ACCV2002: The 5th Asian Conference on Computer Vision, 23--25 January 2002, Melbourne, Australia.

Wheelchair Detection in a Calibrated Environment

Ashish Myles
University of Florida
marcianx@visto.com

Dr. Niels Da Vitoria Lobo
University of Central Florida
niels@cs.ucf.edu

Dr. Mubarak Shah
University of Central Florida
shah@cs.ucf.edu

Abstract

This paper describes a method of detecting wheelchairs in 3D for automated assistance for the disabled. It pre-processes each frame by using background subtraction to remove irrelevant data. It then uses skin detection and a variation of the Hough Transform to locate the face of the person sitting in the wheelchair and the wheels of the wheelchair, respectively. After the raw data output from the previous steps is pre-processed, this information is combined with the parameters of the calibrated background to calculate the world coordinates of the wheelchair.

1. Introduction

Tracking wheelchairs in images is useful for the implementation of automated assistance technology for the disabled. Specific applications of wheelchair detection and tracking include doors that respond automatically to the presence of wheelchairs and autonomous mobile agents that are triggered by motions of a wheelchair that might imply a need for assistance. The wheelchair model detected and tracked in the algorithm described in this paper consists of two parallel wheels touching the floor and a head within a box vertically above the midpoint of the axle of the two wheels (Figure 1). The two wheels and the face are the primary features being detected and tracked in the image. Other areas of the wheelchair are not modeled in any specific manner since they are either not generic or easily detectable. These areas are determined through background subtraction and tracked over time.

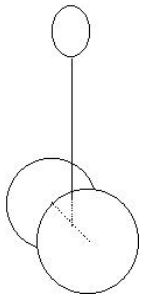

Figure 1: Simple wheelchair model

2. Assumptions

The wheelchair detection and tracking algorithm makes certain important assumptions.

1. The camera is stationary and upright (i.e. no rotation around the optical axis is allowed; all other rotations are allowed).
2. The camera has a little pitch with respect to the ground. This is necessary to determine the orientation of the wheel. It is also important that this pitch is small.
3. The camera calibration information with respect to the ground is known.
4. The face of the person in the wheelchair is exposed; it is not merged with any other skin regions; and it is detectable via a color predicate.
5. The approximate width of the wheelchair is given.
6. The wheel of the wheelchair is large enough to be detected in the presence of considerable noise.
7. All parts of the wheelchair fitting the wheelchair model described above are present in the image.
8. In the detection phase, it is assumed that the wheels of the wheelchair are at such an angle that they are detectable via the Hough Transform (i.e. they are nowhere close to being perpendicular to the image plane).

3. Method

The wheelchair detection method consists of many steps. The following steps are performed in every image in the sequence.

1. Background subtraction
2. Skin detection
3. Hough transform
4. Preprocessing of data
5. Wheelchair detection

After each frame has been processed, a wheelchair-tracking phase is entered to describe the motion of the wheelchair.

3.1. Background subtraction

This part requires a number of still frames of the background. To remove the background, the maximum and minimum background color components – red, green, and blue – are determined for every pixel from this set of frames. When processing new frames, any pixel found to be within its corresponding color range is interpreted as background and discarded. The foreground area thus extracted is dilated by one pixel to fill up any small holes in it. For the purpose of the ground calibration, the shadow area where the wheelchair touches the ground needs to be removed. This can be accomplished as in [4] by comparing a pixel's U/Y and V/Y ratios with the equivalent background pixel to check if the color is the same. In this case, the pixel is marked shadow if its intensity is lower than that of the background pixel. However, as seen in Figure 2(b), the background subtraction step results in non-shadow areas marked as shadow areas. This does not pose a problem since the shadow removal is only for determining the bottom of the closer wheel.

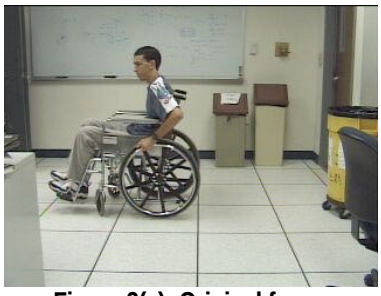

Figure 2(a): Original frame

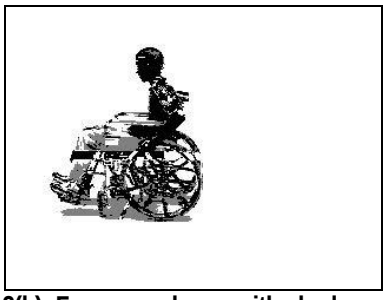

Figure 2(b): Foreground area with shadow areas in gray and non-shadow areas in black

3.2. Skin detection

After the foreground areas of the image are extracted, the color predicate algorithm for skin detection [3] is used to detect possible face areas in the frame. This skin detection phase consists of two steps: a learning step and a detection step.

3.2.1. Learning. The learning step takes as input training images with skins areas and bit masks that mark these skin areas. The colors learned are stored in the form of an accumulator spanning the color space (in this case three-dimensional RGB space). For every pixel in a training image that is marked in the bit mask, its three-dimensional RGB color coordinate is located in the accumulator. Centered at this location, a three-dimensional Gaussian volume is added to support that color. Correspondingly, for every pixel not marked as skin color, a three-dimensional Gaussian volume of a lesser magnitude is subtracted at the corresponding RGB coordinate. After all the images have been learned, all the colors in the accumulator above a certain threshold are marked as skin colors.

3.2.2 Detection. The detection step takes as input am image in which skin needs to be detected. All pixels in the input image that are of skin color as per the thresholded accumulator determined in the learning stage are marked as skin. Skin regions are formed by 8-connectedness of pixels of skin color and are thresholded with respect to their area to remove any spurious regions. The final regions are merged based on their proximity so that a face separated by facial hair is still counted as one region. Only the centroid, area, elongation, and orientation are stored per region.

Please refer to Figure 3. In the figure, the found regions are shown in solid colors overlaid on the original image. In this example, two such regions are found: one for the complete face, and the other for two segments of the arm. The region-merging algorithm overcomes the gap between the hand and the forearm.

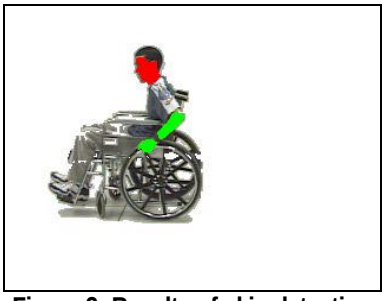

Figure 3: Results of skin detection followed by region merging

3.3. Hough transform for ellipses

Before the Hough transform is performed, the Canny edge detector is used to extract the edges. Irrelevant edges are removed using the foreground area determined from the background subtraction step and a priori information about the location of the background floor. Considerable noise – especially in the form of small loops – is generally present in the vicinity of the wheels due to holes between

the spokes of the wheels and background variation. This noise is removed for better performance by again using connected components to extract various curves in the images and removing those below a certain threshold of length. A higher threshold is used for compact curves than for elongated curves since the former is more likely to represent noise that is not part of the elliptical projection of the wheels in the image. Refer to Figure 4 for the results of edge detection followed by noise removal.

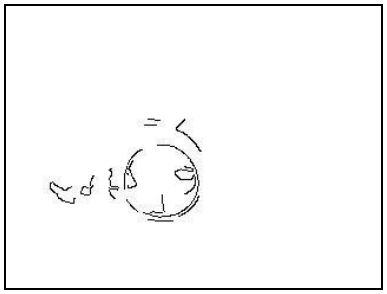

Figure 4: Edge map after noise removal

Then, ellipses are extracted from the image using an efficient variation of the Hough transform [1] that accumulates on every pair of points, utilizing the positional derivatives at each point. Unfortunately, these derivatives can be thrown off in the presence of considerable noise and a large Gaussian mask, while very small Gaussian masks are not too accurate either. However, due to the low resolution of the input data (320x200), we used a sigma of 0.5 to prevent distortion in the edge map.

This ellipse detection algorithm represents its ellipses in the parametric form as:

$$[1] \quad x(q) = a_0 + a_x \sin(q) + b_x \cos(q)$$
$$y(q) = b_0 + a_y \sin(q) + b_y \cos(q),$$

where (a_0, b_0) is the center of the ellipse and (a_x, a_y) and (b_x, b_y) are vectors representing the major and minor axes of the ellipse (Figure 5).

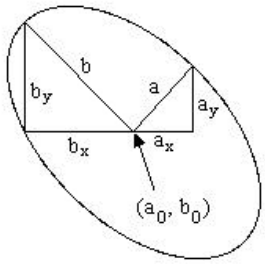

Figure 5: Ellipse representation

The first part of the algorithm described in [1] is to detect the edges and store the gradient magnitude and direction at every point. Eligible pairs of edge points are chosen to prevent divide-by-zeroes and other error-prone situations. For every pair of eligible points $(x(\theta_1), y(\theta_1))$ and $(x(\theta_2), y(\theta_2))$:

1. Calculate the first and second derivatives of the point $(x_p, y_p) = (x(\theta_{av}), y(\theta_{av}))$, where $\theta_{av} = \frac{1}{2}(\theta_1 + \theta_2)$ as follows:

   $$[2] \quad y_p' = \frac{Y}{X}, \quad y_p'' = \frac{2M_1 X - YM_2}{XM_2 - 2Y}$$

   for

   $$Y = y_2 - y_1, \quad X = x_2 - x_1,$$
   $$M_1 = y_1' y_2', \quad M_2 = y_1' + y_2'$$

2. Increment cells in the a_0 vs. b_0 accumulator by varying a_0 and calculating b_0 using the equation:

   $$[3] \quad b_0 = y_m - y_p''(x_m - a_0)$$

   where (x_m, y_m) is the average of the pair of points being used.

3. Increment cells in the K vs. N accumulator, where N is the ratio b_y/a_x, and K is the ratio a_y/a_x. Vary K and calculate N using the relationship:

   $$[4] \quad -N^2 = \tan(\Phi_1 - r)\tan(\Phi_2 - r)$$

   where $\Phi_1 = \arctan(y_p')$, $\Phi_2 = \arctan(y_p'')$, and $\rho = \arctan(K)$.

After all the votes have been accumulated, the next part finds the local maxima. For each pair of (a_0, b_0) and (K, N) maxima, a third accumulator is run to find the value of a_x. For every edge point, the cell corresponding to the value of a_x calculated through the equation below is incremented.

$$[5] \quad a_x = \sqrt{\frac{y_0^2 + x_0^2 N^2}{N^2(1+K^2)}}$$

where

$$x_0 = \frac{(x - a_0) + (y - b_0)K}{\sqrt{1+K^2}}$$
$$y_0 = \frac{-(x - a_0)K + (y - b_0)}{\sqrt{1+K^2}}$$

Peaks in the a_x histogram are determined and the rest of the ellipse parameters can be derived from each set of (a_0, b_0, K, N, a_x) using the following equations:

[6] $$a_y = Ka_x, \quad b_y = Na_x, \quad b_x = -a_y b_y / a_x$$

The third equation is merely an orthogonality constraint for the axes of an ellipse. After ellipses are detected in their analytical form, they are checked against the original edge map for verification. Figure 6 shows the results of the Hough transform.

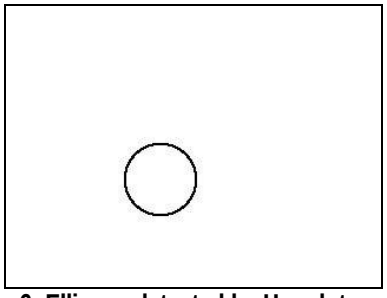

Figure 6: Ellipses detected by Hough transform

3.4. Preprocessing of data

Ellipses and regions are filtered to remove those that do not represent wheels and faces, respectively. A skin region is removed if it is too elongated or too large. An ellipse is removed when one of the following conditions is met.

1. The ratios of the lengths of its axes are too great or too small.
2. It is too large or too small (checked by thresholding the average length of the ellipse axes).
3. It is not sufficiently close to the ground.
4. It is too horizontally elongated to be circular in 3D.
5. It is too tilted to be a wheel.

In addition, ellipses that are concentric and similar in size and shape are merged. Ellipses that are circular are turned into perfect circles, and all the ellipses are verticalized to ease the calculation of their world coordinates. Since wheels tend to make concentric ellipses on an image and in the edge map, one cannot rely on the ellipse output of the Hough Transform to return the outer-most ellipse formed by a wheel. Therefore, the foreground area without the shadow areas is used to fit all ellipses to the ground. This foreground area is traversed vertically downward from the bottom of the ellipse until the bottom-most point of the area is reached. That is interpreted to be the point where the wheel touches the floor and the ellipse is expanded to fit this point.

3.5. Wheelchair detection

The detection algorithm sews the raw information gathered by the previous steps together to construct a wheelchair in the image and to locate it in room coordinates.

The conversion to room coordinates is a two-step process. Firstly, the 3D coordinates with respect to the camera are calculated. This step requires the depth Z_{cam} of the coordinate with respect to the focal point of the camera. Once the depth is known, the X_{cam} and Y_{cam} of the center of the circle can be calculated using the following two formulas:

[7] $$X_{cam} = x_{2d} Z_{cam} / f, \quad Y_{cam} = y_{2d} Z_{cam} / f$$

where (x_{2d}, y_{2d}) is the image coordinate with respect to the center of the image, and f is the focal length of the camera in pixels. The focal length in pixels can be determined by sampling a few data points with known camera coordinates.

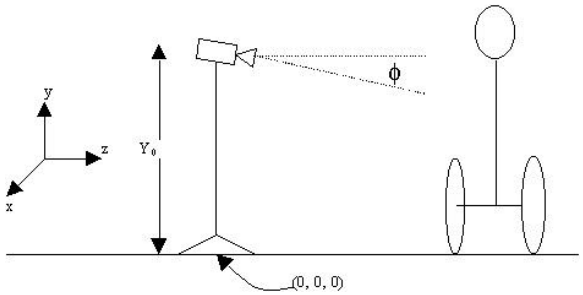

Figure 7: Room coordinates

The second step is the conversion of the coordinate system for easier calculations and wheelchair detection. The new coordinate system has its origin at the bottom of the camera stand with its xz-plane parallel to the floor (Figure 7). This transformation involves a rotation of the yz-plane by the pitch ϕ of the camera followed by a translation downward. This is accomplished by the following equations:

[8] $$\begin{aligned} X_{room} &= X_{cam} \\ Y_{room} &= Y_{cam}\cos(f) - Z_{cam}\sin(f) + Y_0 \\ Z_{room} &= Y_{cam}\sin(f) + Z_{cam}\cos(f) \end{aligned}$$

where ϕ is the pitch, and Y_0 is the height of the camera.

The conversion to room coordinates for the wheel is fairly straightforward. The depth Z_{cam} of a wheel with respect to the camera is calculated using the location of the bottom of the ellipse and the calibration parameters of the background. The rest is calculated with the formulas above. Theoretically, the radius and the room y-coordinate of the wheels should be the same. However,

for verification, the radius is calculated alternatively as the distance between the camera-coordinates of the top and bottom points on the ellipse (assuming that ϕ does not result in a major difference between the depths of the two points). This value can be compared with the y-coordinate of the wheel center for verification.

A wheelchair is initialized if one or two ellipses and a face (skin region) fit the wheelchair model. Calculating the 3D representation of the wheels brings about an ambiguity in the orientation of the wheel since every non-circular ellipse represents two possible 3D circles. The orientation is then determined using the fact that even a slight pitch in the camera results in wheels that are not parallel to the image plane to be projected as skewed ellipses instead of vertical ellipses in the image. The direction of the skew reflects the orientation of the wheel (Figure 8).

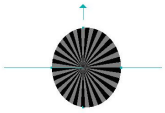

Figure 8(a): A 3D disk rotated 30° around the vertical axis with no rotation around the horizontal axis results in a vertical ellipse

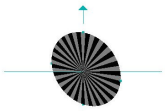

Figure 8(b): The same disk with a little rotation about the horizontal axis results in a skewed ellipse

In the case that the farther wheel is not found through the Hough transform (primarily due to occlusion), it is predicted using the closer wheel. The second wheel is then fitted to the edge map to improve accuracy in calculating the orientation of the wheelchair.

Once the two wheels have been located in 3D, the room coordinates of the face can be computed. If the face is assumed to be directly above the midpoint of the axis connecting the wheels – which is only approximately the case in the data set used – the average depth of the two wheels can be used to approximate the depth of the head in room coordinates. The camera depth of the head can then be derived as follows:

[9]
$$Z_{room} = Y_{cam}\sin(f) + Z_{cam}\cos(f)$$
$$\Rightarrow Z_{room} = (y_{2d}Z_{cam} / f)\sin(f) + Z_{cam}\cos(f)$$
$$\Rightarrow Z_{cam} = \frac{Z_{room}}{(y_{2d} / f)\sin(f) + \cos(f)}$$

Once the camera depth of the face is calculated, the room coordinates are straightforward from equations [7] and [8].

Figure 9 illustrates the results of the wheelchair detection step. At the top of the figure, we have printed the room coordinates in feet of the center of the closer and farther wheels and the centroid of the head, respectively. The fourth number in the wheel coordinates is the radius of the wheel in feet, and the fifth coordinate is its orientation in degrees with respect to the xy-plane. For reference, the closest horizontal line on the floor corresponds to a depth of eight feet, and adjacent lines are two feet apart. The locations of the detected wheels and face have been superimposed on the image.

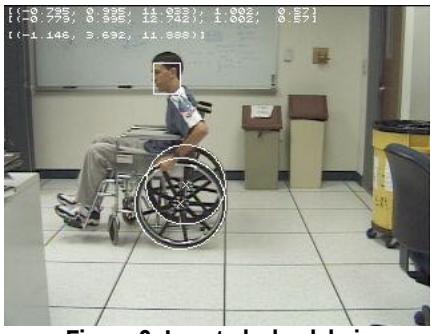

Figure 9: Located wheelchair
The information written at the top of the image is:
[(-0.795, 0.995, 11.033), 1.002, 0.57]
[(-0.779, 0.995, 11.033), 1.002, 0.57]
[(-1.146, 3.692, 11.888)]

3.6. Wheelchair tracking

The tracking phase is different from the other portions of the method since it is not on a per-frame basis. For initialization, the wheel is tracked for three frames until it is considered fully detected. Radii of the wheels are averaged over the frames to determine the actual radius, which is then held constant. The path taken by the wheelchair is smoothed out and used to correct the orientations of the wheelchairs. The wheelchair is then tracked until the wheel's projection is no longer detectable using the Hough transform.

4. Results

The following are the results of wheelchair detection on a few frames of the movie. Frames 110 and 492 were fairly well detected. This is primarily because there was enough of the farther wheel in the edge map that it could

be easily fit to this edge data when it is predicted. In frame #79, however, the farther wheel is falsely fit because it is lost in the background noise and is not visible in the edge map. Frame #35 has the same problem, but to a higher degree. The tracking phase has not yet been completed and hence is not shown in the figures below.

5. Conclusion and Future Work

This paper presents a method for detecting wheelchairs in a calibrated scene. It uses a variation of the Hough transform algorithm to detect wheels and the color predicate algorithm to locate the face of the person on the wheelchair. Inaccuracies in the first wheel's orientation lead to error in the prediction of the location of the farther wheel. These inaccuracies result from occlusion of the second wheel and other background effects. The improvements to the fit of this second wheel will come from three sources. Firstly, one could use better methods for detecting the second wheel despite occlusion. The second is to compute the orientation of the first wheel in a more accurate manner. For example, one can map the original non-verticalized ellipse to the 3D circle that it represents. The third improvement will come when tracking the body and chair is improved and is used to provide better prediction for the second wheel. The tracking itself can be made robust by tracking other stationary parts of the body – including the legs and the torso – and deriving orientation through their motion. Also, the wheelchair can be better verified through the detection of the arms or just hands at the expected locations via skin detection.

6. References

[1] A.S. Aguado, M.E. Montiel, M.S. Nixon, Ellipse Detection via Gradient Direction in the Hough Transform, *Proc. IEE 5th International Conference on Image Processing and its Applications,* pp. 375-378, July 3-6, Edinburgh, United Kingdom, 1995.

[2] J. Canny. A Computational Approach to Edge Detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 8, No. 6, Nov. 1986.

[3] R. Kjeldsen and J.R. Kender. *Finding Skin in Color Images*, in Proceedings of the Second International Conference on Automatic Face and Gesture Recognition, October 14-16, Killington VT, 1997.

[4] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-Time Tracking of the Human Body, *SPIE Conference on Integration Issues in Large Commercial Media Delivery Systems*, volume 2615, 1995.

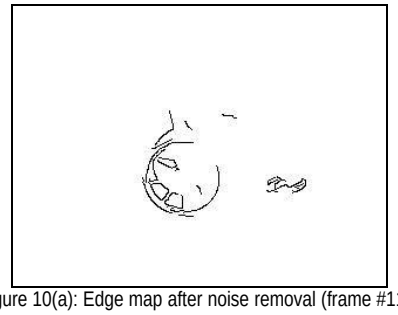

Figure 10(a): Edge map after noise removal (frame #110)

Figure 10(b): Located wheelchair (frame #110)
The information written at the top of the image is:
[(-0.388, 1.008, 11.202), 1.016, 0.86]
[(-0.363, 1.008, 12.911), 1.016, 0.86]
[(0.031, 3.655, 12.057)]

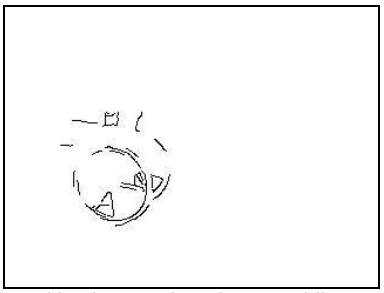

Figure 10(c): Edge map after noise removal (frame #492)

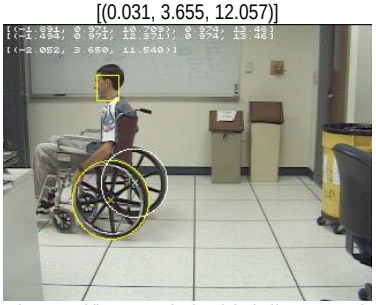

Figure 10(d): Located wheelchair (frame #492)
The information written at the top of the image is:
[(-1.891, 0.971, 10.709), 0.974, 13.46]
[(-1.494, 0.971, 12.371), 0.974, 13.46]
[(-2.052, 3.650, 11.540)]

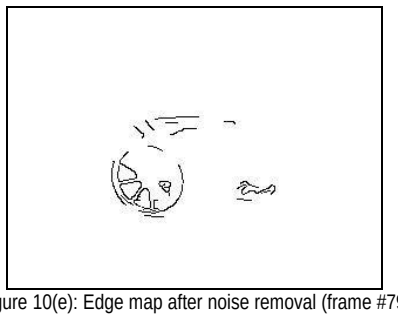

Figure 10(e): Edge map after noise removal (frame #79)

Figure 10(f): Located wheelchair (frame #79)
The information written at the top of the image is:
[(-1.152, 0.962, 11.364), 0.970, 3.78]
[(-1.039, 0.962, 13.068), 0.970, 3.78]
[(-0.651, 3.709, 12.216)]

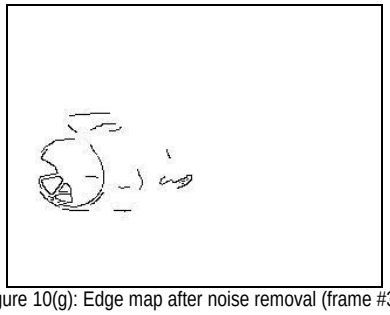

Figure 10(g): Edge map after noise removal (frame #35)

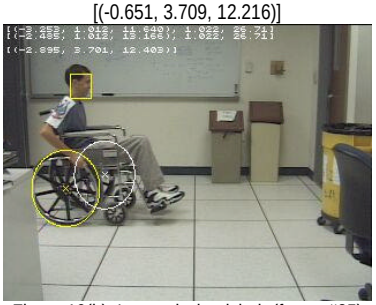

Figure 10(h): Located wheelchair (frame #35)
The information written at the top of the image is:
[(-3.253, 1.012, 11.640), 1.022, 26.71]
[(-3.485, 1.012, 13.166), 1.022, 26.71]
[(-2.895, 3.701, 12.403)]

Figure 10: Various wheelchair detection results