MOBILE INTELLIGENT AGENT WITH A MULTIMODAL INTERFACE

Group IMM971

Adam Cohen, Sergio Ortega, Søren Bach Christiansen, and Thomas Dorf Nielsen

Abstract: In this paper we propose that a multimodal intelligent interface would greatly improve the ease of interaction with a handheld device. To test this hypothesis we constructed a prototype that used spoken dialogue, personalised information and minimal graphics to implement a city guide. The system was then tested on the university campus by several users. The probabilistic user model successfully tailored the available information to each individual user and the users became familiar with the functionalities extremely quickly, requiring very little help. These results indicated that our prototype could be used as a base for further development. Finally, we present some advanced features that could be integrated into the system we have created.

Keywords: Decision Support System, Mobile Phones, Spoken Dialogue System, Global Positioning System, Multimodal User Interface, Intelligent Multimedia.

1. INTRODUCTION

Modern-day businessmen have to be in contact with their business and their data even when they are on the move. Many products are already on the market to meet this demand, from the Nokia 9000 Communicator™ and the Alcatel One Touch Com™ to the Palm III™ and other associated personal digital assistants. However, all these devices depend entirely on their graphical interfaces to communicate with the user. Since these devices have to be small enough to fit in a pocket, their interfaces necessarily become very complicated as numerous functionalities are squeezed into a small display.

We propose that similar features can be provided using minimal graphics combined with a spoken dialogue system and personalisation of the data, resulting in a much simpler interface. This paper represents our efforts to implement and evaluate such an interface which would make the life of both businessmen and tourists easier.

We chose to apply our ideas to the domain of a city guide. A businessman (probably more than anyone else) regularly needs to find a suitable restaurant to have lunch (often in a town he is unfamiliar with) and so requires information about possible factors that might influence his choice, such as the distance away and price of a meal for each place. This task is also one with which the traditional mobile devices would have trouble. In our system the user could retrieve a list of the nearest restaurants (sorted according to a model of her preferences), choose one and then be guided to it using Global Positioning System (GPS) information.

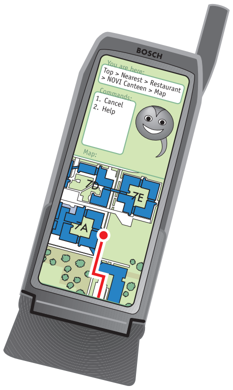


Fig. 1. *Projected system.*

The ideal system would look like the unit in figure 1. It would be the size and shape as a mobile phone, but would have a graphical display covering its entire front (much like Alcatel's One Touch Com™). The top of the display would show feedback from the dialogue while the lower half would show a map of the user's current position and the surrounding area, together with a route to her chosen destination. Ideally, this system should contain a GPS unit and all software necessary to run the system (including the spoken dialogue system). In practice we had to make various compromises.

The prototype mobile unit was implemented in Java on a laptop computer connected to a GPS unit and a wireless modem. The user talked with the dialogue system over a mobile phone and received graphical feedback on the laptop display. The user model consisted of probabilistic preferences representing the user's feeling towards properties of each restaurant (e.g. cost). Feedback from the user model was provided by the behaviour of an animated cartoon creature: when the user model was inaccurate, the creature was confused, otherwise it was happy. The user model was updated using Bayesian decision theory each time the user made a choice.

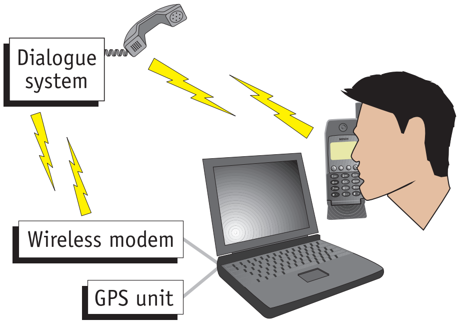


Fig. 2. *Prototype system.*

We constructed a small University campus map with relevant locations. This was used to evaluate the interaction between the user and the system and how well the system learned the user's preferences.

As we intended, the users achieved better interaction with the system the more time they spent with it, as measured by a decrease in errors and requests for help in the dialogue system and a decrease in the amount of time to accomplish their tasks. The user model showed a tightening of the probabilistic preferences and increased the accuracy of their predictions in most cases. Users said they liked the cartoon creature and were concerned when it was confused. Some minor technical problems were the inaccuracy of the GPS unit and the slow speed of the wireless modem.

The small number of people testing the system means that our results can only be pointers to further investigation. Though our test system had a limited number of features, those that were available were understood and made use of extremely quickly. Some advanced functions involving adding to the database of places, and linking together multiple units have already been under consideration and will provide directions for future research. These features can be added without increasing the complexity of the interface, indicating that our design can provide the functionality of a smart phone using only minimal graphics.

2. METHODS AND MATERIALS

2.1 *Introduction*

We designed our interface by first constructing several task-based scenarios. In these scenarios, an imaginary user wanted information specific to her location and personal preferences from a fictitious system with similar capabilities to the one we proposed. We then abstracted the services that our system should provide from the scenarios and specified them as Java interfaces.

The system was then designed according to the agent model described in the Pool Trainer project (Group 870, 1998). This used Java Remote Invocation (RMI) to implement agents providing the specified services. Each agent could run anywhere on the network providing that one agent, the Manager, knew where it was. The Manager then started up the system, connecting the different agents as necessary.

The agents used in our system are shown in figure 3 and a brief description of the services they provide is given below.

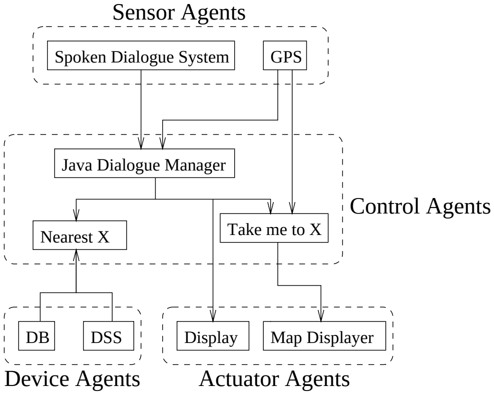


Fig. 3. *Agents in the system.*

Sensor Agents took input from the user and generated events that other agents could understand:

The *Spoken Dialogue System* (SDS) should have been based on the REWARD dialogue system

(Brønsted, Bai and Olsen, 1998). Unfortunately this system was ultimately not available to us and so was replaced by an operator listening to the user's speech and typing into a simulator application. This agent received a list of the currently available commands from the Java Dialogue Manager and returned events indicating that a specific command had been recognised or that the user had said something unrecognisable.

The *GPS* agent provided information about the user's current position. We used a *GARMIN* GPS 25 receiver (Garmin Corp., 1998) combined with the Differential GPS (DGPS) signal from Sonofon's public DGPS service. The DGPS signal is necessary to achieve accuracy due to the US Government's policy of "Selective Availability" (National Security Council, 1996).

Device Agents provided basic services that other agents could make use of:

The *Database* stored the information about all the places that the system knew about. This agent could provide a list of all the different types of places it contained (e.g. Bus Stops, Restaurants and Shops) and could be queried for places of a particular type or that matched specific criteria. Each place was described in terms of several attributes such as location and price range. The Database ensured that places were unique in that no two places shared both the same name and the same location.

The *Decision Support System* (DSS) was the learning part of our system. Using the user model it contained, the DSS could sort places according to how much the user liked them. Whenever the user made a choice, the DSS would update the user model to reflect the knowledge gained by the new decision. This agent also controlled the mood of the cartoon creature: if the user was choosing the places that the DSS predicted then the creature was happy, otherwise it became confused.

Actuator Agents provided output to the user from the system:

The *Map Displayer* showed a map of the area immediately around the user together with a route to his current destination. The map was a bitmap calibrated using two fixed GPS positions and linked to an undirected graph representing its available routes. Each time the Map Displayer received a new position of the user or of the destination the display was updated to reflect the new positions and, if necessary, the new route.

The *Display* agent controlled the main user interface of the device. This was divided into three frames: the top frame displayed the current context of the dialogue (for example, "Top > Nearest > Restaurant"); the middle frame showed any information relevant to the current context, such as the available commands or the list of available restaurants; and the bottom frame consisted of the animated cartoon creature.

Control Agents communicated with other agents to provide more complicated services than the previous agents:

The *Java Dialogue Manager* was the main control agent. It kept track of the dialogue state, which commands were available in that state and the information that should be displayed in that state. The Java Dialogue Manager was also responsible for dispatching the user's commands to the other control agents.

When sent a request from the Java Dialogue Manager the *Nearest X* agent queried the Database for places of the requested type (e.g. Restaurant) and returned a list sorted by the DSS according to the user's profile. This agent was also responsible for relaying the user's choice to the DSS.

The *Take Me To X* agent communicated with the GPS agent and the Map Displayer. When given a place that the user wanted to go to, it informed the Map Displayer of the user's current position and the location of the destination. It then refreshed the user's position in the Map Displayer every time the GPS agent sent a new position event until told to stop by the Java Dialogue Manager.

2.2 *Spoken Dialogue System*

Spoken Language Dialogue Systems (SLDS) are designed to help the user to solve a particular task in which speech input and output play a major part. Their central goal is to enable interactive tasks to be carried out efficiently.

A Dialogue Manager must do three things as far as communication with the user is concerned: advance the domain communication, initiate meta-communication (try to solve a detected problem) and produce appropriate output. As it generates its output to the user it must also update its representation of the current dialogue context and generate whatever constraint based support it can provide to the recognition system.

The Dialogue Structure Internally, a Dialogue Manager must be able to identify the tasks that the user is trying to achieve. In our system these tasks were originally identified from the scenarios we had written which took into account the expected users of the system – business people familiar with the task at hand but not especially familiar with SLDS. We broke the scenarios down into tasks and sub-tasks using a flow chart structure as used in the REWARD dialogue system. This resulted in the hierarchical design shown in figure 4. So for example, a scenario in which the user wanted to update information about a place in the database was broken down into the following subtasks:

(1) Choosing the place the user wanted to change.
(2) Choosing an attribute of the place to alter (e.g. its price range).
(3) Inputting the new value of the attribute.
(4) Verifying their changes and submitting the new information.

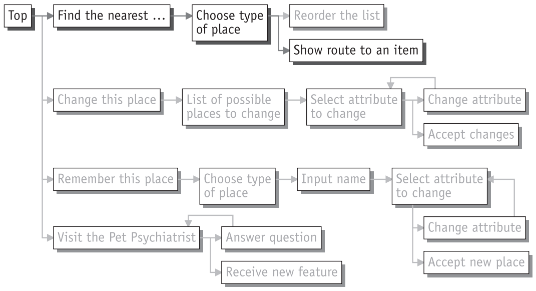


Fig. 4. *Dialogue structure. The grayed out options were not implemented.*

The dialogue was designed with several different tasks in mind, most of which were not implemented in the prototype system. Other tasks involved finding out about other kinds of places than restaurants, for example bus stops and shops; overriding the system's ordering of the displayed list of places (e.g. "Show me the cheapest ones"); altering the database of places by updating existing entries or adding new ones; and a special task of repairing the user model.

This last task was achieved by getting the user to talk to a Pet Psychiatrist. The Pet Psychiatrist would ask him to disclose information about his feelings to the cartoon creature (and thus to the decision support system) so that it could understand him better. The user would be asked questions to ascertain the reasons for a choice that he had made previously and that the system had not predicted (i.e. that had a low probability of being chosen – see section 2.3 on the decision support system). These questions consisted of asking the user to choose which attribute of the available places most influenced their decision. This task was slightly different to the others in that the system, rather than the user, was trying to achieve the goal.

Most of the dialogue was system-directed but the user had the option of initiating meta communication at any point, using the Cancel and Help commands. The former would bring the dialogue back to the top state in the dialogue and the latter would request further explanation of the current state. The top state in the dialogue was user-directed in that the user could choose freely which branch of the hierarchical structure to enter and thus which task they wanted to achieve.

Communication With The User A Dialogue Manager is also responsible for keeping the context of the dialogue up-to-date. This involves informing the both the speech recogniser and the user of the possible actions in the current state (see figure 5).

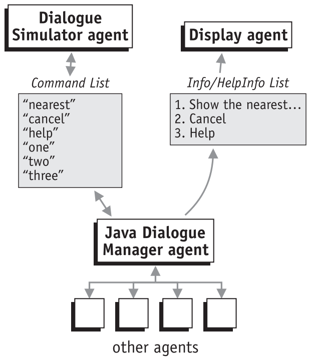


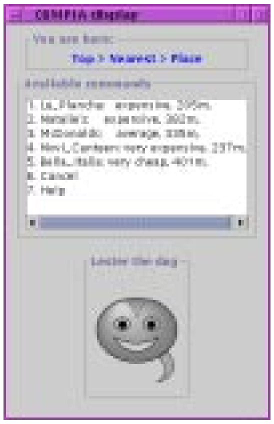
Fig. 5. *Operation of the Dialogue Manager.*

One of the purposes of our system was to make choices on behalf of the user (using the decision support system) in order to cut down the information both displayed and exchanged. This focussed the information provided to the user and the speech recogniser down to a short list of available options.

The REWARD dialogue system contains a Natural Language Parser and a tool to generate subgrammars to recognise the user's spoken commands in each of the dialogue states. However, since this system was not available to us, we decided to use the much simpler technique of word spotting.

The options available in the current state were thus represented to the simulated dialogue system as a list of the words to be spotted (the Command List in figure 5). An operator then listened for these words in the user's spoken commands and used the Dialogue Simulator to send events back to the Dialogue Manager. These events indicated that a specific word had been spotted or that the user had said something unrecognisable.

The same options were presented to the user in the form of the Info List on the Display (see figure 6). This contained brief explanations for each option. More detailed information was available in the HelpInfo List, which replaced the Info List when the user invoked the Help command and when the system did not recognise the user's previous spoken command.



[illegible]

You are here:

Top > Nearest > Place

Available commands

1. La_Planche: expensive, 295m.
2. Natalie's: expensive, 302m.
3. McDonalds: average, 335m.
4. Novi_Canteen: very expensive, 237m.
5. Bella_Italia: very cheap, 401m.
6. Cancel
7. Help

Leslie the dog

Fig. 6. *A sample display during the dialogue.*

Further feedback was provided by displaying the context of the dialogue directly, in the form of the user's current path into the dialogue hierarchy (e.g. "Top > Nearest"). This informed the user exactly what the system had understood so far and meant that a system misunderstanding could easily be observed and corrected by the user.

2.3 *Decision Support System*

The criteria that people use when choosing a restaurant are extremely complex. One person might never want anything more than a sandwich at lunchtime and eat his main meal in the evening; someone else might decide where they want to eat based on how much money they have; another might choose where to go depending on who they are with at the time. All these considerations could also influence the decision of just one individual. We could not model all of these factors in our system so we designed our decision support system to be capable of adding and removing decision criteria.

This was done by describing the user as a set of random variables (RVs). Each RV represented her feelings towards a particular attribute of the restaurants stored in the database. For example, one RV represented the willingness of the user to travel. This was directly linked to the distance from the user's current position to the location of each restaurant.

The system then predicted where the user might want to go using decision theoretic principles (von Neumann and Morgenstern, 1953): we designed a penalty function that took the attributes of each place and combined them with the RVs in the user model to make a single score of how bad the place was. The places could then be ranked according to how much the system thought the user would like to go there.

As a simplification, we assumed that the RVs in the user model were independent. This meant that the penalty function could consist of the sum of the separate penalties for each attribute. However, to make the penalties comparable, they all had to be converted into a common currency.

Since there was no way of measuring absolutely how willing the user was to travel or any other of their subjective feelings, we could not give absolute values to the RVs in the model. Each RV was therefore modelled by a probabilistic distribution which was in turn discretized into different states to allow easy computation.

Each state then had a particular weight which expressed how much a user who felt that way would take into account that particular attribute. So for instance, an RV representing "price aversion" might have three states (see figure 7). The first could be "price no object" and have a very low weight, reflecting a user who didn't really care how much something cost. The second could be "average" and have a weight of 1 and the third could be "student" and have a weight of 20, reflecting someone who would eat anything as long as it was cheap!

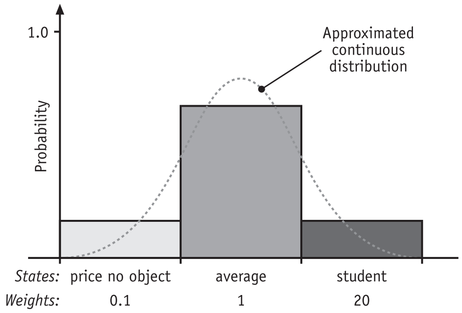


Fig. 7. *Distribution of "Price Aversion".*

The total penalty (TP) assigned to an attribute a was then defined as follows:

$$TP(a) = \sum_{s=1}^{N} kr(a) \cdot w_s \cdot p(s) \tag{1}$$

where N is the number of states in the distribution associated with attribute a, w_s is the weight of state s and kr is a function that converts attributes into the common currency.

The total penalty function for a place with several attributes was then the convolution of the penalty functions for each of the attributes. The resulting value was used to rank the places offered as a choice to the user so that those that fitted her profile were listed first.

We used a Bayesian decision method (Pearl, 1987) to update the RVs when the user made a choice. This interprets Bayes theorem (see Equation 2) as stating that the posterior distribution of an RV (the distribution after an observation has been made) is proportional to the prior distribution ($P(B)$) multiplied by the conditional probability of the observation given the prior distribution ($P(A|B)$). The denominator ($P(A)$) is treated as a normalisation term that ensures that the posterior distribution adds up to 1.

$$P(B|A) = \frac{P(A|B) \cdot P(B)}{P(A)} \tag{2}$$

In our case, the prior distribution was the joint distribution of each of the RVs in the user model before the user made a choice; the posterior was the new joint distribution we wanted to calculate. The conditional probability was the probability that the user chose the place they did given the choices they were offered. This probability was considered to be proportional to the probability density function of a normal distribution with its mean at the prediction with the minimum penalty value (i.e. the expected choice). Figure 8 shows this more clearly.

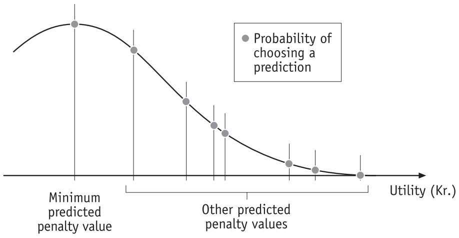


Fig. 8. *Calculating the conditional probability*

This method not only meant that a choice far away from the expected one had a low probability, but that unchosen predictions also had an effect, since their probabilities would be counted into the normalisation factor of the probability of the chosen prediction. For example, several predictions very close to the expected choice would share the probability amongst them, resulting in each of the predictions having quite low probabilities. If the expected choice was by itself it would have a higher probability of being chosen.

The separate distributions of each RV were then calculated by marginalising the posterior joint distribution. This process occurred each time the user made a choice.

3. EVALUATION

To evaluate our system according to our hypothesis that a multimodal intelligent interface greatly improves the ease of interaction with a hand-held device, we performed a user test of our system. Each user was given a task to achieve using the system. Their actions and those of the system were recorded in a dialogue history while they carried out this task. The users were then interviewed about their views on various aspects of the interaction and asked about their opinions of the system as a whole.

Due to time constraints we were only able to test the system with five users. We were also forced to simulate the user's GPS position during testing due to the late arrival of the necessary hardware.

Experimental setup The tests were performed under laboratory conditions with a special test setup involving two "Wizard of Oz" simulations. The user sat in front of the laptop and spoke to the system through a mobile phone. The displays of the system were shown in windows on the laptop screen (see figure 6 and figure 9). On a computer next to the user, a GPS simulator was running showing her simulated position on a map. The user could control their position by pointing at a position on the map. One wizard then used the GPS Simulator agent to inform the system of the change in the user's position. The spoken dialogue system was also simulated in that the users spoke to another wizard over the mobile phone, who informed the system of the users' utterances using the Dialog Simulator agent.

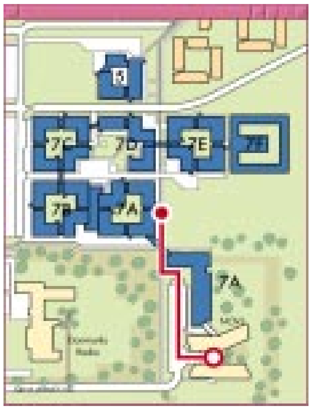


Fig. 9. *System displaying the Map.*

User Task The task the users were presented with was to find a good place to eat for lunch and then a good place for an evening meal given that they were at a specific location. We then asked them to imagine that a day had passed and that they were now in a different location and to try again. We asked each user to go through several "days" use of the system. This was done to evaluate the performance of the decision support system, since it needed several decisions to build up an accurate user model.

Interview Format After the test session each user was interviewed. The interview consisted of several open questions designed to give us feedback on the system by ascertaining whether they liked the interaction; whether they ever lost track of the dialogue; which display items they found most useful and which not useful; what they thought of the cartoon creature; and whether they considered the system useful.

User profile	
Sex	Age
Female	25
Spoken dialogue exp.	Mobile phone
Some	yes

System shows	User says	Recognised
Top	"...near restaurant"	misrecognised
Top(Help)	"..the nearest places"	nearest
Nearest	"Restaurants"	restaurant
Restaurants	"...restaurant one"	one
Map	"Cool'n'funky!!"	misrecognised
Map(Help)	"Cancel"	cancel
Time: 35		

Table 1. *System log from user number 1.*

4. RESULTS

Five users with ages between 20 and 30 years tested our system. All had previous experience with conventional mobile phones and computers and a few had some experience with spoken dialog systems.

4.1 *Dialogue Histories*

In all cases, the users became familiar with the dialogue system extremely quickly (after three or four uses). This can be seen in figure 10. There were very few help requests since most users either spoke the commands exactly as they saw them on the display or else tried to converse with the system naturally, receiving the help text when a command was not recognised. However, the misrecognition rate was quite high amongst the latter group. A small part of a dialogue history is shown in table 1.

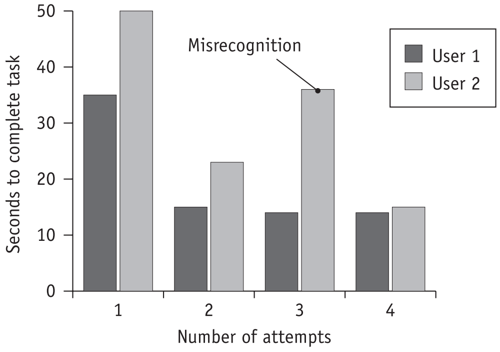


Fig. 10. *Times to complete the tasks for two users.*

The users were confused when the system was busy, as indicated by repetition of their commands. This mostly occurred in the pause whilst the system was bringing up the map display. Commands were also repeated when a misrecognition occurred while the system was displaying the help information.

4.2 *Interviews*

The results of the interviews were mostly positive. The users like the interaction with our system and did not get lost in the dialogue interface. They found the display showing the state of the dialogue and the options available extremely useful. They also enjoyed watching the animated cartoon creature and showed concern when it became confused. However, several users complained that it did not respond directly to their commands and did not reflect the state of the system at all times (e.g. when the system was busy).

Most users found it hard to imagine that several days had passed during the few minutes of their testing period and so either made the same decisions again or picked new criteria for choosing a meal out of thin air. Clearer instructions and a more natural testing environment, for example with the users carrying the laptop around with them for a whole day, would have avoided this (though it would have involved the use of the real GPS unit). Despite the unnatural situation, the decision support system was able to build up some kind of a user profile in each case, as measured by a tightening of the probability distributions of the RVs. In some cases there was also a definite increase in the probability of the prediction that the user chose as the task proceeded. Some users even said that the ordering of the restaurants directly influenced their decision and that they chose some items *because* they were near the top of the list.

All the users said that the system could be useful if integrated into a mobile phone and provided with information about lots of restaurants in different places.

5. DISCUSSION

Information overload is a growing problem in hand-held devices, as manufactures cram more and more features into a smaller and smaller unit. The result of this is products with very complex and compact user interfaces. In this paper we investigate users' responses to changing the traditional interface of a mobile phone to an intelligent multimodal interface. The small number of people testing the system means that our results can only be pointers to further investigation.

Our user test indicated that the system was easy to use and provided a simple interface to a complex task. The users all succeeded in their tasks of finding suitable restaurants. The tasks were completed quickly and without difficulty and the users soon became "expert" at using the system as shown by a reduction in the time it took them to complete their tasks. However, the tests did suggest some that some improvements could be made.

Many of the views expressed by users in their debriefing can fall under the heading of requests for additional feedback. Their suggestions included the aforementioned need to know when the system was busy processing their commands and a request for more behaviour for the animated creature, such as making it look at the map when it was being displayed. However, giving the creature complicated behaviours might work against the generality of the interface: the creature provides meta communication with the user that is not related to the task at hand and mixing this together with the task-based communication could cause confusion. Making the creature too complex might also raise the users' expectations of its intelligence too far (Koda, 1996). This is the reason we chose to have a simple, animated cartoon rather than a realistic, humanoid representing the system.

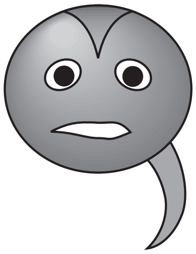


Fig. 11. *The cartoon creature in its confused state.*

Dialogue As can be seen from the results, the spoken dialogue system worked as intended. The dialogue structure we implemented was a very simple one and therefore lacked some of the features one would expect of a real system. However, the focus of the present study was not on generating a perfect spoken dialogue system but creating one which worked sufficiently well to serve as part of an intelligent multimodal system. This allowed us to investigate other issues not directly related to the spoken dialogue system. Also the dialogue system was not designed work by itself, as the user and the system were able the refer to information displayed on the screen.

As the ideal unit would be active all the time (you always take your mobile unit with you) there was no need for the usual opening and closing interaction in the dialogue system. The use of the cartoon creature also brought out this aspect of the dialogue: talking to a simple creature requires simple language.

In spite of the uncomplicated structure of our dialogue, the users did experience some inconvenience when misrecognitions occurred. This was partly due to the primitive facilities for repair in our system: the only meta command available to initiate dialogue repair was "Cancel", which took the user back to the top level of the dialogue. If the system kept a task history, it would be possible to undo the last command and go just one level back up in the dialogue structure or even switch to a different branch of the dialogue structure without losing entered information. This would make the dialogue more user directed and offer the user a more natural method of communication.

Another reason for the inconvenience of misrecognitions was the lack of graceful degradation in our error handling system. There was only one level of help in all states of the dialogue: when the user was already in this state and said something unrecognisable the system did not appear to respond at all. This is a major problem with our system, but it could be fixed quite easily. At the moment each state only has one HelpInfo list which is shown if the system does not recognise the user's input or if the user requests help. Extending this to a set of HelpInfo lists that gradually increase the level of detail would be much more effective. After all the help has been exhausted, the Dialogue Manager should recognise this and halt the current task, taking the user back to the top of the dialogue (or maybe to the Pet Psychiatrist to ask them why their pet can't understand them!).

The purpose of the entire system was to limit the amount and complexity of the information presented to the user. To do this the system made some choices on the user's behalf. However, sometimes the user may wish to do something different and "out of character". For example, a student going out to eat with her parents would not go to the same places she would normally eat (especially if her parents were paying). A command to reorder the list of displayed places (e.g. "Show me more expensive restaurants") would then be needed. This again increases the naturalness of communication by offering a more user directed approach.

Decision Support System The major assumption in the Decision Support System was that the different RVs in the user model were independent of each other and of the different places in the database. This is not necessarily true, especially if the database contains different types of places. For example, one user might really like food and be willing to travel a longer distance to go to a restaurant, but might not care where he had a haircut and always choose the closest barber.

This assumption cuts down the number of inferences that can be made about the user, but it fits quite well with the impression of the system as a cartoon creature with limited intelligence – people don't expect a pet to make complex logical deductions.

Another possible problem with the user model is that the RVs are time independent, i.e. the user's preferences are considered to stay the same the whole day long. This had an impact on our test since the users applied different criteria when choosing a restaurant for dinner than when choosing one for lunch. This fault could be rectified by having separate distributions for different times of the day. However, this would come at the price of the system taking longer to learn since it would then have to learn each part of the day separately.

System Flexibility As described in the Evaluation, the GPS unit was not available for the test and a simulator was be used. This was made simple by the agent and services model around which our system is built. To replace the real GPS unit we merely had to inform the Manager agent that this service was being simulated and implement a simulator that could provide the same service. We have improved the design from the Pool Training project (Group 870, 1998) in that we can now have several simulators running at the same time, each simulating several different services.

The same model allows us to add new features without changing the structure of the system. New services can be defined and agents implemented to provide them almost entirely separately of the rest of the system. Only the Manager configuration and the Dialogue System have to change.

5.1 *Future Development*

More features can be added without increasing the complexity of the interface, suggesting that our design could provide the functionality of a smart phone using only minimal graphics. We have already designed several additional features of our system. Some of these were mentioned briefly in the description of the Dialogue System above: changing and inserting entries in the database of places, reordering the list of places according to specified attributes (e.g. "Show me the cheapest places"), and the Pet Psychiatrist.

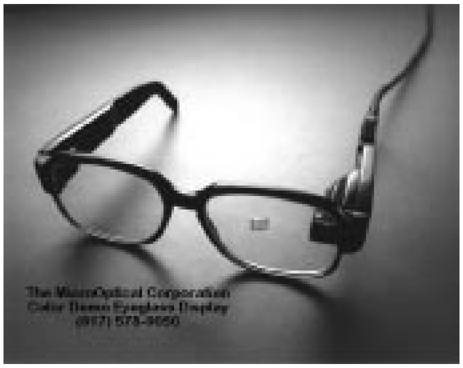


Fig. 12. *The MicroOptical eyeglass display.*

The purpose of the Pet Psychiatrist was to introduce some deeper communication with the user, making her feel some responsibility for the creature, especially when it became confused. This concept of responsibility was found to be a key part of the attraction of Tamagotchis when we investigated them briefly at the beginning of our project. It is also an effective method of keeping the user model up-to-date.

The Pet Psychiatrist also had another function in our original design: that of gradually introducing new features. The idea is that the system starts off with a limited set of features and that users *earn* new commands. This is done by keeping a good relationship with the creature for a certain amount of time – visiting the Pet Psychiatrist when it shows signs of getting confused. The system thus becomes personalised to the users' level of experience as well as to their preferences.

Multiple Units Another direction of development is the involvement of multiple units. One feature that we designed was the possibility of choosing another unit as the destination rather than a fixed point. This is like the game of "tag" in that one user chooses to be "tag" and broadcast her position. Other users can then follow a "tagged" person and will be shown a map with a route to that person which will be updated every time they or the "tag" moves.

Once many users are using the system, it begins to open up to community use. If the database of places is stored centrally rather than on each individual unit, an attribute could then be added to each place reflecting how many people had visited it: each user would then have a preference that reflected how much they believed in the group mentality. See Yoshiyasu Nishibe, *et al.* (1998) for more details on a system designed for multiple users to share their feelings about places they visit. In a future commercial application the network operator could even sell access to new maps and database information.

Advanced Technologies The map display we use is not the most economical way of showing someone the way. If the direction that the unit is pointing is known, then spatial relations can be used to guide the user to the left, to the right or straight ahead. GPS information cannot provide this directional information but an electronic compass can. Some in-car navigation systems already use this kind of information and reduce the display to a minimum by only telling the user about the next turn she should make.

A completely different approach would be to use a Head-Up-Display (HUD) to provide the user with a large amount of graphical information without making the unit any bigger. If our unit was linked to the MicroOptical Eyeglass display (MicroOptical Corp., 1998. See figure 12), the user would have a big, transparent screen floating in front of her. Alternatively, the HUD can be mounted in the unit itself as in the FaxView (Reflection Technology, 1996). Another possible configuration would be a flip-out view-finder attached to the mobile phone, much as some phones already have a flip-out mouthpiece.

A device with these extra technologies would still benefit from our multimodal intelligent interface. Spoken dialogue interaction would still be a very natural way of communicating with the system and the decision support system would still save the user from searching through all the available information.

6. CONCLUSION

The main result of our tests is that the users quickly become familiar with our system as shown by the decrease in the times it took them to complete the tasks. We can also see that our system is able to adapt to the users' individual preferences.

Users liked the idea of a device that was specially adapted to their needs. They thought it was a good idea to personalise the data to cut down on the information presented to them. Our prototype has proved its feasibility and could easily be used as a base for further development. Bosch Telecom Danmark A/S has shown great interest in the system and has decided to sponsor further projects in Intelligent Multimedia at Aalborg University.

ACKNOWLEDGEMENTS

Bosch Telecom Danmark A/S, for the loan of the laptop, GPS unit and mobile phones.
Lars Ørnbo, R&D S/W Engineer, Bosch Telecom, for his time.
Sonofon, for the mobile phone accounts
Arden Bogtrykkeri A/S, for letting us use their digital map of the university campus.

7. REFERENCES

[Brønsted et al., 1998] Brønsted, T., Bai, B. N., and Olsen, J. Ø. (1998). The Reward service creation environment, an overview. In *ICSLP Sidney.*

[Garmin Corporation, 1998] Garmin Corporation (1998). *Garmin 1200 E.* 151st ST Olathe, Kansas 66062. http://www.garmin.com.

[Group 870, 1998] Group 870 (1998). Intelligent multimedia based pool trainer. 8th Semester Project in IMM, 1998, Aalborg University, DK.

[Koda, 1996] Koda, T. (September, 1996). Agents with faces: A study on the effects of personification of software agents. Master's thesis, MIT Media Laboratory.

[MicroOptical Corporation,] MicroOptical Corporation. Eyeglass Display FAQ. http://www.microopticalcorp.com/.

[National Security Council, 1996] National Security Council (1996). Fact sheet – U.S. global positioning system policy. http://www.whitehouse.gov/WH/EOP/OSTP/html/gps-factsheet.html.

[Nishibe et al., 1998] Nishibe, Y. et al. (Summer 1998). Mobile digital assistants for community support. *AI Magazine.*

[Pearl, 1987] Pearl, J. (1987). Bayesian decision methods. In *Encyclopedia of AI*, pages 48–56. John Wiley & Sons, Inc.

[Pirjanian et al., 1998] Pirjanian, P., Madsen, C. B., and Granum, E. (1998). Bouncy: An interactive life-like pet. http://www.vision.auc.dk/~paolo/staging/bouncy.

[Reflection Technology, Inc.,] Reflection Technology, Inc. Faxview. http://www.reflectiontech.com/faxview.html.

[von Neumann and Morgenstern, 1953] von Neumann, J. and Morgenstern, O. (1953). *Theory of games and economic behaviour.* Wiley, New York.