A VISION-BASED ARCHITECTURE FOR LONG-TERM HUMAN-ROBOT INTERACTION

Christopher King, Xavier Palathingal, Monica Nicolescu, Mircea Nicolescu
Department of Computer Science and Engineering
University of Nevada, Reno NV 89557, USA
email: {cjking, xavier, monica, mircea}@cse.unr.edu

ABSTRACT
Advances in robotics research bring robots closer to real world applications. Although robots have become increasingly capable, productive interaction is still restricted to specialists in the field. In this paper, we propose an interactive architecture, based on visual capabilities, which allows robots to interact with people in a natural way, to deal with multiple users, and to be constantly aware of their surroundings. First, we endow our robot with visual capabilities, which allow it to detect when people are requesting to engage it in interaction. Second, we provide the robot with flexibility in dealing with multiple users, such as to accommodate multiple user-requests and task interruptions, over extended periods. The visual capabilities we propose allow the robot to identify multiple users, with multiple postures, in real-time and in dynamic environments, where both the robot and human users are moving. We demonstrate our approach on a Pioneer 3DX mobile robot, performing service tasks in a real-world environment.

KEY WORDS
Tracking, posture recognition, human-robot interaction.

1 Introduction

Robotics holds a tremendous potential for improving the quality of people's lives. In particular, robots for service and assistive applications will soon be within reach and would benefit a wide spectrum of users. A major challenge in designing robots for real-world applications is to enable natural and accessible interaction between robots and non-technical users, while ensuring long-term, robust performance in complex environments without the direct control of a human operator [1].

We propose a control architecture that provides robots with *social-awareness*, allowing them to monitor their surroundings for other social agents (robots or people), detect their need for interaction, and respond appropriately. Our architecture provides means for *long-term autonomy*, enabling robots to manage a large repertoire of tasks over extended periods. Additionally, our system is designed for realistic assistive applications, where multiple people may be simultaneously competing for the robot's assistance.

Vision-based perception provides the richest information required for effective human-robot interaction. In particular, the ability to distinguish between different people and to identify basic human postures is essential for the success of the interaction. While significant work has been done in these areas [2, 3], robust recognition routines are frequently too computationally expensive to be run in real time, especially when a robot must identify multiple object patterns in a dynamic, real-world environment. In this paper, we describe a real-time person identification and posture recognition algorithm and we demonstrate its use in interactive experiments using a mobile robot.

The contribution of this paper is a framework that addresses two key issues in human-robot interaction: *awareness* of the environment and other agents, and *long-term interaction* with multiple users.

The remainder of the paper is structured as follows: Section 2 presents our interactive framework, Section 3 discusses our vision-based perceptual approach and Section 4 describes the experimental setup and results. We present our conclusions in Section 5.

2 Interactive Framework

This work aims to increase a robot's autonomy over extended periods, while providing it with the perceptual awareness necessary for the assistive-application domain.

A service or personal robot will likely have to perform in the presence of multiple users, who may solicit services during overlapping intervals (one person may request a robot that is attending to another's task). To respond accordingly, the robot must interrupt its current activity, detect the new request, and determine which task to perform next. Our framework enables this functionality using linked *awareness* and *control* modules. The *awareness module* identifies known users and determines their needs by recognizing certain postures (described in Section 3). If a posture is detected for a sufficient duration, the information is relayed to the *control module*, which determines the appropriate action. Postures are associated with a list of tasks (robot services) that users can request. These tasks have an associated priority, such that when a posture is detected, the robot will perform the associated task, only if the priority of the new task exceeds the priority of any current activity. The task with the lower priority will be suspended and stored to a priority-based queue. Lower-priority tasks will be resumed when higher-priority tasks are completed.

Prioritized task switching resembles human decision-making behavior, and therefore, is a logical addition to the service robot domain. This provides a predictable and useful form of robot interaction that should feel natural to the user. This sense of natural robot-interaction can be further enhanced by using our posture-based control mechanism as described in the next section.

3 Vision-Based Human Posture Recognition

The role of the *visual awareness* module is to provide the robot with the capability of detecting the presence of humans that might be interested in interacting with the robot. Toward this end, we developed visual capabilities that can recognize human postures that are likely to be relevant to the robot-human interaction. The postures are described below and examples are shown in Fig. 1 (first row).

The Standing Posture – The most obvious posture to recognize as it is displayed frequently and is often an indication that a human is on the move or engaging in a task.

Arms-Up Posture – Humans learn at a young age that they can attract another's attention by raising their hand and a robot should respond accordingly.

Kneeling Posture – Since many robots are significantly smaller than humans are, one must crouch or kneel to pass an object to, or take an object from the robot's gripper. A robot should therefore recognize a crouching human and be able to determine if the human is holding an object.

Object Posture – Held-objects were trained independently from the human. This increases model robustness and allows the robot to orient itself toward the object.

3.1 Related Work in Visual Identification / Tracking

The identification and tracking of objects in a video feed is reasonably easy when a relatively static background can be maintained. In the simplest case, background pixels are modeled using a single video-frame, or they can be represented using Gaussian [4] or non-parametric distributions [5]. Models can be adaptive to slowly changing conditions [5, 6, 7], and even robust to smooth and linear camera movements [8]. In all these cases, pixels in subsequent frames are compared to the background model. If the pixel features (e.g., color, texture, motion) are inconsistent with the model, they are grouped and segmented as foreground.

Despite the success of background modeling techniques, they are unsuitable for use on a mobile robot in an uncontrolled environment. Camera movement is usually too complex to be stabilized by motion-modeling algorithms and the limitless variability of the shape, color, and texture of background features precludes the use of standard feature-based background models. Consequently, robotics applications typically use foreground-modeling techniques, which use object features to identify and track the object. Foreground modeling traditionally requires an offline acquisition period where values in the foreground models are assigned or trained. The robot is then switched

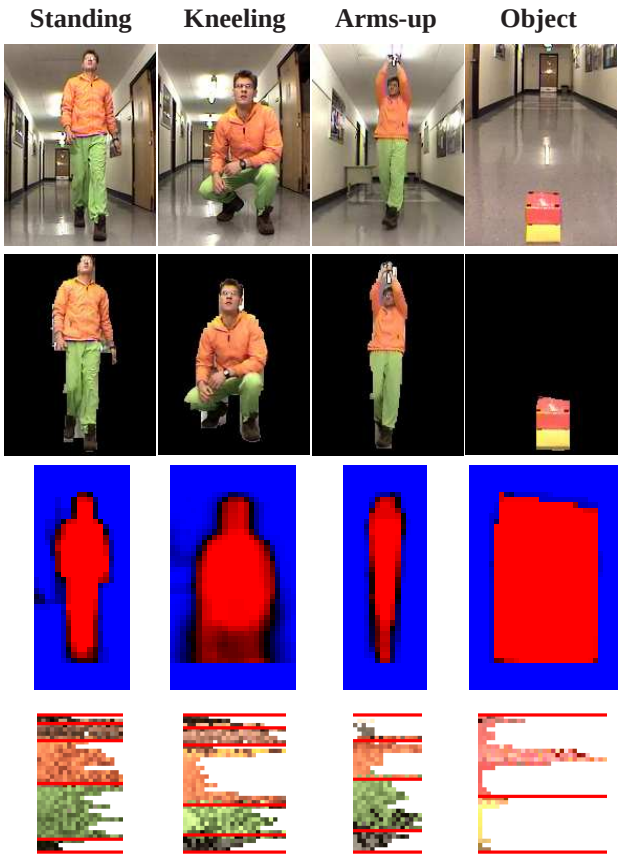


Figure 1. Set of postures. First row: original images. Second row: detected foreground. Third row: shape model. Fourth row: color model.

to a tracking phase, where it searches each incoming image for a region that is sufficiently similar to the model.

A foreground modeling technique commonly used in robotic applications models an object as a colored blob. During acquisition, users must manually select a region of interest in the image. This process can be repeated on the same object under different lighting conditions and the algorithm will assign a range of values to represent the region. Schlegel *et al.* [9] automates this process by requiring users to stand directly in front of the robot's camera for an 'introduction'. The system then generates a two-dimensional color histogram (in red-green chrominance space) describing a rectangular region on the user's chest. Though this approach offers some convenience, it requires users to stand in a specific location during training and the resulting model can only be used to track colors corresponding to the user's shirt.

3.2 Overview of Approach

Our proposed method improves the convenience and speed of previous techniques, and is particularly suitable for robotic applications. Unlike [9], the training stage is fully automated, by using a background modeling technique to segment the person from the background. This allows the subject to move freely during the training stage. We also

propose improvements to Schlegel's color modeling approach [9]. Instead of modeling a single region of color within the object, we identify and model multiple color-regions. Each region is modeled in three-dimensional RGB color-space, and is represented as a mixture of Gaussians. For increased robustness, our model also incorporates shape information without sacrificing speed.

The following sections describe how models are generated during training, how these models are used to locate and track users and their postures, and presents some implementation optimizations that allow for real time operation, even while searching for multiple models.

3.3 Training

Each posture from each person is separately trained during an off-line acquisition process, requiring about 10-20 seconds per posture. Training is done on a stationary robot and under lighting conditions that are similar to those to be encountered in subsequent operation.

Before the object can be modeled, it must be segmented from the image. Segmentation is accomplished using an adaptive background modeling technique similar to the one described in [7]. Each pixel of the background is modeled as a Gaussian distribution in the RGB color space, with the mean (μ_r,μ_g,μ_b) and standard deviation (σ_r,σ_g,σ_b). For each new pixel (x_r,x_g,x_b), the Gaussians are updated using a learning rate α, as follows:

$$\mu_i = \alpha x_i + (1-\alpha)\mu_i \tag{1}$$

$$\sigma_i^2 = max(\sigma_{min}^2, \alpha(x_i-\mu_i)^2 + (1-\alpha)\sigma_i^2) \tag{2}$$

where i = *r, g, b*.

The σ_{min}^2 term is introduced to prevent the variance from decreasing below a minimum value when the background remains constant for a long period.

Object segmentation is accomplished by comparing new pixels to the background model. A pixel is labeled as part of the foreground object if, for any value:

$$(x_i-\mu_i)^2 > (2\sigma_i)^2 \tag{3}$$

where i = *r, g, b*.

Segmented pixels are grouped together as connected-components to form a blob corresponding to the target object. Examples of a segmented image are shown in Fig. 1 (second row). As described in the next sub-sections, the segmented objects from each frame are combined to produce shape and color models.

3.3.1 Modeling Shape

Despite the fact that a human silhouette can be highly variable, there is enough regularity to warrant the inclusion of a shape-based model. The segmented foreground region from each frame is first normalized in terms of position and height, then quantized into a $32 \times w$ array of square blocks, where w is a function of the object's height to width ratio.

A map is then generated that contains (for each block) the likelihood of that block being a part of the foreground. Given N training frames, the probability at each block i is:

$$p_{shape}(i) = \frac{1}{N}\sum_{k=1}^{N} fg_k(i) \tag{4}$$

where $fg_k(i)$ is 1 if block i belongs to the foreground in frame k, and 0 if it belongs to the background.

High values in this map thus correspond to regions that are likely to be foreground and low values correspond to likely background regions. An illustration is provided in Fig. 1 (third row), where bright red regions correspond to foreground, blue regions correspond to background and black regions do not strongly correspond to either region.

3.3.2 Modeling Color

A common characteristic of human figures is that color remains relatively constant in the horizontal direction, while demonstrating more variability vertically. Variability usually occurs at the transitions between the hair and face, face and shirt, shirt and pants, and pants and shoes. It should also be noted that the relative size and location of these regions remain reasonably consistent even as a human moves.

To exploit the natural grouping of colors, our approach divides a target object into a vertical stack of horizontal color bands. We use 32 bands to represent our models. This number was selected because it provides sufficient resolution for representing the object, and because it allows for certain optimizations on a 32-bit system. Each band is modeled separately using a mixture of Gaussians.

During training, the foreground region is normalized in terms of height and quantized into the 32 bands. The pixel-values corresponding to each band are then accumulated into a histogram in RGB color space. At the end of the training period, the histogram is modeled as a mixture of Gaussians.

To produce a more robust representation of the object, adjacent bands are merged if their color distributions are sufficiently similar. Grouping begins with the most similar bands and continues with progressively less similar bands until the model is reduced to between one and six regions. Fig. 1 (forth row) shows the color distribution of each object. Colors are arranged so that the most dominant values are shown on the left and the least dominant on the right. The red lines delineate the regions after merging.

The resulting model contains information about the color composition, vertical location, and size of the prominent regions of color within an object and can be used for the detection and tracking of the object in a video sequence.

3.4 Detection and Tracking

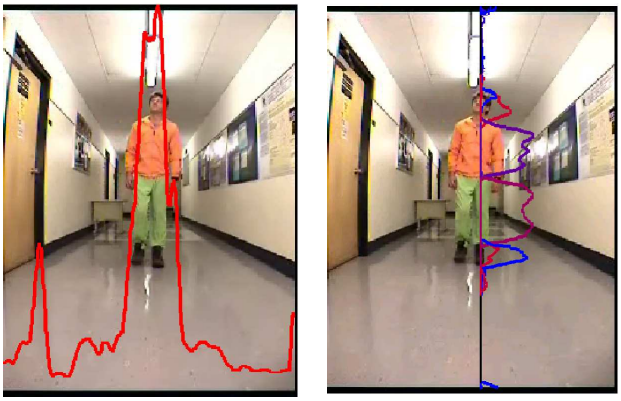
Since humans tend to assume an upright posture, they will usually occupy a larger proportion of an image in the vertical direction than they will in the horizontal direction. This property simplifies an object search because it allows

promising x-axis locations to be identified before considering y-axis locations.

Every pixel in the image is assigned a probability, which represents the likelihood that that pixel color is present in the foreground object. Pixels with color values matching the most prominent colors in the target model are assigned high probabilities, while colors not found in the model are assigned a probability of zero. For a given model, the probability that pixel i with color (x_r,x_g,x_b) belongs to the model is determined using all Gaussians in all bands of the color model:

$$p_{color}(i) = \frac{1}{N_{bands}} \sum_{bands} \frac{e^{-\left(\frac{(x_r-\mu_r)^2}{2\sigma_r^2}+\frac{(x_g-\mu_g)^2}{2\sigma_g^2}+\frac{(x_b-\mu_b)^2}{2\sigma_b^2}\right)}}{\left(\sqrt{2\pi}\sigma_r\right)\left(\sqrt{2\pi}\sigma_g\right)\left(\sqrt{2\pi}\sigma_b\right)} \quad (5)$$

The resulting probability values are then summed for every column of pixels to form a probability distribution with respect to the x-axis. The most prominent local maxima in this distribution are identified as promising x-axis locations. An example of such a probability mapping is shown in Fig. 2(a).



a) X-axis candidates b) Region probabilities

Figure 2. Person detection and tracking.

For each x-axis candidate, a 16-pixel wide column is defined, centered at the x-location. Pixels in every row of the column are assigned probability values, which represent the likelihood that the pixel's color is present in the corresponding object-color-region. Probabilities are determined as in Eq. 5, but with the sum computed over bands in that region. Probabilities are summed for every row, to find the y-axis distribution. Fig. 2(b) shows a model, which has been divided into four prominent regions. Probability mappings for each region are superimposed on the image.

As seen in Fig. 2(b), the probability maps tend to be high within the areas of their corresponding colors and the intersection between two probability mappings offer a good estimation of the border between adjacent colors. This effectively determines the location and vertical size of each region of color within an object.

In order to determine a final measure of similarity, the object-shape probability map is incorporated. As described previously, certain blocks of the shape-map will have a high probability of falling on the figure while other areas of the map (typically towards margins) will have a low probability. The shape-based probabilities are used to weight the color-based probabilities for each region, in order to produce a final similarity score.

3.5 Efficiency

It should be noted that although the model describing each object can comprise as many as six different color regions, each containing several Gaussian distributions, our implementation executes a one-time preprocessing step that compiles the Gaussians into a 3D array indexed by (R,G,B). With a single reference to this array, a probability measure can be obtained to determine the likelihood that that pixel is part of an object. If the probability of being present in an object was greater than zero, up to 6 additional arrays can be referenced to determine the probability that the pixel is contained in the object's sub-regions. These optimizations allow the tracking to be performed in real-time (20 frames/sec), even when 15 models are involved, and on a modest 1 GHz computer. Although the probability arrays require more memory than any other data structure, the color-space is sub-sampled to minimize the demands. The current implementation quantizes the color-space into $32\times32\times32 = 32{,}768$ different colors, and requires a total of 64-bits to store the region and sub-region probability values. This requires about 262 KB of memory per model.

4 Experimental Setup and Results

We validated our approach with a Pioneer 3DX mobile robot equipped with a SICK LMS-200 laser rangefinder, front and rear sonar, and a pan-tilt-zoom (PTZ) camera. For robot control we use the Player robot device interface [10]. To allow for safe maneuvering, the robot executes laser and sonar-based obstacle avoidance routines. Our validation consists of quantitative and qualitative evaluation for the *visual awareness* and the *robot control* modules.

4.1 Validation of Visual Awareness

Tracking and distance estimation. We tested the tracking component using an experiment where the robot was programmed to pursue a person, while maintaining a fixed distance. Computation of the person's distance from the robot was based on the camera calibration procedure described in [11] and on the assumption that the floor is flat and the person's feet are always on the floor.

For this trial, the robot accurately pursued a human-target, who alternated between forward and backward movement through a 100 meter-long hallway. Frames from the robot's camera are shown in Table 1, where the green rectangle and the green outline have been generated by the detection and tracking module. In order to assess the accuracy of distance estimation, the computed positions at

each displayed frame are shown together with those obtained from the robot's laser rangefinder (ground truth).

It is worth emphasizing that the experiment illustrates the ability to perform real-time tracking and distance estimation for a moving target while the robot (and its camera) are also moving.

100	200	300	400	500	600	700	800	900
								
7.3	8.3	7.3	7.3	6.4	5.8	5.8	5.8	7.0
7.7	8.2	7.3	7.3	6.8	5.6	6.1	6.3	6.6

Table 1. Tracking distance estimation. *Top Row*: Frame number. *2nd row*: Frame image. *3rd row:* Estimated distances (meters). *4th row*: Ground truth (laser) distances.

Posture recognition – qualitative validation. Fig. 3 shows the recognition of postures in the presence of multiple persons. The system was trained on postures from 3 users (which are correctly recognized), while the 2 unknown users are (correctly) ignored. This experiment shows that the approach can robustly detect and track multiple postures from multiple users, while ignoring irrelevant persons in a possibly crowded environment.

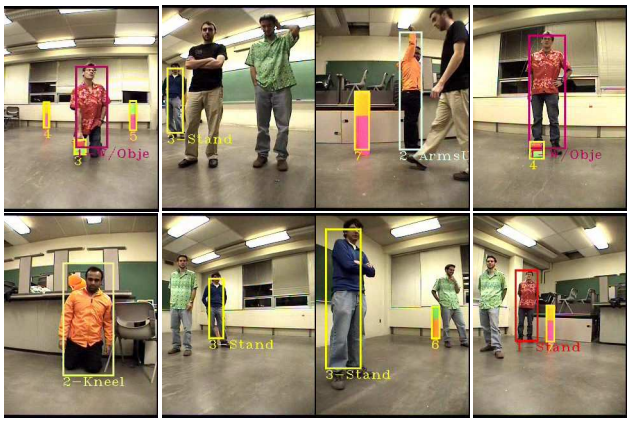


Figure 3. Posture recognition – qualitative validation. Modeled users: red, orange, blue shirt. Unknown users: black, green shirt.

Posture recognition – quantitative validation. To quantitatively estimate the recognition accuracy, we trained the system on 5 users with 3 postures each. Subjects were asked to display each posture for about 30 seconds, while they moved through the camera's field of view. Measurement of correct posture frequency was started 10 frames after each change in posture and was continued for 200 frames. Table 2 shows the percentage of correctly recognized postures.

	User1	**User2**	**User3**	**User4**	**User5**
Standing	92.5%	91%	99.5%	100%	100%
Kneeling	97%	98%	99%	100%	100%
Arms-up	95.5%	100%	100%	99%	100%

Table 2. Posture recognition – quantitative validation.

4.2 Validation of Robot Control

We performed the robot-control experiments in the classroom environment shown in Fig. 3, using two users. For each user, the robot was trained to detect standing, arms-up (AU), kneeling-without-object (Kn), and kneeling-with-object (Ob) postures.

The robot's tasks consist of a series of target reaching and object transport duties, representative for a service robot's potential delivery scenarios. Each task was associated with a user-posture and was given one of three priority levels. The task-posture-associations and priority-levels are shown in Table 3 (e.g. Task 'T5' was associated with Person *2*, Posture *Arms-Up*, and was assigned a *medium* priority). The standing posture is not associated with any task, but rather serves as a trigger from the *visual awareness* module that a user is in vicinity. In addition to the posture-associated tasks, the robot is also equipped with a wandering task (T0), which has a *low* priority, and is executed as long as the robot has no requests to service.

In our experiments, the two users requested services from the robot over an extended period, in order to demonstrate the main features of our approach: 1) awareness to the presence of multiple people, 2) ability to handle multiple requests, 3) ability to handle task interruptions and 4) long-term robot-autonomy.

Upon starting, the robot wanders through the room, waiting for a request from a user. If a posture is witnessed for more than a few seconds, the robot responds with an auditory confirmation signal, and either performs the associated task, or queues it, depending on the task-priority. To reduce false detection during posture-transitions, the robot briefly ignores a person if they just requested a new task. The robot also ignores requests for tasks that are in the queue or are currently being executed.

We performed experiments with the task scenario shown in Table 3, and we repeated each experiment four times. We chose to use the same sequence of requests in order to establish a baseline for evaluation, both from the perspective of task execution (the *control module*) and from the perspective of the posture recognition (the *visual awareness module*).

In Table 3, we show the sequence of tasks that were requested during the scenarios and, for validation, we indicate the correct action. The *Posture* row lists the task-requests that were issued. Each request requires that either the new task, or the current task be suspended and pushed to the queue (depending on priority). In the absence of new requests (indicated by a '–'), the current task is allowed to

Posture	–	1 Ob	2 AU	1 Kn	2 Kn	2 Ob	1 AU	–	–	–	–	–	–
Task (priority)	T0 (low)	T3 (med)	T5 (med)	T1 (med)	T4 (high)	T6 (high)	T2 (high)	–	–	–	–	–	–
Action	–	T0→Q	T5→Q	T1→Q	T3→Q	T6→Q	T2→Q	Q→T6	Q→T2	Q→T3	Q→T5	Q→T1	Q→T0
Current	T0	T3	T3	T3	T4	T4	T4	T6	T2	T3	T5	T1	T0
Queue	–	T0	T5,T0	T5,T1, T0	T3,T5, T1,T0	T6,T3, T5,T1, T0	T6,T2, T3,T5, T1,T0	T2,T3, T5,T1, T0	T3,T5, T1,T0	T5,T1, T0	T1,T0	T0	–

Table 3. Task requests for the task-execution scenario. *Postures row*: 1 & 2 indicate User 1 and User 2 respectively. *Tasks row*: the tasks associated with the posture above. *Actions row*: *'→'* indicates pushing task to, or popping task from the queue.

proceed to completion. The suspended task with the highest priority is then popped from the queue and resumed.

Results. On all trials, the robot correctly identified the postures (and thus the requests), took the correct decisions regarding which tasks to perform, and correctly executed every phase of each task. Each run took approximately 20 minutes. During these runs, the priority method used was to process tasks with highest priority first; for tasks of equal priority, a FIFO (first-in-first-out) method was used. For these experiments, we also recorded the robot's progress as it completed each phase of a task. This allowed us to demonstrate that if the robot is interrupted in the middle of a task, upon resuming its prior-execution, the robot will continue from the point of interruption, instead of performing it from the start.

5 Conclusion

In this paper, we proposed a framework for developing robot assistants that addresses two key issues of human-robot interaction: *awareness* of the environment and other agents, and *long-term interaction* with multiple users. Our awareness mechanism is built on visual capabilities that allow the robot to identify multiple users, with multiple postures, in real-time, in dynamic environments in which both the robot and human users are moving. Long-term human-robot interaction is supported by a novel control architecture, which enables the representation of complex, sequential and hierarchical robot tasks. The architecture provides the robot with flexibility in dealing with multiple users, such as to accommodate multiple user requests and task interruptions, over extended periods. We validated our approach on a Pioneer 3DX mobile robot, performing service tasks in a real-world environment. Our experimental results demonstrate the robot's ability to engage in interactions in a natural way, to deal with multiple users, and to be constantly aware of their surroundings, thus advancing service robotics toward deployment of robots into the real world.

Acknowledgements. Supported by the National Science Foundation CAREER Award IIS-0546876 and by the Office of Naval Research Award N00014-06-1-0611.

References

[1] Terrence Fong, Illah Nourbakhsh, and Kerstin Dautenhahn. A survey of socially interactive robots. *Robotics and Autonomous Systems*, 42:143–166, 2003.

[2] G. Mori and J. Malik. Estimating human body configurations using shape context matching. In *ECCV02*, page III: 666, 2002.

[3] A. Agarwal and B. Triggs. Recovering 3d human pose from monocular images. *PAMI*, 28(1):44–58, 2006.

[4] C. Stauffer and W. Grimson. Learning patterns of activity using real-time tracking. *IEEE Transactions on PAMI*, pages 747–757, 2000.

[5] A. Elgammal, R. Duraiswami, D. Harwood, and L. Davis. Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. *Proceedings of the IEEE*, 90:1151–1163, 2002.

[6] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: Principles and practice of background maintenance. In *Intl. Conf. on Computer Vision*, pages 255–261, 1999.

[7] M. Nicolescu, G. Medioni, and M.-S. Lee. Segmentation, tracking and interpretation using panoramic video. In *IEEE Workshop on Omnidirectional Vision*, pages 169–174, 2000.

[8] J. Kang, I. Cohen, and G. Medioni. Continuous tracking within and across camera streams. In *Computer Vision and Pattern Recognition*, pages 267–272, 2003.

[9] C. Schlegel, J. Illmann, H. Jaberg, M. Schuster, and R. Worz. Vision-based person tracking with a mobile robot. In *British Machine Vision Conference*, pages 418–427, 1998.

[10] Brian Gerkey, Richard T. Vaughan, and Andrew Howard. The player/stage project: Tools for multi-robot and distributed sensor systems. In *Proc., the 11th International Conference on Advanced Robotics*, pages 317–323, 2003.

[11] Z. Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *Intl. Conf. on Computer Vision*, pages 666–673, 1999.