Machine Vision and Applications manuscript No.
(will be inserted by the editor)

MOBSY: Integration of Vision and Dialogue in Service Robots

Matthias Zobel, Joachim Denzler, Benno Heigl, Elmar Nöth, Dietrich Paulus, Jochen Schmidt, Georg Stemmer

Lehrstuhl für Mustererkennung, Universität Erlangen-Nürnberg
Martensstr. 3, 91058 Erlangen, Germany
e-mail: info@immd5.informatik.uni-erlangen.de,
URL: http://www5.informatik.uni-erlangen.de

Received: date / Revised version: date

Abstract This contribution introduces MOBSY, a fully integrated autonomous mobile service robot system. It acts as an automatic dialogue based receptionist for visitors of our institute. MOBSY incorporates many techniques from different research areas into one working stand-alone system. The involved techniques range from computer vision over speech understanding to classical robotics.

Among the two main aspects vision and speech we focus also on the integration aspect both on the methodological and on the technical level. We describe the task and the involved techniques. Finally, we discuss the experiences that we gained with MOBSY during a live performance at our institute.

Key words: Service robots – Computer vision – Speech understanding and dialogue – Integrated system

1 Introduction

In service robots many different research disciplines are involved, e.g. sensor design, control theory, manufacturing science, artificial intelligence, and also computer vision and natural language dialogue. The latter two are especially important since service robots should serve as personal assistants. As a consequence service robots differ from other mobile robotic systems mainly by their intensive interaction with people in natural environments. In typical environments for service robots, like hospitals or day care facilities for elderly people, the demands on the interface between robot and humans exceed the capabilities of standard robotic sensors, like sonar, laser, and infra-red sensors. Thus, in many cases computer vision as well as natural language dialogue components become essential parts of such a system.

In this paper we mainly concentrate on the following two aspects of a service robot: computer vision and natural language dialogue. We show a particular example application to demonstrate how a mobile platform becomes a service robot by integrating current research results from both areas into one system. In contrast to other systems, e.g. [5, 18], we neither concentrate on the technical design of a mobile platform, nor on the learning ability of a mobile system in general [9]. Instead, we are mainly interested in the integration of vision and speech to improve the capabilities of such systems, and even to increase them.

The proposed integration mechanism is intentionally kept as simple as possible. It is this simplicity that allows the transfer and scaling of the techniques to other applications without big effort. This is extremely useful in science and research, where rapid prototyping of real systems is of great interest, especially for experimental evaluation of achieved theoretical results. Beyond this, software modules that implement a solution of a dedicated problem (e.g. visual self-localization or speech recognition) are often developed independently in their very own framework with a specialized set of tools being particularly suitable. Our integration mechanism is able to embed such modules in a common environment but keeps their individuality.

Currently, we provide a fully functional human-machine-interface by natural language processing that cannot be found for systems like MINERVA [36] or RHINO [9]. Several systems are known that include speech as one means for a human-machine-interface (e.g. [6]); they mostly use simple spoken commands. We provide a real dialogue component in our system that takes as input spoken language and thus allows for the most natural way of communication. An active vision system is used for localization of the platform in a natural environment without the need of adding artificial markers in the scene. Additionally, events are recognized based on the visual information acquired by the binocular camera system. Thus, the camera system is essential for the robot's functionality, and not just an anthropomorphic feature.

The paper is structured as follows. In the next section we formulate the task that we want MOBSY to execute. In Section 3 we shortly describe the involved techniques of our system, i.e. the computer vision, dialogue, and robotic modules. Especially, we emphasize self-localization based on visual information, because in general this is a typical problem for the classical sensors. Technical details on the whole service robot MOBSY are given thereafter in Section 4 as well as a short discussion on the integration process. We conclude in Section 5 with results and experiences that we gained when MOBSY was in action. Finally, we give an an outlook to future improvements and applications.

2 Task Description

As a test bed for our developments we chose a setup as can be seen in Fig. 1. The scenario is an indoor area at our institute in front of the elevators. In this environment we want MOBSY to act as a mobile receptionist for visitors, i.e. it has to perform the following steps (cf. Fig. 2):

- MOBSY waits at its home position for one of the three doors of the elevators to open. It moves its head to see the doors in the sequence *left*, *middle*, *right*, *middle*, *left*, ...
- If a person arrives, MOBSY approaches him/her on the paths that are shown as lines in Fig. 1; during the approach it already addresses the person, introducing itself as a mobile receptionist, and asks the person not to leave.
- After arrival in front of the person, it starts a natural language information dialogue. Simultaneously, the binocular camera system starts to track the person's head to initiate a first contact.
- When the dialogue is finished, MOBSY turns around and returns to its home position where it has to reposition itself, as the odometric information is not very accurate.
- Then the waiting process for a new person to arrive resumes.

This main loop is repeated until MOBSY is stopped externally. Accomplishing the previously described steps requires the coordinated combination of

- object detection and classification,
- visual face tracking and camera control,
- natural language dialogue,
- robot navigation including obstacle avoidance, and
- visual self-localization and recalibration.

The methods we used for these five areas are described in more detail in the following section.

3 Modules

Object classification. For our scenario we expect visitors of our institute to arrive by one of three elevators. It follows, that the arrival of a person is necessarily preceded by the opening of one of the elevator doors. Therefore we use a person indicator mechanism based on distinguishing between open and closed elevator doors.

For that purpose we decided to use a support vector machine (SVM) as the classification technique that is predestinated for solving two-class problems (cf. [34] for detailed description). The SVM takes as input color images of size 96×72 of the doors of the elevators and it returns *open* or *closed* as a result.

For training the SVM we compiled a training set of 337 images of elevator doors: manually labeled into 130 *closed* and 207 *open*. An elevator door is regarded as *open* in the range from open to half open, otherwise *closed*. The training phase results in 41 support vectors that determine the discriminating hyperplane between the two classes. We used SVM$^{\text{light}}$ [22] for the implementation of a SVM framework.

Of course, an open door is not sufficient to decide for the arrival of a person. Think of a visitor that wants to depart from the institute and an elevator door opens to pick him up, or think of the situation of open doors but the elevator is empty. In our current implementation such a detection error would cause MOBSY to start to approach to that person, too. Therefore, this situation has to be intercepted in an appropriate way.

Face tracking. While MOBSY approaches an arrived person and during the dialogue phase both cameras of the binocular vision system should fixate on the person's face to maintain contact. If the person moves slightly the cameras should maintain the fixation. This makes the person feel the attention of MOBSY is focused on him. It could also be used for the system to validate visually if there is still a person it should serve, or if the person is already gone. Another aspect that is not yet realized is that recognition of faces could also take place during tracking.

Therefore two major problems must be solved: face detection and controlling the motion of the cameras. Face detection is based on discriminating skin colored regions from other areas in the images [11] by computing a color distance for each pixel. To reduce computation time we are using an image resolution of 96×72 pixels. The center of gravity of the skin colored pixels is assumed to be the position of the face in the image. From the determined face positions in each camera, steering commands can be calculated for the tilt and vergence axes of the binocular system to bring the face's positions into the centers of the images. Attention must be payed to the fact that no independent tilt motion is possible because of the mechanical constraints of the binocular camera system. To keep the motions smooth and to let them look more natural, the vergence motions are compensated by appropriate motions of the pan axis.

It is obvious, that skin color segmentation is not very specific to faces. But the following facts justify our choice. First, it is very likely that detected skin color in a height between approximately 1.60 m and 1.80 m is caused by a face, and second, the algorithm works very fast and robust.

Dialogue. When the robot has reached its position in front of the person, the dialogue module initiates the conversation with a greeting and a short introduction into MOBSY's capabilities. Our goal was to enable the robot to perform a natural language conversation with the user. For this purpose, the dialogue module consists of four sub-units which form a processing pipeline: for each utterance, the speech recognizer computes an hypothesis of the spoken word sequence. The word sequence is transformed into a semantic-pragmatic

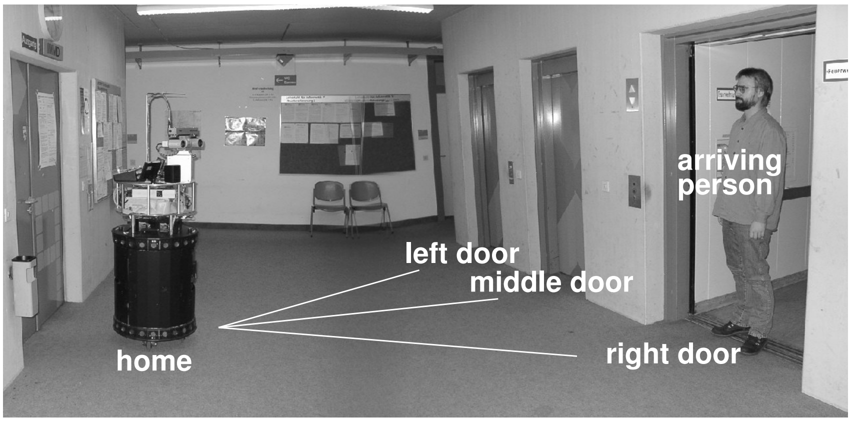

Fig. 1 Environmental setup for the task.

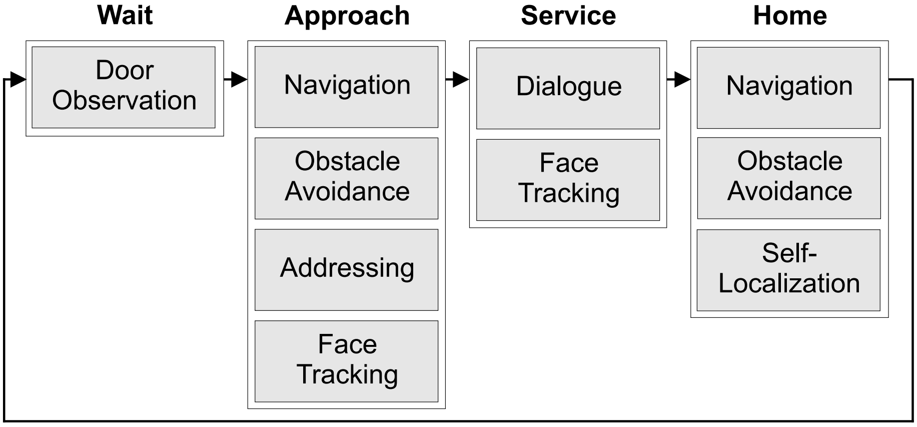

Fig. 2 Flowchart describing the main loop.

representation by the language understanding unit. In combination with the current dialogue state, the semantic-pragmatic representation is used by the dialogue management for answer generation. The generated system prompt is transformed into an audible result by the speech synthesis unit. All sub-units of the dialogue module have to be designed to deal with high noise levels as well as with a diversity of the person's utterances. The noise level is partly caused by the environment of the robot, for example, the elevator doors, other people on the floor, and partly by the robot itself, because of its several built-in fans and electric motors. The person's utterances have an unusually high degree of diversity, because of MOBSY's job to welcome visitors. In addition to this, MOBSY is among the first 'persons' a visitor of the institute meets, so visitors usually do not get any introduction into the system, except by MOBSY itself. In the following, we describe in more detail, how the sub-units of the dialogue module meet these challenges (cf. Fig. 3).

In order to remove the background noise before and after the user's utterance, speech recognition starts only if the energy level in the recorded signal exceeds a threshold for a predefined duration and stops immediately after the energy level falls below a threshold for more than a fixed amount of time. High-frequency noise gets eliminated by a low-pass filter. Our robust continuous speech recognizer with a lexicon of 100 words uses mel-cepstrum features and their derivatives. A more detailed description of the recognizer can be found in [15, 16]. We initialized the acoustic models of the recognizer on training data of a different domain and adapted them to the scenario with approx. 900 utterances of read speech. The recognizer uses a simple bigram model. The language understanding unit searches for meaningful phrases in the recognized word sequence. For additional information, please refer to [28]. Each phrase has a predefined semantic-pragmatic representation. Words that do not belong to meaningful phrases are ignored by the language understanding unit. This simple strategy results in a very high robustness to smaller recognition errors and user behavior. The dialogue management contains a memory for the state of the dialogue. It uses rules to choose a suitable system prompt based on the dialogue memory and the current input. For example, if the user asks, "Where can I find it?", the system provides information on the location of the item that it was asked for in a previous question. If the meaning of the recognized sentence does not fit to the dialogue memory, an error of the speech recognizer is assumed, and an appropriate answer is generated. In order to prevent the robot from stupidly repeating always the same prompts for greeting, etc. most system prompts are represented by a set of several different pre-recorded speech files. One speech file for the current prompt is chosen randomly

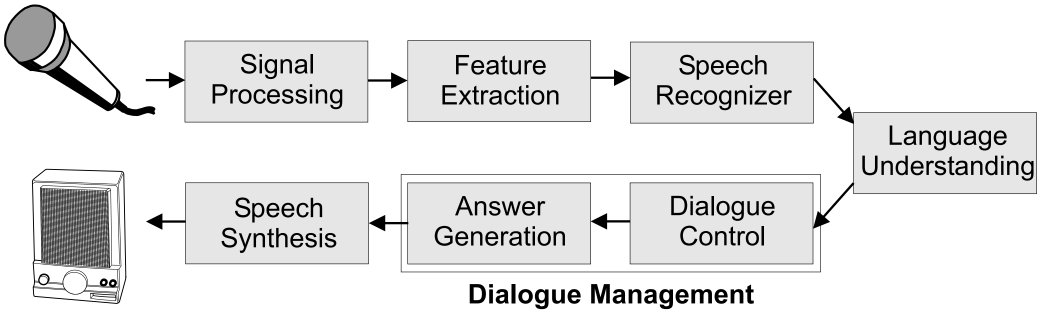

Fig. 3 A closer look on the dialogue module.

and played. For maximum intelligibility, we decided to use the German Festival speech synthesis system [7,25] to record the system prompt speech files.

Navigation and obstacle avoidance. If MOBSY recognizes an arriving person, the robot moves straight to a predefined position in front of the door (cf. Fig. 1). While moving, MOBSY looks in the direction of its movement, thus facing the person waiting at the elevator. After the dialogue phase MOBSY returns straight to its initial position. Currently, the navigation tasks are only implemented in a rudimentary way, i.e. no path planning nor other intelligent strategies are used. They will be integrated in a later stage of the project.

With the robot operating in an environment where many people are present, obstacle avoidance is essential. Non-detected collisions could cause serious damage, because of the robot's considerable weight. We use the infra-red and tactile sensors for the detection of obstacles. Thus, we are able to detect persons at a distance up to 30 - 50 cm away from the robot, depending on the illumination of the room and the reflectivity of the peoples' clothes. The tactile sensors are used for reasons of safety only, i.e. they react as a last instance in cases where the infra-red sensors fail. If an obstacle is detected, MOBSY stops immediately and utters a warning that it cannot move any further.

Self-localization. For self-localization we use a neon tube that is mounted at the ceiling. By analyzing an image of the lamp we can calculate its direction as well as its position relative to the robot. Knowing the desired direction and position from measurements in advance, correction movements can be determined to reposition the robot to its true home.

Fig. 4 shows the 3-D configuration that is used here. The position of the lamp is defined by the end point $\boldsymbol{p}_1$ and a second distinct vector $\boldsymbol{p}_2$ on the lamp which can be chosen arbitrarily. We move one of the stereo cameras such that it points to the presumed position of $\boldsymbol{p}_1$ and take an image as shown in Fig. 5. If the lamp is not fully visible in the first view we perform a heuristic search for it. The extraction of the projections $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ of $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ into the image can be done easily by analyzing the binarized image. Note that $\boldsymbol{q}_2$ may be located arbitrarily on the line. We approximate a line by doing linear regression using the bright points and find the visible end point by a simple search along this line.

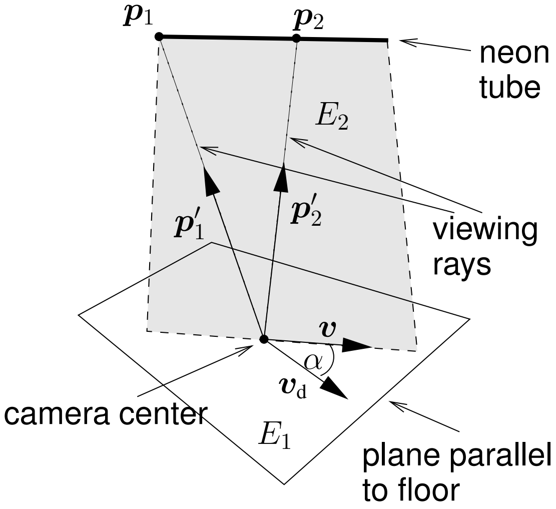

Fig. 4 3-D configuration for self-localization.

The coordinate system is defined to be such that its origin corresponds to the projection center of the camera, the z-axis is perpendicular to the floor, and the y-axis points to the front of the robot. It is also assumed that the camera center is at the intersection of the pan and tilt axes of the binocular camera system and that it also intersects the rotation axis of the robot. In reality, these axes do not intersect exactly, but this approximation works fine in our experiments.

The plane E_1 is defined to be parallel to the floor. The plane E_2 contains both, the origin and the line that describes the lamp. The vector $\boldsymbol{v}$ is defined to point in the direction of the intersection of these two planes and can be calculated by the formula $\boldsymbol{v} = (\boldsymbol{p}'_1 \times \boldsymbol{p}'_2) \times (0,0,1)^\mathrm{T}$.

From our setup we can measure the desired coordinates $\boldsymbol{p}_\mathrm{d}$ of the lamp's end point relative to the coordinate system and also the desired direction $\boldsymbol{v}_\mathrm{d}$ of the lamp (in our example $\boldsymbol{v}_\mathrm{d} = (0,-1,0)^\mathrm{T}$). If the robot would be located at the desired position, $\boldsymbol{p}_\mathrm{d}$ would point to the same direction as $\boldsymbol{p}'_1$ and $\boldsymbol{v}_\mathrm{d}$ to the same direction as $\boldsymbol{v}$. If they are not the same, the robot must be rotated by the angle $-\alpha$. The necessary corrective translation can be determined by rotating $\boldsymbol{p}'_1$ by α around the z-axis, scaling the result to the size of $\boldsymbol{p}_\mathrm{d}$, and subtracting $\boldsymbol{p}_\mathrm{d}$.

For the general case of vision based localization and navigation we already presented a method using lightfields as

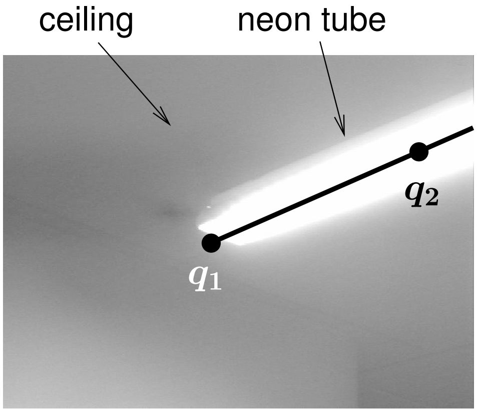

Fig. 5 Image of a lamp used for self-localization.

scene models [21] and particle filters for state estimation [20]. Since currently a self-localization is necessary only when the robot is back in its home position, a iterative self-localization over time is not suited. Later on, when the robot is also used to guide visitors to offices of employees, we will substitute the current module with the lightfield based localization framework.

4 System Design

In this section we present the design of our autonomous mobile platform in more detail. At first we explain the most important hardware components, followed by a description of the software integration process.

Robot hardware. MOBSY is designed to be completely autonomous. All equipment for proper operation must be carried by MOBSY, e.g. additional batteries, axis controllers, speakers, etc.

The freedom in system design is restricted by the mobile platform itself, i.e. the lower part of MOBSY that contains the drive. The robot is a cylindrically shaped XR4000 from Nomadic Technologies. Inside the platform there are two PCs running under Linux. One of the PCs is responsible for motion and sensor control, the other PC, a dual Pentium II 300 MHz, is used for all high level tasks like vision, path planning, speech recognition.

On top of this “out of the box” platform XR4000, we mounted a two-storied rack for carrying all additional equipment (cf. Fig. 6), because there is no room left inside the platform itself. Especially all components of the vision system (controllers for the head’s axes, cameras, batteries, the head itself) require most of the available space. Integrating the vision system was also the most challenging part in our system design.

One advantage of the chosen rack solution is that the “level of interaction” is lifted from the platform’s height to one that is more similar to a human being. Among other aspects, “normal” height may yield an increased acceptance of the robot by the users, because, at least in this point, users do not need to adapt to the robot. Possibly this lets a person forget a little bit about that he is interacting with a robot, and not with a human vis-a-vis.

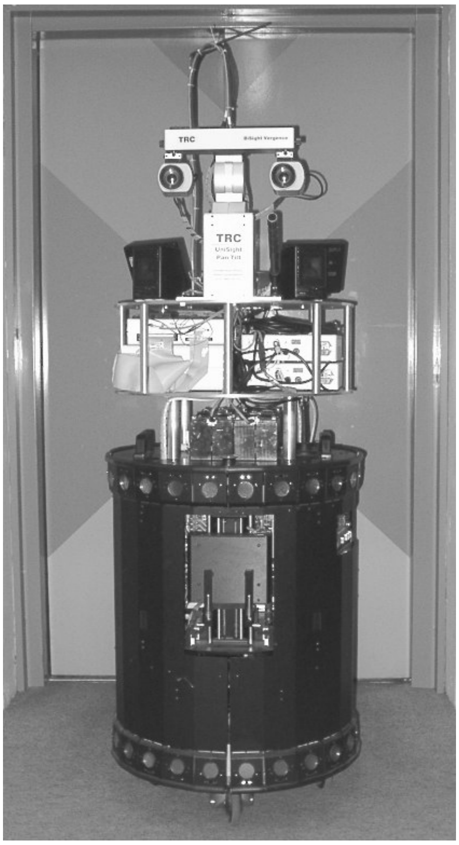

Fig. 6 MOBSY the autonomous mobile service robot.

The vision system that we use is a 10 DOF Bisight/Unisight binocular system from HelpMate Robotics. Because no display is connected to the PCs while the platform is in operational mode, and therefore no images from the cameras could be investigated online, we mounted two small LCD panels for monitoring what the cameras see.

Also on the most upper level of the rack we mounted the interface for the dialogue component, i.e. microphone and two loudspeakers. The microphone has a high degree of directivity that helps to make the speech recognition process more robust against high background noise.

Software integration. Programming or configuring a pattern analysis system requires specialized knowledge about the effects of signal processing algorithms as well as knowledge of the implementation and interfaces. Several programming languages and programming systems are available for this task; many systems were written in C (e.g. [14,29]), Java (e.g. [23]), Fortran (e.g. SPIDER [35]), etc. The programming language C++ has been used widely, e.g. in the context of an iconic kernel system IKS [17] and the activities that lead to the international standard PIKS [8,10,1,33]. Speech processing is mostly done in C, e.g. for the HTK system [37].

Several libraries for image processing routines are available for each of these languages, e.g. the PIKS standard mentioned above, or the libraries in the image understanding environment [19,26]. The larger the application gets that uses such libraries, the higher is the importance of well-structured software.

Summarizing from the modules and algorithms described in the previous sections, the requirements for a common software platform include implementations of Hidden Markov Models for word recognition, statistical classifiers for images and speech, hardware interfaces for digitizers of sound and video, controllers for motors, filters for 1D and 2D signals, Kalman and particle filters for tracking objects in images or to model temporal events in sound, just to mention some.

As vision and speech analysis is carried out from the signal level up to an interpretation, a knowledge-based or model driven system is required that should also be shared wherever possible. High demands on efficiency result from the real-time constraints imposed to parts of the processing chain.

As the flowchart in Fig. 2 indicates, there exist two levels on which an integration of software is necessary.

- The task level, indicated in Fig. 2 by the boxes named *Wait*, *Approach*, *Service*, and *Home* and the arrows that connect them.
- The modules level that is indicated by the gray shaded boxes.

Depending on the level, different integration techniques are appropriate.

On the most abstract level, the task level, the overall behavior of the system is determined and integration focuses mainly on the combination of the modules to build the correct sequence. The tasks are solved sequentially. In our system, the module sequence is scheduled and synchronized by the Unix shell. For example, the module *Addressing* is started when the module *Door Observation* reports an arrived person. Parallelism is achieved by the Unix process mechanisms that allow of background processes and the controlled stopping of processes by signaling. Proceeding like this has one important advantage: exchanging modules is relatively simple. We experienced this, when we had to replace our first revision of the door detection that used color histograms by the more effective technique based on SVM.

On the second level, the modules level, that defines the functionality of the system, integration is not that simple. The modules are executed simultaneously and require synchronization. For example, for face tracking computer vision and active camera control have to be carried out in one process. As each module represents a research task on its own in our institute, a separate solution is required for each. On the other hand, as re-inventing the wheel too many times delays real scientific and technical progress, the contributors were urged to share software from the very beginning of their implementation. For over a decade such cooperation lead to integration of partially divergent work, as e.g. shown for object recognition and tracking in [13].

As we have shown in [31,32], class hierarchies in C++ can be used to encapsulate actors that are used in active vision. As our system is equipped with an active camera, these classes are used to rotate camera axes. A similar abstract interface is used for robot control and hides controller details; some details on the class hierarchy for these hardware interfaces can be found in [30]. This is useful for the navigation and obstacle avoidance module. Robot control and image analysis have thus been successfully integrated. Synchronization and message passing on the control level for robotic hardware have simply been possible by the use of threads. Other class hierarchies provide the common interfaces for the required algorithms mentioned in the beginning of this section, such as classifiers and Hidden Markov Models. Currently no knowledge base is used in the vision system of our autonomous robot, but a common knowledge-based system applied simultaneously in vision and speech has already been established, as demonstrated in [2,27]. The modules of our vision software system are to some extent portable to other software systems, as the experiments on the system ADORE [4] prove, when this system was used to work with our algorithms to generate hypotheses for object recognition in an office room [3].

Currently, there has to be little time-critical interaction and information exchange between the dialogue and the face tracking modules. Therefore we preferred to separate the dialogue part from the robotic and vision tasks. It turned out to be sufficient to run the dialogue and face tracking processes in parallel on the task level.

5 Conclusion and Future Work

First we would like to mention that our system operated non-stop for more than two hours without any human intervention at the 25th anniversary of our institute. The robot had to deal with many people coming out of the elevator or standing around while talking to each other, thus generating a high noise level (cf. Fig. 7; many more images and movies are available from the web site [24]). The robustness is regarded as a great success, especially since the whole integration process took only about two man-months.

In this contribution we introduced MOBSY as an example of an autonomous mobile service robot, acting as a receptionist for visitors of our institute. Research results from many different areas were combined into one fully operational system. In addition to the performance at the anniversary, in the meanwhile MOBSY demonstrated its robustness as autonomous receptionist many times, not only for invited visitors of the institute but also for groups of students and pupils. One can really say that MOBSY is an attraction that transforms research to a practical experience.

Increasingly, the aspect of smart interaction with people plays an important role in service robotics. Therefore natural language dialogue and computer vision components have to be integrated with classical robotic sensors. Choosing an appropriate integration policy is a crucial point for building a running and robust system of these very diverse components. Especially, if the modules that become integrated were developed independently of each other. With MOBSY being

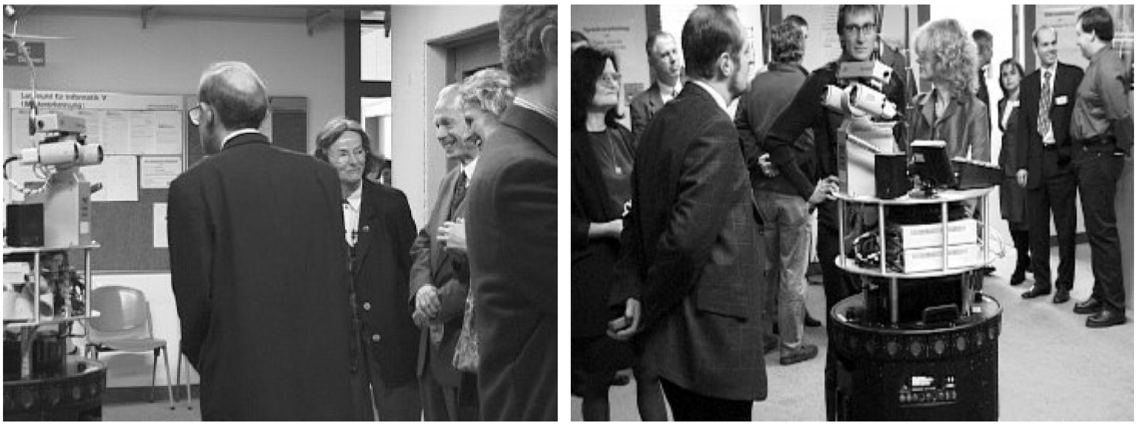

Fig. 7 MOBSY operating at the 25th anniversary.

a success as a mobile receptionist and regarding the time it took to get MOBSY working as expected, we have demonstrated that integration does not necessarily need to be highly sophisticated to yield a complex system behavior. Quite the contrary, we think that simplicity of integration keeps the system open to further developments and extensions, because modules can be easily exchanged and new modules can be embedded without big effort.

Safety is one of the major topics when autonomous mobile systems deal with people, especially if there are many of them. Due to the short range of the used infra-red sensors we were forced to move MOBSY at relatively low speed, so that the platform stops early enough in front of a detected obstacle or person. This leads to the drawback that it takes a relatively long time, approximately five seconds, for the robot to reach its destinations in front of the elevator doors. People arriving at the institute may leave the elevator and go away because they are not aware of the mobile receptionist. Therefore we introduced that MOBSY addresses the person immediately after it was detected. This reduced the number of situations where the dialogue module started to talk to a closed elevator door. If however this case happened, a timeout in the dialogue module recognizes the absence of a person and initiates the homing of the platform. This simple attention mechanism will be replaced in the future by a visual module that checks for the presence of people.

The definition of evaluation criteria for the performance of a system like MOBSY is not a trivial task. Of course it would be possible to evaluate the reliability and accuracy of each of the system's subunits, but there exist additional aspects concerning the whole system that cannot be expressed by a simple number, for example, the acceptance by the users. Currently, this topic remains under further investigations.

In the future we will extend the capabilities of MOBSY. Beside the classical robotic tasks we will especially focus on the vision and dialogue components of the robot. For example, we want the platform to guide visitors to the employees or to special places of the institute, based on visual tracking and navigation. Beyond pure navigation our robot should be able to press the button to call the elevator, if someone wants to leave the institute. Pressing the button requires the ability of MOBSY to visually localize it and to press it with a gripper.

Acknowledgements This work was supported by the "Deutsche Forschungsgemeinschaft" (DFG) under grant SFB603/TP B2,C2 and by the "Bayerische Forschungsstiftung" under grant DIROKOL.

References

1. International Standard 12087. Image processing and interchange. Technical report, International Standards Organization, Genf, CH, 1994.
2. U. Ahlrichs, J. Fischer, J. Denzler, Ch. Drexler, H. Niemann, E. Nöth, and D. Paulus. Knowledge based image and speech analysis for service robots. In *Proceedings Integration of Speech and Image Understanding*, pages 21–47, Corfu, Greece, 1999. IEEE Computer Society.
3. D. Paulus B. Draper, U. Ahlrichs. Adapting object recognition across domains: A demonstration. In B. Schiele and G. Sagerer, editors, *Computer Vision Systems, Second International Workshop, ICVS 2001 Vancouver, Canada, July 7-8, 2001, Proceedings*, pages 256–267, Heidelberg, 2001. Springer.
4. J. Bins B. Draper and K. Baek. Adore: Adaptive object recognition. In Christensen [12], pages 522–537.
5. R. Bischoff. Recent advances in the development of the humanoid service robot hermes. In *3rd EUREL Workshop and Masterclass - European Advanced Robotics Systems Development*, volume I, pages 125–134, Salford, U.K., April 2000.
6. R. Bischoff and T. Jain. Natural communication and interaction with humanoid robots. In *Second International Symposium on Humanoid Robots*, pages 121–128, Tokyo, 1999.
7. A. Black, P. Taylor, R. Caley, and R. Clark. The festival speech synthesis system, last visited 9/26/2001. http://www.cstr.ed.ac.uk/projects/festival/.
8. Christof Blum, Georg Rainer Hofmann, and Detlef Kromker. Requirements for the first international imaging standard. *IEEE Computer Graphics and Applications*, 11(2):61–70, March 1991.
9. W. Burgard, A.B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun. The interactive museum tour-guide robot. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pages 11–18, Madison, Wisconsin, July 1998.

10. T. Butler and P. Krolak. An overview of the Programmer's Imaging Kernel (PIK) proposed standard. *Computers and Graphics*, 15(4):465–472, 1991.
11. Douglas Chai and King N. Ngan. Locating facial region of a head-and-shoulders color image. In *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*, pages 124–129, Nara, Japan, 1998. IEEE Computer Society Technical Commitee on Pattern Analysis and Machine Intelligence (PAMI).
12. H. Christensen, editor. *Computer Vision Systems, First International Conference, ICVS '99 Las Palmas, Gran Canaria, Spain, January 13-15, 1999 Proceedings*, Heidelberg, 1999. Springer.
13. J. Denzler, R. Beß J. Hornegger, H. Niemann, and D. Paulus. Learning, tracking and recognition of 3D objects. In V. Graefe, editor, *International Conference on Intelligent Robots and Systems – Advanced Robotic Systems and Real World*, volume 1, pages 89–96, München, 1994.
14. M. R. Dobie and P. H. Lewis. Data structures for image processing in C. *Pattern Recognition Letters*, 12:457–466, 1991.
15. F. Gallwitz. *Integrated Stochastic Models for Spontaneous Speech Recognition*. Studien zur Mustererkennung. Logos Verlag, Berlin, 2002. (to appear).
16. F. Gallwitz, M. Aretoulaki, M. Boros, J. Haas, S. Harbeck, R. Huber, H. Niemann, and E. Nöth. The Erlangen Spoken Dialogue System EVAR: A State-of-the-Art Information Retrieval System. In *Proceedings of 1998 International Symposium on Spoken Dialogue (ISSD 98)*, pages 19–26, Sydney, Australia, 1998.
17. P. Gemmar and G. Hofele. An object–oriented approach for an iconic kernel system IKS. In *Proceedings of the* 10^{th} *International Conference on Pattern Recognition (ICPR)*, volume 2, pages 85–90, Atlantic City, 1990. IEEE Computer Society Press.
18. U. Hanebeck, C. Fischer, and G Schmidt. Roman: A mobile robotic assistant for indoor service applications. In *Proceedings of the IEEE RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 518–525, 1997.
19. R. M. Haralick and V. Ramesh. Image understanding environment. In R. B. Arps and W. K. Pratt, editors, *Image Processing and Interchange: Implementation and Systems*, pages 159–167, San Jose, CA, 1992. SPIE, Proceedings 1659.
20. B. Heigl, J. Denzler, and H. Niemann. Combining computer graphics and computer vision for probabilistic visual robot navigation. In Jacques G. Verly, editor, *Enhanced and Synthetic Vision 2000*, volume 4023 of *Proceedings of SPIE*, pages 226–235, Orlando, FL, USA, April 2000.
21. B. Heigl, R. Koch, M. Pollefeys, J. Denzler, and L. Van Gool. Plenoptic modeling and rendering from image sequences taken by a hand-held camera. In W. Förstner, J.M. Buhmann, A. Faber, and P. Faber, editors, *Mustererkennung 1999*, pages 94–101, Heidelberg, 1999. Springer.
22. Th. Joachims. Making large-scale support vector machine learning practical. In Schölkopf et al. [34], pages 169–184.
23. Douglas A Lyon and Hayagriva V. Rao. *Java Digital Signal Processing*. M&T Books, M&T Publishing, Inc., 501 Galveston Drive, Redwood City, CA 94063, USA, November 1997.
24. http://www5.informatik.uni-erlangen.de/˜mobsy.
25. Gregor Möhler, Bernd Möbius, Antje Schweitzer, Edmilson Morais, Norbert Braunschweiler, and Martin Haase. The german festival system, last visited 9/26/2001. http://www.ims.uni-stuttgart.de/phonetik/synthesis/index.html.
26. J. Mundy, T. Binford, T. Boult, A. Hanson, R. Veveridge, R. Haralick, V. Ramesh, C. Kohl, D. Lawton, D. Morgan, K Price, and T. Strat. The image understanding environments program. In *Image Understanding Workshop*, pages 185–214, San Diego, CA, Jan. 1992.
27. H. Niemann, V. Fischer, D. Paulus, and J. Fischer. Knowledge based image understanding by iterative optimization. In G. Görz and St. Hölldobler, editors, *KI–96: Advances in Artificial Intelligence*, volume 1137 (Lecture Notes in Artificial Intelligence), pages 287–301. Springer, Berlin, 1996.
28. E. Nöth, J. Haas, V. Warnke, F. Gallwitz, and M. Boros. A hybrid approach to spoken dialogue understanding: Prosody, statistics and partial parsing. In *Proceedings European Conference on Speech Communication and Technology*, volume 5, pages 2019–2022, Budapest, Hungary, 1999.
29. J.R. Parker. *Algorithms for image processing and computer vision*. Wiley computer publishing, New York, 1997.
30. D. Paulus, U. Ahlrichs, B. Heigl, J. Denzler, J. Hornegger, and H. Niemann. Active knowledge based scene analysis. In Christensen [12], pages 180–199.
31. D. Paulus and J. Hornegger. *Applied pattern recognition: A practical introduction to image and speech processing in C++*. Advanced Studies in Computer Science. Vieweg, Braunschweig, 3rd edition, 2001.
32. D. Paulus, J. Hornegger, and H. Niemann. Software engineering for image processing and analysis. In B. Jähne, P. Geißler, and H. Haußecker, editors, *Handbook of Computer Vision and Applications*, volume 3, pages 77–103. Academic Press, San Diego, 1999.
33. W. K. Pratt. *The PIKS Foundation C Programmers Guide*. Manning, Greenwich, 1995.
34. B. Schölkopf, Ch. Burges, and A. Smola, editors. *Advances in Kernel Methods: Support Vector Learning*. The MIT Press, Cambridge, London, 1999.
35. H. Tamura. Design and implementation of spider - a transportable image processing software package. *Computer Vision, Graphics and Image Processing*, 23:273–294, 1983.
36. S. Thrun, M. Bennewitz, W. Burgard, A. Cremers, F. Dellaert, D. Fox, D. Hahnel, C. Rosenberg, J. Schulte, and D. Schulz. Minerva: A second-generation museum tour-guide robot. In *Proceedings of the IEEE International Conference on Robotics Automation (ICRA)*, pages 1999–2005, 1999.
37. S. Young, J. Jansen, J. Odell, D. Ollason, and P. Woodland. *The HTK Book*. Entropic Cambridge Research Laboratory Ltd., Cambridge, 1996.