Useful Computer Vision Techniques for Human-Robot Interaction

O. Deniz, A. Falcon, J. Mendez, M. Castrillon

Universidad de Las Palmas de Gran Canaria
Departamento de Informática y Sistemas
Edificio de Informática y Matemáticas. Campus de Tafira, 35017, Las Palmas, Spain
{odeniz,afalcon,jmendez,mcastrillon}@dis.ulpgc.es

Abstract. This paper describes some simple but useful computer vision techniques for human-robot interaction. First, an omnidirectional camera setting is described that can detect people in the surroundings of the robot, giving their angular positions and a rough estimate of the distance. The device can be easily built with inexpensive components. Second, we comment on a color-based face detection technique that can alleviate skin-color false positives. Third, a simple head nod and shake detector is described, suitable for detecting affirmative/negative, approval/dissaproval, understanding/disbelief head gestures.

1 Introduction

In the last years there has been a surge in interest in a topic called social robotics. As used here, social robotics does not relate to groups of robots that try to complete tasks together. For a group of robots, communication is simple, they can use whatever complex binary protocol to "socialize" with their partners. For us, the adjective social refers to humans. In principle, the implications of this are much wider than the case of groups of robots. Socializing with humans is definitely much harder, not least because robots and humans do not share a common language nor perceive the world (and hence each other) in the same way. Many researchers working on this topic use other names like human-robot interaction or perceptual user interfaces. However, as pointed out in [1] we have to distinguish between conventional human-robot interaction (such as that used in teleoperation scenarios or in friendly user interfaces) and socially interactive robots. In these, the common underlying assumption is that humans prefer to interact with robots in the same way that they interact with other people.

Human-robot interaction crucially depends on the perceptual abilities of the robot. Ideal interaction sessions would make use of non-invasive perception techniques, like hands-free voice recognition or computer vision. Hands-free voice recognition is a topic that is still under research, being the most attractive approaches the combination of audio and video information [2] and microphone arrays [3].

Computer vision is no doubt the most useful modality. Its non-invasiveness is the most important advantage. In this paper, three computer vision techniques for human-robot interaction are described. All of them have been used in a prototype social robot [4]. The robot is an animal-like head that stands on a table and has the goal of interacting with people.

2 Omnidirectional Vision

Most of social robots built use two types of cameras: a wide field of view camera (around 70 deg), and a foveal camera. The omnidirectional camera shown in Figure 1 gives the robot a 180 deg field of view, which is similar to that of humans. The camera is to be placed in front of the robot. The device is made up of a low-cost USB webcam, construction parts and a curved metallic surface looking upwards, in this case a kitchen ladle.

Fig. 1. Omnidirectional camera.

As for the software, the first step is to discard part of the image, as we want to watch only the frontal zone, covering 180 degrees from side to side. Thus, the input image is masked in order to use only the upper half of an ellipse, which is the shape of the mirror as seen from the position of the camera.

A background model is obtained as the mean value of a number of frames taken when no person is present in the room. After that, the subtracted input images are thresholded and the close operator is applied. From the obtained image, connected components are localized and their area is estimated. Also, for each connected component, the Euclidean distance from the nearest point of the component to the center of the ellipse is estimated, as well as the angle of the center of mass of the component with respect to the center of the ellipse and its largest axis. Note that, as we are using an ellipse instead of a circle, the nearness measure obtained (the Euclidean distance) is not constant for a fixed real range to the camera, though it works well as an approximation, see Figure 2.

The background model M is updated with each input frame:

$$M(k+1) = M(k) + U(k) \cdot [I(k) - M(k)] \tag{1}$$

, where I is the input frame and U is the updating function:

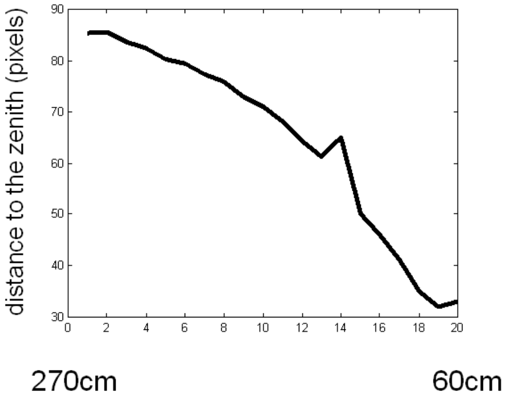

Fig. 2. Approximate distance measure taken with the omnidirectional camera as a person gets closer to the robot.

$$U(k) = \exp(-\beta \cdot D(k)) \tag{2}$$

$$D(k) = \alpha \cdot D(k-1) + (1-\alpha) \cdot \alpha |I(k) - I(k-1)| \tag{3}$$

α (between 0 and 1) and β control the adaptation rate. Note that M, U and D are images, the x and y variables have been omitted for simplicity. For large values of α and β the model adaptation is slow. In that case, new background objects take longer to enter the model. For small values of α and β, adaptation is faster, which can make animated objects enter the model.

The method described up to this point still has a drawback. Inanimate objects should be considered background as soon as possible. However, as we are working at a pixel level, if we set the alfa and beta parameters too low we run the risk of considering static parts of animate objects as background too. This problem can be alleviated by processing the image D. For each foreground blob, its values in D are examined. The maximum value is found, and all the blob values in D are set to that level. Let the foreground blobs at time step k be represented as:

$$B_i = \{x_{ij}, y_{ij}\} \ \ ; \ i = 1,..,NB \ ; \ j = 1,..,N_i \tag{4}$$

There are NB blobs, each one with N_i pixels. Then, after (3) the following is applied:

$$m_i = \max_{j=1,..,N_i} D(x_{ij}, y_{ij}, k) \ \ ; \ i = 1,..,NB \tag{5}$$

$$D(x_{ij}, y_{ij}, k) = m_i \ ; \ i = 1,..,NB \ ; \ j = 1,..,N_i \tag{6}$$

With this procedure the blob only enters the background model when all its pixels remain static. The blob does not enter the background model if at least one of its pixels has been changing.

3 Face Detection

Omnidirectional vision allows the robot to detect people in the scene, just to make the neck turn towards them (or somehow focus its attention). When the neck turns, there is no guarantee that omnidirectional vision has detected a person, it can be a coat stand, a wheelchair, etc. A face detection module should be used to detect people (and possibly facial features). Facial detection commonly uses skin-color as the most important feature. Color can be used to detect skin zones, though there is always the problem that some objects like furniture appear as skin, producing many false positives. Figure 3 shows how this problem affects detection in the ENCARA facial detector [5], which (besides other additional cues) uses normalized red and green color components for skin detection.

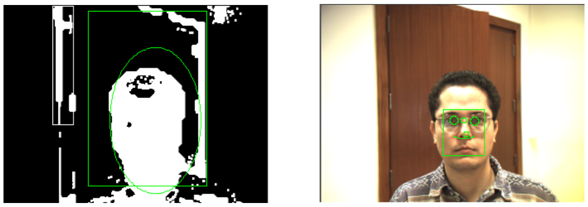

Fig. 3. Skin color detection. Note that wooden furniture is a distractor for facial detection. Both the bounding box and the best-fit ellipse are rather innacurate (left).

In order to alleviate this problem, stereo information is very useful to discard objects that are far from the robot, i.e. in the background. Stereo cameras are nowadays becoming cheaper and faster. A depth map is computed from the pair of images taken by the stereo camera. For some cameras, the depth map is efficiently computed with an included optimized algorithm and library. The map is thresholded and an AND operation is performed between this map and the image that the facial detector uses. Fusion of color and depth was also used in [6,7,8]. The results are shown in Figure 4. Note that most of the undesired wood colored zones are filtered out.

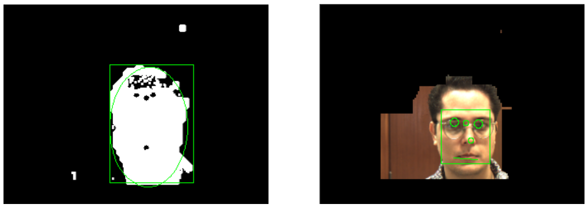

Fig. 4. Skin color detection using depth information.

4 Head Nod/Shake Detection

Due to the fact that practical (hands-free) voice recognition is very difficult to achieve for a robot, we decided to turn our attention to simpler (though useful) input techniques such as head gestures. Head nods and shakes are very simple in the sense that they only provide yes/no, understanding/disbelief, approval/disapproval meanings. However, their importance must not be underestimated because of the following reasons: the meaning of head nods and shakes is almost universal, they can be detected in a relatively simple and robust way and they can be used as the minimum feedback for learning new capabilities.

The system for nod/shake detection described in [9] achieves a recognition accuracy of 78.46%, in real-time. However, the system uses complex hardware and software. An infrared sensitive camera synchronized with infrared LEDs is used to track pupils, and a HMM based pattern analyzer is used to the detect nods and shakes. The system had problems with people wearing glasses, and could have problems with earrings too. The same pupil-detection technique was used in [10]. That work emphasized the importance of the timing and periodicity of head nods and shakes. However, in our view that information is not robust enough to be used. In natural human-human interaction, head nods and shakes are sometimes very subtle. We have no problem in recognizing them because the question has been clear, and only the YES/NO answers are possible. In many cases, there is no periodicity at all, only a slight head motion. Of course, the motion could be simply a ’Look up’/’Look down’/’Look left’/’Look right’, though it is not likely after the question has been made.

For our purposes, the nod/shake detector should be as fast as possible. On the other hand, we assume that the nod/shake input will be used only after the robot has asked something. Thus, the detector can produce nod/shake detections at other times, as long as it outputs right decisions when they are needed. The major problem of observing the evolution of simple characteristics like intereye position or the rectangle that fits the skin-color blob is noise. Due to the unavoidable noise, a horizontal motion (the NO) does not produce a pure horizontal displacement of the observed characteristic, because it is not being tracked. Even if it was tracked, it could drift due to lighting changes or other reasons. In practice, a horizontal motion produces a certain vertical displacement in the observed characteristic. This, given the fact that decision thresholds are set very low, can lead the system to error. The performance can be even worse if there is egomotion, like in our case (camera placed on a head with pan-tilt).

The proposed algorithm uses the pyramidal Lucas-Kanade tracking algorithm described in [11]. In this case, there is tracking, and not of just one, but multiple characteristics, which increases the robustness of the system. The tracker looks first for a number of good points to track, automatically. Those points are accentuated corners. From those points chosen by the tracker we can attend to those falling inside the rectangle that fits the skin-color blob, observing their evolution. Note that even with the LK tracker there is noise in many of the tracking points. Even in an apparently static scene there is a small motion in them. The procedure is shown in Algorithm 1.

The method is shown working in Figure 5. The LK tracker allows to indirectly control the number of tracking points. The larger the number of tracking points, the more robust (and slow) the system. The method was tested giving a recognition rate of

Algorithm 1 LK tracking-based head nod/shake detector

```
repeat
  Compute the absolute displacement of eack tracking point
  Let (Mv,Mh) be the mean absolute displacement of the points inside the skin-color rectangle
  if an output has not been given yet then
    if Mv > threshold OR Mh > threshold then
      if Mv > Mh then
        output=head nod
      else
        output=head shake
      end if
    end if
  end if
until an output is available
```

100% (73 out of 73, questions with alternate YES/NO responses, using the first response given by the system).

Fig. 5. Head nod/shake detector.

What happens if there are small camera displacements? In order to see the effect of this, linear camera displacements were simulated in the tests. In each frame, an error is added to the position of all the tracking points. If (D_x, D_y) is the average displacement of the points inside the skin-color rectangle, then the new displacement is $D_x + e_x$ and $D_y + e_y$. The error, which is random and different for each frame, is bounded by $-e_{max} < e_x < e_{max}$ and $-e_{max} < e_y < e_{max}$. Note that in principle it is not possible to use a fixed threshold because the error is unknown. The error also affects to the tracking points that fall outside the rectangle. Assuming that the objects that fall outside the rectangle are static we can eliminate the error and keep on using a fixed threshold, for $(D_x + e_x) - (F_x + e_x) \approx D_x$ and $(D_y + e_y) - (F_y + e_y) \approx D_y$. For the system to work well it is needed that the face occupies a large part of the image. A zoom lens should be used. When a simulated error of $e_{max} = 10$ pixels was introduced, the recognition rate was 95.9% (70 out of 73). In this case there is a slight error due to the fact that the components F_x and F_y are not exactly zero even if the scene outside the rectangle is static.

Another type of error that can appear when the camera is mounted on a mobile device like a pan-tilt unit is the horizontal axis inclination. In practice, this situation is common, especially with small inclinations. Inclinations can be a problem for deciding between a YES and a NO. In order to test this effect, an inclination error was simulated in the tests (with the correction of egomotion active). The error is a rotation of the displacement vectors $\mathbf{D}$ a certain angle α clockwise. Recognition rates were measured for different values of α, producing useful rates for small inclinations: 90% (60 out of 66) for $\alpha = 20^o$, 83.8% (57 out of 68) for $\alpha = 40^o$ and 9.5% (6 out of 63) for $\alpha = 50^o$.

5 Conclusions

Three simple but useful computer vision techniques have been described, suitable for human-robot interaction. First, an omnidirectional camera setting is described that can detect people in the surroundings of the robot, giving their angular positions and a rough estimate of the distance. The device can be easily built with inexpensive components. Second, we comment on a color-based face detection technique that can alleviate skin-color false positives. Third, a simple head nod and shake detector is described, suitable for detecting affirmative/negative, approval/dissaproval, understanding/disbelief head gestures. The three techniques have been implemented and tested on a prototype social robot.

Acknowledgments

This work was partially funded by research projects *PI2003/160* and *PI2003/165* of Gobierno de Canarias and *UNI2002/16*, *UNI2003/10* and *UNI2003/06* of Universidad de Las Palmas de Gran Canaria.

References

1. Fong, T., Nourbakhsh, I., Dautenhahn, K.: A survey of socially interactive robots. Robotics and Autonomous Systems **42** (2003)
2. Liu, X., Zhao, Y., Pi, X., Liang, L., Nefian, A.: Audio-visual continuous speech recognition using a coupled Hidden Markov Model. In: IEEE Int. Conference on Spoken Language Processing. (2002) 213–216
3. McCowan, I.: Robust speech recognition using microphone arrays. PhD thesis, Queensland University of Technology, Australia (2001)
4. Deniz, O., Castrillon, M., Lorenzo, J., Guerra, C., Hernandez, D., Hernandez, M.: CASIMIRO: A robot head for human-computer interaction. In: Proceedings of 11th IEEE International Workshop on Robot and Human Interactive Communication (ROMAN'2002). (2002) Berlin, Germany.
5. Castrillon, M.: On Real-Time Face Detection in Video Streams. An Opportunistic Approach. PhD thesis, Universidad de Las Palmas de Gran Canaria (2003)

6. Darrell, T., Gordon, G., Harville, M., Woodfill, J.: Integrated person tracking using stereo, color, and pattern detection. In: Procs. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Santa Barbara, CA (1998) 601–608
7. Moreno, F., Andrade-Cetto, J., Sanfeliu, A.: Localization of human faces fusing color segmentation and depth from stereo (2001)
8. Grange, S., Casanova, E., Fong, T., Baur, C.: Vision based sensor fusion for human-computer interaction (2002)
9. Kapoor, A., Picard, R.: A real-time head nod and shake detector (2001)
10. Davis, J., Vaks, S.: A perceptual user interface for recognizing head gesture acknowledgements. In: Proc. of ACM Workshop on Perceptual User Interfaces, Orlando, Florida (2001)
11. Bouguet, J.: Pyramidal implementation of the Lucas Kanade feature tracker. Technical report, Intel Corporation, Microprocessor Research Labs, OpenCV documents (1999)