ヒューマンインタフェース 88－2
（2000．5．12）

列に並ぶロボットの開発
— 人とロボットの社会的インタラクションに向けて —

中内 靖 *　　Reid Simmons†
* 防衛大学校 情報工学科
†School of Computer Science, Carnegie Mellon University

人と共存するサービスロボットでは、人の社会的行動規範を理解する必要がある。本研究では、その一例として、列に並ぶロボットを開発した。パーソナルスペースの概念に基づいた列の認識ならびに行動戦略等について報告する。

A Social Robot that Stands in Line
— Towards Social Interaction between Human and Robot —

Yasushi Nakauchi* and Reid Simmons†
*Dept. of Computer Science, National Defense Academy
†School of Computer Science, Carnegie Mellon University

Recent research results on mobile robot navigation systems make it promising to utilize them in service fields. But in order to utilize the robot in a peopled environment, it should recognize and respond to people's social behaviors. In this paper, we describe a social robot that stands in line as people do. Our system uses the concept of personal space for modeling a line of people and we have experimentally measured the actual size of the personal space when people form lines.

1 Introduction

Recent research results on mobile robot navigation systems make it promising to utilize them in service fields [6]. In general, the environments where service robots perform tasks are shared with humans. Thus, the robots have to interact with humans, whether they like it, or not.

Humans also interact with each other. Sometimes, the interactions lead to resource conflicts. In order to maintain order, humans use social rules. For example, at bank counters or grocery stores, humans stand in line and wait for their turn. If a person does not understand or obey the social rules, he/she will not be able to get the services.

This is also true for service robots. If a service robot is asked to purchase merchandise and it does not know the social rules of humans, it may treat the humans who are standing in line as obstacles and will not be able to achieve its task. Therefore, it is also required for service robots to understand the social behaviors of humans, and to obey the rules.

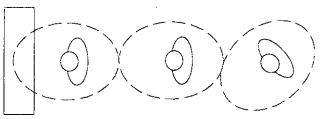
So far, meal deliver robots in hospital, tour guide robots in museum and secretary robots in office have been developed and are actually used in real environments. But these robots can not interact with people by taking social behaviors of human, as mentioned above, into consideration.

There are many aspects to the social rules and behaviors in human society. Among them, standing in line is one of the most highly socialized and crucial skill required for robots which execute tasks in peopled environments. Therefore, as a first step towards the social behavioral robot, in this research, we have developed an autonomous robot system that can stand in line with other people.

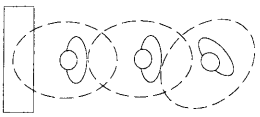
2 Modeling a Line of People

In order to realize a robot that stands in line, the robot should be able to recognize a line of people and also should be able to stand in line as people do. To recognize a line of people, we must first under-

stand the social activities of humans and how they form lines. Then, we can realize a robot that recognizes the social behavior of human and performs actions as people do.



(a) Farthest case.



(b) Closest case.

Figure 1: A line of people modeled by a chain of personal spaces.

The notion of "human territoriality" or "personal space" has been studied in the research field of cognitive science [4]. Personal space, which is a person's own territory, can be modeled as oval in shape, wider towards the front of a person. A person feels uncomfortable when other people are in his/her personal space.

When people form a line, in general, they keep a certain distance from other people. They also usually stand so that they face towards the person in front. These phenomena can be described based on the concept of personal space. The idea is that the person who is standing in line keeps enough distance to the person in front so that he/she does not feel uncomfortable. On the other hand, he/she stands close enough to the person in front to avoid other person cutting in the line. Thus, we employed this notion and modeled a line of people as a chain of personal spaces as shown in figure 1.

It is reported that the size of personal space varies depending cultural norms and on the task being performed (e.g., standing in line, sitting on a bench, chatting together) [4]. Since we know of no work that explicitly addresses the size of personal space for the task of standing in line, we performed some experiments to estimate the actual size of personal space when people stand in line. In particular, we asked subjects to start $200cm$ away from the person at the end of the line and to move forward until they felt uncomfortable. Since personal space varies by body orientation, we collected data for eight different orientations of both the subject and the person at the end of the line.

In general, there is some distance we feel that we actually are standing in line. We assumed that the closest distance is observed when we stand as close as possible so that other person feels uncomfortable. In this case, the personal spaces of two people overlap, but a person may not be in the other person's personal space as shown in figure 1 (a). On the other hand, we assumed the farthest distance is observed when we stand as far as possible so that other person may not cut in a line. In this case, personal spaces of two persons are connected but not overlapped as shown in figure 1 (b). So in the experiment, we asked subjects to move in line in two ways. One is "to move in line as uncomfortable as possible," to estimate the closest range. The other is "to move in line as far as possible so that other person may not cut in a line," for the farthest case. We also asked subjects not to make eye contact or talk to the person in front, to eliminate the influence of affinity.

We performed these experiments with 10 subjects. When calculating the size of the personal space, we assumed the size of personal space of the person in front is identical to the personal space of the subject. This is because a subject has no way of knowing the size of the other person's personal space. Therefore, when the same body direction of two persons are facing, the personal space towards that direction is the half of the distance between two persons. The average size of personal space, as estimated from these experiments, is illustrated in figure 2. As shown in figure, the experimentally derived personal space is oval, as has been reported in the cognitive literature.

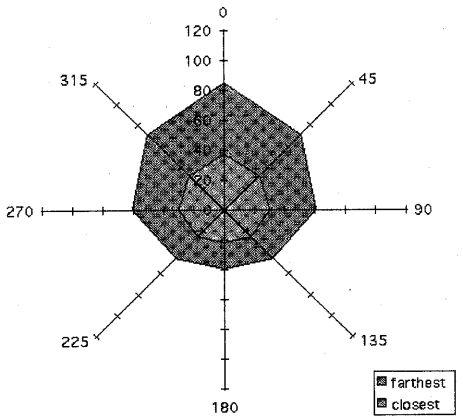


Figure 2: Actual size of personal space when people stand in line.

3 Robot That Stands in Line

In this section, we describe how we detect people using stereo vision, how we use that information to estimate which people are in line, and how the robot decides how to move to get into line.

3.1 Finding a Line of People

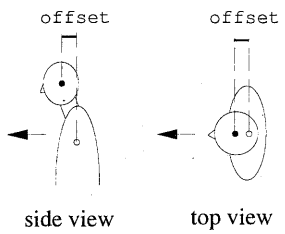


Figure 3: The body model of human.

To recognize a line of people, the robot detects each person's position and orientation. Several face detection systems have been developed to detect people using vision [1]. Unfortunately, most current face detection methods work only for frontal views. Since the robot typically observes the people from the side or from the back, these face detection methods cannot be used for our purpose. Oren et al. have developed a system to detect human bodies in an image [5]. This system only detects segments of people in an image and it cannot determine the positions and orientations of the people. Thus, we needed to develop a novel method for detecting people that is useful for our application.

In general, three dimensional information is required to determine the position and orientation of people. Typically, stereo vision systems and laser range finders are used for that purpose. In this research, we use stereo vision system to detect people for two main reasons. First, stereo is cheap in cost compared with laser range finders. Second, there are readily available packages that can do real-time stereo calculations on images of modest size [3].

We model the shape of human body as shown in figure 3. We assume that a person has an oval shape around the body, is round around the head, and that the neck is the narrowest part of the upper body. We also assume that the head juts over the body in front. Based on these assumptions, we have developed a person-detection algorithm using stereo vision.

(a-1) Left raw image. (a-2) Right raw image.

(b) Disparity image.

Figure 4: Raw images and the derived disparity image.

The first step of the algorithm captures left and right camera images using two CCD cameras and

then calculates a disparity image from the two images using triangulation method (see figure 4). The problem is made more difficult by the appearance of multiple figures in the image, as well as other objects in the distance (such as furniture or walls). We use a clustering method to collect disparity data for each object in the image. Data points that are within $5cm$ of each other are connected, and all connected data points are categorized as a single cluster. We then discard clusters that have fewer than a threshold number of data points, since these typically represent noise or objects other than humans.

We then determine the position and orientation of each person from the clustered disparities. For each set of clustered disparities, we find the nose and the chest height position data by using the characteristic body shape. First, we find the highest data point in a cluster and assume it is the top of the head. This is reliable since there are no data above the head, in general. Then, we find the narrowest point in width below the head, and assume it is the neck position. Finally we estimate the nose height as the middle point between the head top and the neck, and the chest height as $20cm$ below the neck. At the same time, to eliminate the objects otherthan humans, we discarded the cluster whose heighest point is lower than a certain threshold.

We then find the ellipse that best fits the stereo data at the chest height, and find the circle that best fits the stereo data at nose height. The mathematical expression of an ellipse that has arbitrary size, location and orientation is as follows:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a\cos\theta_0\cos\theta + b\sin\theta_0\sin\theta + x_0 \\ -a\sin\theta_0\cos\theta + b\cos\theta_0\sin\theta + y_0 \end{bmatrix} \quad (1)$$

where a and b denote the length of major and minor axes, respectively, x_0 and y_0 denote the offset of the center from the origin, and θ_0 denotes the inclination of the ellipse. To simplify things, we use constant values for a and b for all people, assuming that human are approximately the same size. Thus, the problem becomes finding the variables x_0, y_0 and θ_0 that best fit the observed data points.

Least-square fitting algorithms are the efficient methods to calculate the best fitting ellipse from data points [2]. However, these methods do not restrict the size of ellipses they consider, so they can fit the data to huge ellipses if the data points are mainly linear. Also, even if we modify the algorithm to restrict the ellipse size, due to noise it may decide that the best fitting ellipse lies in *front* of the data points. This is not physically possible, since the observed data points are on the surface of a human body. Thus, we need to constrain the body (ellipse) to fall "behind" the data points, relative to the gaze of the camera.

The Hough transform is also an efficient and intuitive method to calculate geometric features from data points. Since it has fewer limitations than the least-squares method, we use the Hough transform with some modifications to make it applicable for people detection.

Since there are three variables to estimate, we need to have a three dimensional Hough transform space. To do this, we discretize orientation (θ_0) in 15 degrees increments and use 24 separate x_0-y_0 planes, each corresponding to a different θ_0. The x_0-y_0 planes are represented as grids, discretized at $1cm$ resolution. The basic idea behind our Hough-transform based algorithm is to draw ellipses centered around each data point. The ellipses are drawn at different orientations for each x_0-y_0 plane. For example, Hough images for $\theta_0 = 0$ and $\theta_0 = 60$ are the x_0-y_0 planes as shown in figure 5-(b-1) and (b-2), respectively. In each plane, the grid point that has the most number of ellipse intersections is considered to be the most likely center of the ellipse. The plane that has the highest ranking center point determines the most likely orientation.

In the example shown in figure 5, the data points came from viewing a person whose orientation is $\theta_0 = 0$ (for clarity, in this example we are using data points with no noise). Thus, in the projected x_0-y_0 plane where $\theta_0 = 0$ (figure 5-b-1), the ellipses all intersect at the position where the person is actually located. But in the x_0-y_0 plane where $\theta_0 = 60$ (figure 5-b-2), the ellipses do not intersect very much. Thus, $\theta_0 = 0$ is determined to be the best orientation.

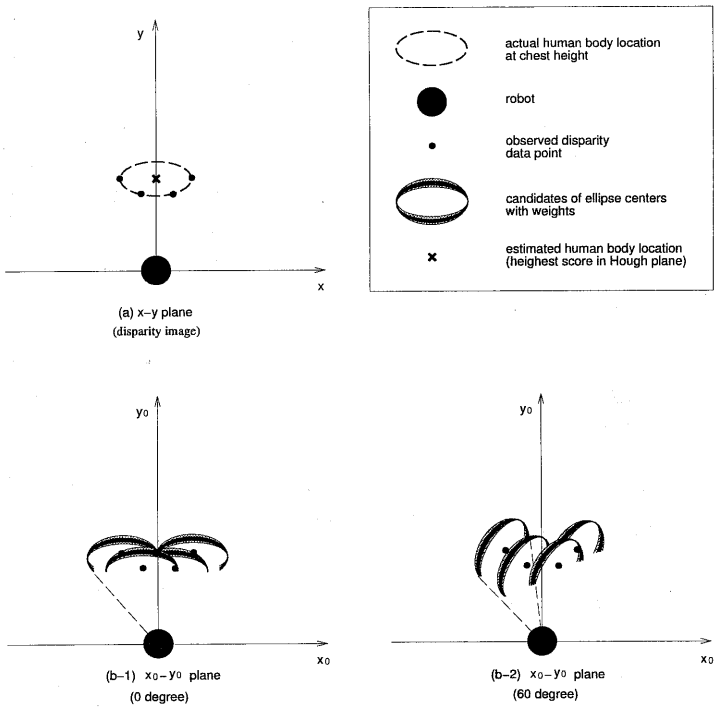


Figure 5: Hough transform used for ellipse fitting.

In addition to the above procedures, we have modified original Hough transform as follows. Since noise may be contained in disparity data, if the ellipses are drawn with a thin line, the number of intersections may be small or, in some cases, the wrong position may become dominant. We reduce that possibility by widening the rims of the ellipses (figure 5-b). At the same time, we weighted the widen rims so that the middle of ellipse becomes higher score and the edges become lower score.

Also, in order to increase reliability, we assume that the candidate ellipses should be behind the data points. To do so, we use only the portion of the ellipse which can observed by the camera. In other words, the ellipse is cut at the tangent lines from the origin (camera position), denoted by dotted lines in figure 5-(b-2).

The algorithm to find a circle that corresponds to the head position is the same as for ellipse fitting. However, since the circle does not have orientation, we need only one x_0-y_0 plane.

Finally, we determine the body direction using the best fits for the ellipse and the circle. The result of ellipse fitting on the chest height data determines the body orientation. However, it is ambiguous whether the person is facing forward or backward. So, based on the assumption that the head juts out in front of the body, we resolve the ambiguity by calculating the jutting direction of the head. An example of recognized humans with their locations and orientations are as shown in figure 6. Section 5 presents experimental results on the accuracy and reliability of this procedure.

3.2 Entering and Waiting in Line

To enter a line, the robot needs to know where to expect the head of the line (the *point of service*) and the (approximate) direction in which the line

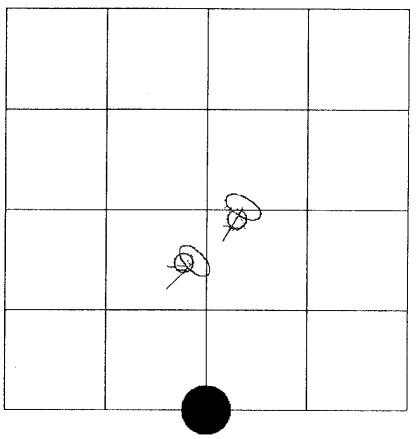


Figure 6: The detected people by the stereo vision system. (black circle: robot, ellipse: chest, circle: head, arrow: body direction, grid size: $1m$)

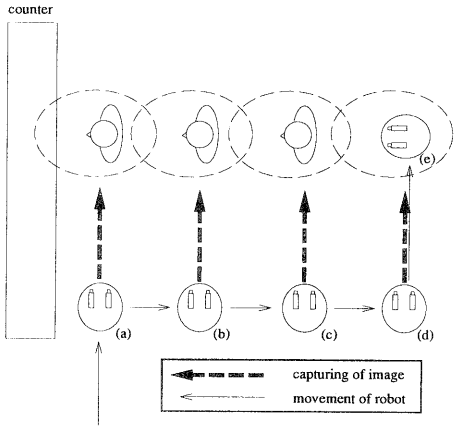


Figure 7: The movement of the robot to enter a line.

will form. The robot navigates (see Section 4) to a point $150cm$ from the point of service and faces perpendicular to the expected line direction (Figure 7-a). If it detects nothing at the point of service, it moves forward, rotates 90 degrees, and assumes that it is at the head of the line.

If a person is detected, on the other hand, the system turns and moves down the line. The distance moved is based on the position of the last person detected so far, their orientation, and the estimated size of their personal space. For this calculation, we use the average size of the closest and the farthest personal space. Thus, the robot will travel further if the person is standing backwards than if he/she is facing forwards (towards the point of service).

After moving, the robot tries again to detect people. If there is no one detected, or if the distance between the last detected person and the current detected person is greater than the farthest personal distance, then the robot moves into line. Otherwise, it moves further from the counter until it eventually finds the place to stand and joins the line (e.g., the movements from positions (b) to (e) in Figure 7).

Once the robot joins the line, it uses a combination of sonar and laser sensors to track the position of the person in front of it. The robot moves up in line to keep its personal space connected to the presumed personal space of the person in front. It will also back up if that person backs up enough to impinge on the robot's own personal space. The robot also recognizes when it has reached the head of the line. For this, we assume that the point of service is a counter and we use data from a laser range finder, set at the height of human legs. If the observed object surface is smooth enough, it is recognized as the counter. If not, it is recognized as human.

We have extended the above procedure to handle curved lines, as well. The basic idea is to keep an ordered list of all the detected people. Each time the robot detects a new person, it adds that person to the list and computes a vector between the positions of the last two people on the list. This vector is the robot's estimate of the local direction of the line. The robot then turns so that it is heading parallel to that vector and moves a sufficient distance so that it will be across from the expected position of the next person in line (under the assumption that the line continues in the given direction). Once the robot is in line, it moves in the direction of the next person on the list, so that it follows the curve of the line as it moves forward. This algorithm has been

show to work reliably, as long as the line does not bend too sharply.

As a practical application that requires the robot to stand in line, we developed a program that enable the robot to purchase a cup of coffee from a kiosk in the foyer of our building. First, the robot gets an order from a user, asking it to purchase a cup of coffee. Currently, this is done via keyboard input, but we are working on having voice-activated commands. The robot then navigates to the coffee shop. Here, we assume that the robot has an environmental map so that it can navigate to the coffee kiosk and precisely localize in front of the counter. The robot then maneuvers to get into line, as described earlier in this section, and moves up in line until it detects that it is at the head of the line (the point of service).

At the counter, the robot uses synthesized speech to place an order and waits for the attendant to put the coffee in a cup holder attached to the front of the robot. To determine when it has received the coffee cup, the robot tilts its camera so that it can view the cup holder and waits until the camera image changes significantly. After politely thanking the attendant (and asking for the coffee to be put on its tab), the robot navigates back to the place where it got the order and announces that it has arrived with the (hopefully still hot) cup of coffee.

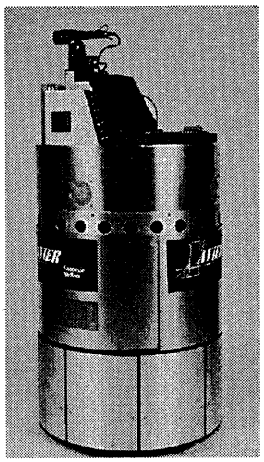


Figure 8: Mobile robot Xavier.

4 Implementation

We have implemented these algorithm on the autonomous mobile robot Xavier [6]. Xavier is built on top of a 24 inch diameter Real World Interface base (see figure 8). The base is a four-wheeled synchro-drive mechanism that allows for independent control of the translational and rotational velocities. The sensors of Xavier include bump panels, wheel encoders, a 24 element sonar ring, a Nomadics front-pointing laser light striper with a 30 degree field of view, and two Sony monochrome cameras on a Directed Perception pan-tilt head. Xavier also has a speaker and a text-to-speech card. Control, perception, and planning are carried out on two 200 MHz Pentium computers, running Linux.

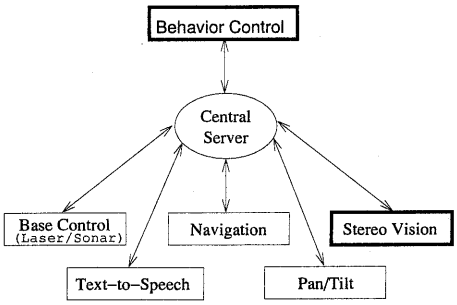


Figure 9: The Software Architecture of Xavier.

The software architecture of Xavier is based on the Task Control Architecture. TCA facilitates synchronization amongst a collection of processes and provides facilities for interprocess communication among the processes (TCA) [6]. The following modules were previously developed and were used in our research without modification (see figure 9).

Base Control Module This module controls the base wheels, providing translation and rotation of the robot, and does obstacle avoidance. It also manages the laser and sonar sensors, and reports the current readings to other modules.

Navigation Module This module maintains the environmental map. It uses decision-theoretic path planning and a probabilistic navigation scheme to reliably navigate the robot to a specified goal position [6].

Text-to-Speech Module This module synthesizes and outputs arbitrary natural language text from the speaker.

Pan/Tilt Module This module pans and tilts the cameras.

For this project, we developed several additional modules that perform stereo vision and that control the overall "stand in line" task. The stereo vision module utilizes two monochrome CCD cameras and recognizes humans based on the algorithm described in section 3.1. For the calculation of disparity image, we used the SVS[1] stereo engine [3]. The cycle time for people detection was about $800msec$. The behavior control module controls the whole sequence to purchase coffee as described in section 3.2.

5 Experimental Results

To measure the performance and reliability of the system, we performed a number of experiments with the robot in various situations. First, we conducted experiments to determine the quality of the person detection algorithm and confirmed its accuracy.

Then, to confirm how the robot performs in the overall task, we ordered the robot to purchase coffee at the real coffee shop located in the foyer on the same floor of the building as our laboratory. The robot was able to detect the line correctly, enter it, and move up to the head of the line in 70% out of 20 trials.

Many of the trials were done during normal operating hours of the coffee shop, with the experimenters unseen. While many of the customers had seen Xavier previously, they had only encountered it navigating through the corridors. Nevertheless, several customers got in line behind Xavier and waited patiently in line together with the robot. At the least, this provides anecdotal evidence that Xavier is considered, in some sense, to be a social member of human society.

[1] SVS is a trade mark of SRI International.

6 Conclusion

This paper presented a social robot that can stand in line as people do. It presents a model of lines of people using the notion of personal space, and describes how we obtained an estimate of the size of personal space by experiments. We then described an algorithm that uses this model, together with a human detection algorithm based on stereo vision, to enter and wait in line. Finally, the paper presents an implementation of the algorithms on the mobile robot Xavier, and experimentally demonstrates its performance.

References

[1] R. Brunelli and T. Poggio, "Template Matching: Matched Spatial Filters and Beyond," M.I.T., Cambridge, MA, *A.I. Memo*, No.1536, 1995.

[2] A.W. Fitzgibbon, M. Pilu and R.B. Fisher, "Direct Least Square Fitting of Ellipses," Dept. of Artif. Intell., The University of Edinburgh, Scotland, *DAI Research Paper*, No.794, 1996.

[3] K. Konolige, "Small Vision Systems: Hardware and Implementation," *In Proc. Eighth International Symposium on Robotics Research*, 1997.

[4] M. Malmberg, *Human Territoriality: Survey of Behavioural Territories in Man with Preliminary Analysis and Discussion of Meaning*, Mouton Publishers, 1980.

[5] M. Oren, C. Papageorgiou, P. Shinha, E. Osuna and T. Poggio, "A Trainable System for People Detection," *In Proc. of Image Understanding Workshop*, pp.207–214, 1997.

[6] R. Simmons, R. Goodwin, K. Zita Haigh, S. Koenig and J. O'Sullivan, "A Layered Architecture for Office Delivery Robots," *In Proc. of Autonomous Agents*, pp.245–252, 1997.