HOMER: Human Oriented MEssenger Robot

Pantelis Elinas and **Jesse Hoey** and **James J. Little**
Department of Computer Science
University of British Columbia
2366 Main Mall, Vancouver, BC, CANADA, V6T 1Z4
{elinas,jhoey,little}@cs.ubc.ca

Abstract

HOMER, the Human Oriented MEssenger Robot, is a stereo-vision guided mobile robot for performing human-interactive tasks. Our design concept for HOMER combines mobile robotic techniques for navigation, localization, map building and obstacle avoidance with human interaction capacities for person recognition, speech, facial expression and gesture recognition, and human dynamics modeling. HOMER's capabilities are modular and independent, and are integrated in a consistent and scalable fashion under the umbrella of a decision-theoretic planner, which models the uncertain effects of the robot's actions. The planner uses factored Markov decision processes, allowing for simple specification of tasks, goals and state spaces. We demonstrate HOMER performing a message delivery task, which is rich and complex both in robot navigation and in human interaction.

Introduction

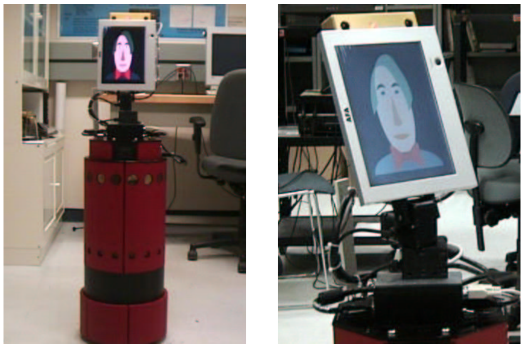
This paper describes our work on HOMER, the Human Oriented MEssenger Robot, a mobile robot that communicates messages between humans in a workspace. The message delivery task is a challenging domain for an interactive robot. It presents all the difficulties associated with uncertain navigation in a changing environment, as well as those associated with exchanging information and taking commands from humans using a natural interface. In designing HOMER, however, we are concerned with the more general problem of building mobile robotic systems with capacities to interact with humans independently of the task they are asked to perform. Such robots will need navigation, mapping, localization and obstacle avoidance capabilities to deal with moving around in an uncertain and changing environment. They will also need to model the dynamics of people in an environment, including their locations in space and their behavioral patterns. Finally, human users will require such robots to present clear, simple and natural interactive interfaces, which enable easy exchanges of information between robot and human. The systems to deal with each of these problems can be made fairly independent, and thus can be implemented modularly.

The remaining challenge is then to integrate the modules to perform a given task. The task specification should therefore be in a simple language that enables efficient extension or re-assignment of a robot's task. This paper presents our modular and scalable design of HOMER's hardware and software systems, which provides for easy integration of sensor and actuator modules for a given task specification. Although we are describing HOMER's application to the message delivery task, our system can easily be extended or re-assigned to other human-interactive mobile robotic tasks.

Building service robots to help people has been the subject of much recent research. The challenge is to achieve reliable system that operate in highly dynamic environments and have easy to use interfaces. This involves solving both the more traditional robot problems of navigation and localization and the more recent problems in human-robot interaction. Another challenge arises from the large scope of these systems and the many pieces that must be integrated together to make them work. RHINO (Burgard *et al.* 1998), was one of the most successful service robots ever built. It was designed as a museum tour guide. RHINO successfully navigated a very dynamic environment using laser sensors and interacted with people using pre-recorded information; a person could select a specific tour of the museum by pressing one of many buttons on the robot. RHINO's task planning was specified using an extension to the GOLOG language called GOLEX; GOLEX is an extension of first order calculus, but with the added ability to generating hierarchical plans and a run-time component monitoring the execution of those plans. MINERVA (Thrun *et al.* 1999), was the successor of RHINO. MINERVA differed from RHINO in that it could generate tours of exhibits in real-time as opposed to choosing one of several pre-determined tours. MINERVA also improved on the interaction by incorporating a steerable head capable of displaying different emotional states.

More recently, (Montemerlo *et al.* 2002) designed PEARL, a robot for assisting the elderly. PEARL's main task is to escort people around an assisted living facility. Its navigation and localization uses probabilistic techniques with laser sensors. PEARL is more focused on the interaction side with an expressive face and a speech recognition engine. PEARL's largest contribution is the use of a partially observable Markov decision process for modeling uncertainty at the highest level of task specification.

We begin this paper by introducing our mobile robot,



(a) (b)

Figure 1: (a) HOMER the messenger robot (b) closeup of HOMER's head

HOMER, describing his hardware and software systems. We then show how HOMER navigates through the world, maps the world, localizes himself, recognizes faces and searches for people. Following this, we describe how HOMER plans his actions. We then introduce the domain of message delivery, and show results of some experiments showing HOMER's performance in a simple domain. We discuss the directions we are currently pursuing and conclude the paper.

Hardware

Our robot, HOMER, shown in Figure 1(a), is a Real World Interface B-14 robot, and has a single sensor: a Point Grey Research [1] Bumblebee^{TM} stereo vision camera. The Bumblebee is mounted atop an LCD screen upon which is displayed a pleasant and dynamic animated face. We call the combination of Bumblebee and LCD screen *the head*. The face displays non-verbal invitations to humans to approach and speak, expresses emotions, or emphasizes or conveys further information. The head is mounted on a Directed Perception pan-tilt unit, as shown in Figure 1(b), which provides lateral and dorsal movement for the camera system and animated face, enabling visual search and following for realistic interaction.

The use of a single stereo camera for all sensing of his environment is what makes HOMER stand out as a robot. Vision provides rich, high bandwidth, two dimensional data containing information about color, texture, depth and optic flow, among others. This multi-modal data source can be exploited universally for the accomplishment of many different tasks. It is a harmonious host of information about a robot's environment, and is an alternative to more specialized sensors such as sonar or laser range finders. Mounting the stereo camera on a pan-tilt unit adds flexibility to HOMER's real-time navigation and interaction.

HOMER is equipped with 4 on board computers based on the Intel Pentium III processor and running the LINUX operating system. The computers communicate among each other using a 100Mbps local area network. A Compaq wireless 10Mbps network connection allows the robot's computer to communicate with any other computer in our lab's LAN for additional compute power as needed. The Bumblebee unit outputs images and dense stereo maps over IEEE-1394 (Firewire) connection to one of the on board workstations. Both images and stereo maps are used by HOMER to build two dimensional maps of his environment, to localize himself with respect to that map, and to detect and recognize objects and people (Murray & Little 2000; Elinas *et al.* 2002; Elinas & Little 2002). HOMER's onboard processors are used for all modules which need extensive and rapid access to image data or to the motors, including the face recognition software and all lower controllers for motion and pan-tilt action. The planning engine and manager module run on two different off-board machines as they require less bandwidth for communication.

HOMER's actuators include motors for translation and rotation in two dimensions, motors for movement of the head, speech through on-board mono speakers, and facial expression generation in the animated face on the LCD screen.

[1] www.ptgrey.com

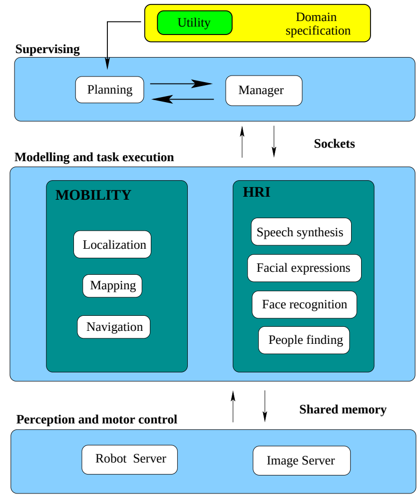


Figure 2: Control Architecture

Software Architecture

HOMER's software system is designed around a Behavior-based architecture (Arkin 1998; Brooks 1986). For HOMER, a behavior is an independent software module that solves a particular problem, such as navigation or face recognition. We refer to behaviors interchangeably as modules in what follows. Behaviors exist at 3 different levels, as shown in Figure 2. The lowest level behaviors interface with the robot's sensors and actuators, relaying commands to the motors or retrieving images from the camera. These are described more fully elsewhere (Elinas & Little 2002). Behaviors at the middle level can be grouped in two broad categories: mobility and human-robot interaction (HRI). Mobility modules perform mapping, localization and navigation (Murray & Little 2000). HRI modules are for face recognition, people finding, speech synthesis, facial expression generation. In the coming months we plan to add more middle level behaviors, including speech recognition and natural language understanding, 3D occupancy grid mapping, facial expression recognition, sound localization and gesture recognition (Elinas & Little 2002). Middle level modules interface with the lowest level through a shared memory mechanism. Each middle level module outputs some aspect of the state of the environment. For example, the face recognition module reports a distribution over people's faces in its database, while the navigation module reports the current location of the robot with respect to the maps. These outputs are typically reported to the highest level modules. Each module further offers a set of possible actions the module can effect. All communication among the middle and high level behaviors is done using sockets.

There are two high-level modules: a manager and a planner. The manager maps between the output of the modules and the inputs to the planning engine. The manager may also have some internal state which it controls. The current outputs of all the modules (and the manager's state) is the current state of the robot. The manager's job is to integrate all the state information from the modules with its own, and present the result to the planning engine, which consults a policy of action and recommends some action. The manager then delegates this action to whatever modules respond to it.

In an architecture of this style, the challenges are in the task divisions among behaviors, and in allowing for easily constructed, debugged, and extended manager and planning modules. In our past work (Elinas *et al.* 2002), we implemented the manager and planner together as a finite state machine, which is typically difficult to debug and extend. In this work we separate planning and management tasks, and use a Markov decision process (MDP) domain representation for the planner (Puterman 1994). This allows the robot's tasks to be encoded at a high-level, and makes the high level modules much easier to implement and extend in the future.

Modules

Crucial to the design of a human-interactive mobile robot is to is the ability to rapidly and easily modify the robot's behavior, specifying the changes in a clear and simple language of high-level concepts. For example, we may wish to modify our message delivery robot so that it also delivers coffee. The robot will need new hardware (an actuator to grab the coffee with) and new sensors (to operate the new actuators, to recognize cash money,...). Further it will need to be able to plan solutions to deal with the extended state space of coffee delivery. For example, it now needs to plan for the situation in which one buys for and receives a coffee from an attendant. These design constraints call for a mobile robot to have a modular software system, with a planning module written in a simple and easy to use language for specifying new world states and goals. The additional resources, sensors and actuators needed for the additional tasks, should be simple to add to the existing system. Further, the solution concept for the robot must be easily expanded to include the new facets of its job. HOMER is an implementation of a human-interactive mobile robot with such design principles in mind. Independently operating modules from the core of HOMER's architecture, as shown

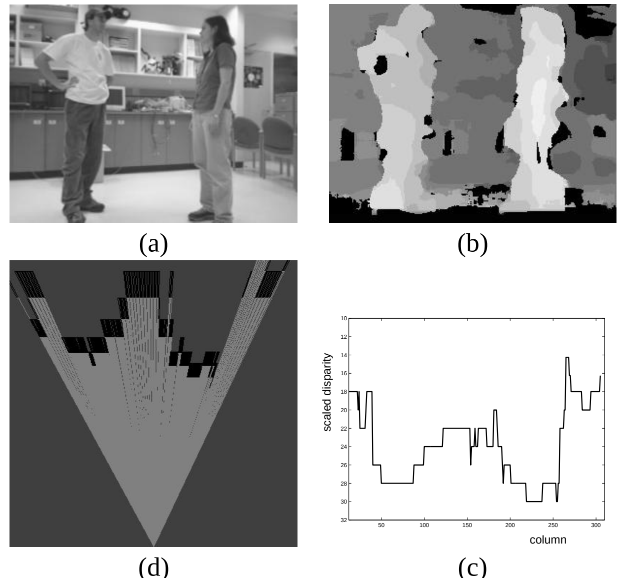


Figure 3: From stereo images to radial maps. (a) greyscale image (b) disparity image (black indicates invalid, otherwise brighter indicates closer to the cameras) (c) depth vs columns graph (depth in cm) (d) the resultant estimate of clear, unknown and occupied regions (light grey is clear, black is occupied and dark grey is unknown)

in Figure 2. They report their states to a manager, who collects information from all the robot's modules, synthesizes a current world state, which is reported to a planning engine. The planning engine returns an optimal action, which the manager delegates to one or more modules. The following sections present the different modules, the manager and the planning engine. HOMER's current modules perform navigation, mapping and localization (Murray & Little 2000; Se, Lowe, & Little 2002),face recognition, and person location modeling. Following the descriptions of the modules is a description of the manager and planner.

Mapping, Localization and Navigation

HOMER uses 2D occupancy grid maps for navigation and localization. Figure 3 shows the construction of the 2D occupancy grid sensor reading from a single 3D stereo image. Figure 3(a) shows the reference camera greyscale image (320x240 pixels), and (b) the resulting disparity image. Black regions indicate image areas which were invalidated. Otherwise, brighter areas indicate higher disparities (closer to the camera). The maximum disparities in each column are converted to depth to produce a *radial map*, as shown in Figure 3(c). Figure 3(d) shows these depth values converted into an occupancy grid representation; light grey indicates clear regions, black indicates occupied, and dark grey indicates unknown areas. The process illustrated in Figure 3 generates the input into our stereo vision occupancy grid. Because the camera is mounted on a pan-tilt unit, care must be taken to transform these occupancy values to the robot's coordinate frame before adding them to the global occupancy grid. The mapping module integrates the local maps over time, keeping the global map current over time. We identify an obstacle at all locations in the occupancy grid where the value is above a threshold.

Safe mobility involves simultaneous localization and mapping (SLAM). The robot must build a map of the environment and track its position relative to that environment. However, accurate localization is a prerequisite for building a good map, and having an accurate map is essential for good localization. This problem has been a central research topic for the past few years (Simmons & Koenig 1995; Burgard *et al.* 1998; Dellaert *et al.* 1999; Thrun 2000). Our vision-based SLAM algorithm uses stereo data of features detected by the Scale Invariant Feature Transform (SIFT) (Se, Lowe, & Little 2002). Simply put, HOMER finds out where he is by recognizing and locating previously observed visual features in his environment. SIFT features are invariant to image translation, scaling, rotation, and partially invariant to illumination changes and affine or 3D projection. These characteristics make SIFT features suitable landmarks for mapping and localization, since when mobile robots are moving around in an environment, landmarks are observed from different angles, distances and under different illuminations.

The navigation task is to find the shortest and safest path connecting two locations given the occupancy grid map, The path planning algorithm we use is a mixture of shortest path and potential field methods. In clear areas, the method operates as a shortest path planner with a fixed distance constraint from obstacles. In cluttered areas, the method turns into a potential field planner, to avoid getting stuck. The combination of the two allows the robot to navigate efficiently in clear environments without getting stuck in cluttered areas. Our navigator is described more fully in (Murray & Little 2000).

Face recognition

Our face detection and recognition process takes place in two steps at each frame. It first searches for candidate face regions using skin color segmentation, followed by connected components analysis. Although we have found this to be a relatively robust method for detecting candidate face regions, it can fail due to changing lighting conditions. A more sophisticated approach may be desirable in the future (Viola & Jones 2001). We maintain a set of color templates of people's faces in a database, and a set of mappings from skin color segmented regions to color template matches. These mappings allow the face region to be found regardless of how the skin segmentation algorithm responds to given person's skin color. A new image is first segmented and the largest skin-colored regions are found. The mappings are then applied to each region for each template, and the input image regions are correlated with the templates, using raw squared match scores. A small local search fine-tunes the location of the match, and the log likelihood of the observation given each template, i, $\Pr(O|T_i)$ can be estimated. A probability distribution over N_p persons, $H_k, i \in 1..N_p$, can then be estimated using Bayes' rule by summing over all the templates of that person in the database, weighted by this

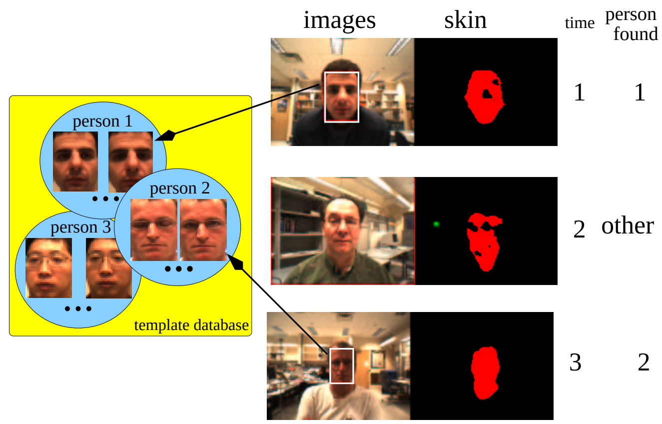


Figure 4: Face recognition. Top row shows a database of exemplars for three persons. Bottom row shows some input images, linked to their most likely exemplar. Most likely match scores and reported person are also shown.

likelihood.

$$\Pr(H_k|O) = \sum_{i=1}^{N_t} \Pr(O|T_i) P(T_i|H_k) P(H_k),$$

where N_t is the number of templates, $P(T_i|H_k)$ is the probability of each template given the person, and $P(H_k)$ is the prior probability of observing each person. We have found that this method works relatively well with a small database of few people. However, an approach using Eigenfaces (Turk & Pentland 1991), may be more desirable for a larger database. Figure 4 (top row) shows a set of exemplars in a database of three persons. Along the bottom row is shown a series of input images linked to their best-match exemplars. Also shown is the skin segmentation for each image. Below each image is the best match score and the reported person among none, other, person 1, person 2 or person 3. The face recognition module waits for an instruction from the manager to start classifying any faces in its field of view. It then analyzes five images taken over about a 10-20 second period, and classifies each as having *no* face, an *other* face, or the face of $person_i$. $Person_i$ then gets reported only if three or more (out of five) images are consistently reporting that person as the most likely candidate. The system reports an *other* face if there are valid skin regions, but no valid template match, and otherwise reports *no* face.

Locating People

In order to find message recipients, HOMER must maintain some model of people's behaviors and their usual whereabouts. At present, HOMER's people-finding module maintains a *location likelihood* function of finding the k^{th} person at each location, $\vec{x}$, in the map: $\mathcal{L}_k(\vec{x})$. These maps are initially constructed by the designer and are updated dynamically as the robot recognizes people during his quests. When searching for a message recipient, HOMER maintains a dynamic version of the likelihood function, $\mathcal{L}'_t$, for the subject of the search at each time, t. When starting the search for person k, $\mathcal{L}_0'(\vec{x}) = \mathcal{L}_k(\vec{x})$. The people finding module then reports the closest unvisited maximum of $\mathcal{L}'$ as the next best location to search for the message recipient. The dynamic map is updated as the search progresses and HOMER discovers that the recipient is not present at various locations.

Our experiments in various locations have shown that this likelihood function produces reasonable goals. If no information about a person is available, HOMER will wander the room in an exploratory fashion. Otherwise, HOMER will start going through all the possible locations starting from the one closest to him. Once all possible known locations are searched, the people finding module indicates that the recipient is not currently reachable. However, more sophisticated people behavior models could be implemented. For example, person following behaviors may be necessary for HOMER to chase down a person who is on the move. Modeling people's temporal behavior patterns may also be useful for finding people who have recently been observed (Bennewitz, Burgard, & Thrun 2002).

Management and Planning

The manager collects information from each module, and integrates it together into a single, fully observable state vector. That is, the state of the system is described by a vector $\mathbf{X} = \bigcup_{i=1}^{N} \mathbf{X}_i$, where $\mathbf{X}_i$ is the state vector for each module and N is the number of modules. The manager's job is to map between the outputs of the modules and the inputs to the planning domain specification. Since we use a planner which requires full observability of the state, the manager may be responsible for compressing the probabilistic belief state reported by a module by choosing a maximum value. For example, the face recognition module reports a vector of $P(H_k|O), k \in 1...N_p$. The manager must then report to the planner a binary vector describing the presence of each person $i \in 1...N_p$. To do so, it must be able to threshold the input vector. Of course, the module itself can take care of the thresholding, in which case the manager simply appends it to the state vector.

This belief compression technique clearly removes information which may be useful to the planner. In general, the modules will not only report their state, but also some variance information about the measurement of the state. If this information is included in the message from manager to planner, the planner, to take best advantage of all information available, should use a partially observable Markov decision process (POMDP). However, it is P-SPACE hard in the general case to find policies for POMDPs. Approximate or hierarchical techniques have been used for robotics applications (Simmons & Koenig 1995; Theocharous, Rohanimanesh, & Mahadevan 2001; Montemerlo *et al.* 2002). HOMER makes the simplifying approximation of full observability as a fair tradeoff between the extra computational burden imposed on the manager, and that taken by the planner. In general, however, modeling the uncertainty in the robot's measurements, if tractable, will improve the high-level plans generated. A POMDP planner could be easily fit into HOMER's architecture if needed.

The planner has access to a specification of the domain in terms of the random variables for each module, and any others the manager may need to define independently, and to a utility, or reward, function which encodes the goals and preferences of the robot in its domain. The planner models the temporal progression of the state variables reported to it by the manager with a fully observable Markov decision process (MDP). Markov decision processes (MDPs) have become the semantic model of choice for decision theoretic planning (DTP) in the AI planning community (Puterman 1994). They are simple for domain experts to specify, or can be learned from data. They are the subject of much current research, and have many well studied properties including approximate solution and learning techniques. An MDP is a tuple $\{\mathcal{S}, \mathcal{A}, \Pr, R\}$, where $\mathcal{S}$ is a finite set of states and $\mathcal{A}$ is a finite set of actions. Actions induce stochastic state transitions, with $\Pr(s, a, t)$ denoting the probability with which state t is reached when action a is executed at state s. $R(s)$ is a real-valued reward function, associating with each state s its immediate utility $R(s)$.

We use a factored, structured, MDP solver, SPUDD (Hoey *et al.* 1999), which takes as input the conditional probabilities, $\Pr$, and the reward function $R(s)$, and computes an optimal infinite-horizon policy of action for each state, assuming a *expected total discounted reward* as our optimality criterion. SPUDD uses the value iteration algorithm to compute the policy. The policy can then be queried with the current state vector reported by the manager, and returns the optimal action to be taken. The manager then delegates this action command to the appropriate set of modules.

SPUDD uses a structured representation of MDPs as decision diagrams. It is able to take advantage of structure in the underlying process to make computation more efficient, and is therefore scalable towards larger environments. The modularity of our system makes representation as a factored MDP simple and typically results in a sparsely connected Markov network. Such sparseness leads to very efficient calculations when using a structured solution approach as in SPUDD. However, since the optimal policy is defined over the entire state space, the resulting structure can become intractably large. Such situations require the use of hierarchical models, or of approximation techniques.

Message delivery

HOMER's message delivery task consists of accepting messages, finding recipients and delivering messages. The planner models this environment using six variables. The first describes if HOMER has a message or not (**has_message**). The next three encode whether HOMER has a goal location (**has_goal**), whether he has reached that goal (**at_goal**), and if the goal is deemed unreachable (**goal_unreachable**). Finally, the last two describe whether HOMER has a sender (**has_sender**) and a recipient (**has_recipient**). HOMER's high-level actions are shown in Table 1, along with the module that is responsible for performing each action, and the main state variables which are effected by the action. The reception and delivery of messages will eventually be delegated to a speech recognition and facial expression interaction module, but is currently handled by the manager. The optimal policy for this domain specification is to accept a

action	module	effects
i.d. person	face recognition	**has_sender**
		has_recipient
get goal	people finder	**has_goal**
		at_goal
navigate	navigator	**at_goal**
		has_goal
receive message	manager	**has_message**
deliver message	manager	**has_message**

Table 1: Homer's high-level actions, the modules which effect them, and the effects of the actions on the robot's state.

message from a *sender*, navigate to potential recipient locations, attempt to recognize the recipient at each location, and deliver the message once the recipient is recognized. There are three major components to this task. The first is the interaction with humans when accepting or delivering messages. The second is the modeling of people's behavior within a workspace, which allows the message delivery robot to infer where to find a given message recipient. The third is the navigation and localization necessary to get from place to place.

In his quiescent state, HOMER waits for a call from a message sender. A potential sender can initiate an interaction with HOMER by calling his name, or by presenting herself to the robot. HOMER uses face recognition to find the person who has called him. Once a person has been recognized, HOMER accepts a message, which includes a recipient's name. In the future, HOMER will use speech, facial expression, and gesture recognition during interaction with people. At present, we plan to implement these as separate modules. However, due to the inherent coupling between these communication modalities, we may wish to integrate them into a single module in the future.

Once HOMER has a message to deliver, he must find the recipient. This requires some model of the typical behavioral patterns of people withing HOMER's workspace. We use a static map of person locations, which is updated when new information is obtained about the presence or absence of persons. This map allows HOMER to assess the most likely location to find a person at any time. Navigation to that location is then attempted. If the location is not reachable, HOMER finds another location and re-plans. If the location is reached, then HOMER attempts to find a potential receiver using face recognition and sound localization. Upon verifying the receivers presence, HOMER delivers the message.

Experiments

We have run some simple tests of HOMER's message delivery capabilities in our laboratory. We first built an occupancy grid map of our laboratory, as shown in Figure 5 (a). We then gathered five templates of each of three persons' faces, some examples of which are shown in Figure 4. Finally, we manually specified the location likelihood for each person.

HOMER waited at home base, attempting to recognize

(a)	(b)	(c)
time 1	2	3

Figure 5: HOMER test run. Black: obstacles, light grey: free space, dark grey: unexplored area, white: $\mathcal{L}'$, star: HOMER, disk: most likely recipient location. (a) HOMER starts and (b) navigates to the first location. Having not found the recipient there, he (c) navigates to the second location, finds the recipient and navigates home.

people by their faces. The first person he recognized, person 1, was taken as the sender of the message, as shown at time 1 in Figure 4. The messages consisted solely of a recipient's name, person 2, who HOMER set out to find. Figure 5 (a) shows HOMER's map, the location likelihood function at the start of the run, and HOMER's initial position. The occupancy grid map is shown with obstacles marked as black, free space as light grey, and unexplored space as dark grey. The location likelihood function, $\mathcal{L}'$, is shown in white. HOMER is shown as a star. while the most likely recipient location is shown as a disk. HOMER proceeds to the most likely recipient location, as shown in Figure 5 (b). Figure 4 shows how, at time 2, HOMER found some unknown person there (not in his face database), which prompted him to move to a second location, at which he successfully recognized the recipient, person 2, as shown in Figure 4 at time 3. After delivering the message, HOMER returned home. The complete robot trajectory is shown in Figure 5 (c). We also enabled HOMER to receive a return message from the recipient, at which point the recipient becomes the sender, and the process repeats. We also performed experiments where HOMER is not able to locate the recipient, at which point he returns to the sender and reports the situation.

Conclusion and Future Work

In this paper, we presented the Human Oriented MEssenger Robot (HOMER). HOMER is a robot designed to solve the problem of message delivery among people in a workplace. Even though the robot is designed with a specific task in mind, we are using algorithms and software in such a way as to enable us to create,in a straightforward manner, other robots to perform different tasks. Such ease comes from building re-usable software components in a distributed control architecture while task specification is done at a high level by an expert of the domain using an MDP-based planner. We have presented experimental results of HOMER successfully receiving and delivering a simple message in a dynamic environment.

In the near future, we plan to integrate components from our previous work in sound localization, gesture recognition and facial expression analysis and improve the face recognition module. We are also in the process of developing a module for creating 3D occupancy grid maps that we hope to use for better navigation, people finding and interaction. Later, we will focus on adding a natural language understanding module in order to remove the restriction on the interaction among people and the robot.

References

Arkin, R. C. 1998. *Behavior-based Robotics*. MIT Press.

Bennewitz, M.; Burgard, W.; and Thrun, S. 2002. Learning motion patterns of persons for mobile service robots. In *Proc. of the IEEE International Conference on Robotics & Automation (ICRA '02)*.

Brooks, R. 1986. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation* 2(1):14–23.

Burgard, W.; Cremers, A.; Fox, D.; Hahnel, D.; Lakemeyer, G.; Schulz, D.; Steiner, W.; and Thrun, S. 1998. The interactive museum tour-guide robot. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI '98)*.

Dellaert, F.; Burgard, W.; Fox, D.; and Thrun, S. 1999. Using the condensation algorithm for robust, vision-based mobile robot localization. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR'99)*.

Elinas, P., and Little, J. J. 2002. A robot control architecture for guiding a vision-based mobile robot. In *Proc. of AAAI Spring Symposium in Intelligent Distributed and Embedded Systems*.

Elinas, P.; Hoey, J.; Lahey, D.; Montgomery, J.; Murray, D.; Se, S.; and Little, J. J. 2002. Waiting with Jose, a vision based mobile robot. In *Proc. of the IEEE International Conference on Robotics & Automation (ICRA '02)*.

Hoey, J.; St-Aubin, R.; Hu, A.; and Boutilier, C. 1999. SPUDD: Stochastic planning using decision diagrams. In *Proceedings of International Conference on Uncertainty in Artificial Intelligence (UAI '99)*.

Montemerlo, M.; Pineau, J.; Roy, N.; Thrun, S.; and Verma, V. 2002. Experiences with a mobile robotic guide for the elderly. In *Proceedings of the AAAI National Conference on Artificial Intelligence (AAAI '02)*.

Murray, D., and Little, J. 2000. Using real-time stereo vision for mobile robot navigation. *Autonomous Robots* 8:161–171.

Puterman, M. L. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* New York, NY.: Wiley.

Se, S.; Lowe, D.; and Little, J. J. 2002. Mobile robot localization and mapping with uncertainty using scale-invariant landmarks. *International Journal of Robotics Research* 21(8):735–758.

Simmons, R., and Koenig, S. 1995. Probabilistic robot navigation in partially observable environments. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI '95)*, 1080–1087.

Theocharous, G.; Rohanimanesh, K.; and Mahadevan, S. 2001. Learning hierarchical partially observable Markov decision process models for robot navigation. In *Proc. of the IEEE International Conference on Robotics & Automation (ICRA '01)*.

Thrun, S.; Bennewitz, M.; Burgard, W.; Cremers, A.; Dellaert, F.; Fox, D.; Hahnel, D.; Rosenberg, C.; Roy, N.; Schulte, J.; and Schulz, D. 1999. Minerva: A second-generation museum tour-guide robot. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA'99)*.

Thrun, S. 2000. Probabilistic algorithms in robotics. *AI Magazine* 21(4):93–109.

Turk, M., and Pentland, A. 1991. Eigenfaces for recognition. *Journal of Cognitive Neuroscience* 3(1):71–86.

Viola, P., and Jones, M. 2001. Rapid object detection using a boosted cascade of simple features. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR '01)*.