Lashkari, M. Metral, and P. Maes, "Collaborative interface agents," in *Proceedings of AAAI '94 Conference*, 1994.

[9] R. C. Luo *et al.*, "Development of intelligent electrical wheelchair for hospital automation," in *Proceedings of International Conference on Mechatronic Technology (ICMT'98)*, Nov. 1998.