OBSTACLE DETECTION AND AVOIDANCE ON SIDEWALKS

D. Castells
Universidad Politécnica de Madrid
delicast@gmail.com

J.M.F. Rodrigues and J.M.H. du Buf
Vision Laboratory, Institute for Systems and Robotics (ISR), University of the Algarve (ISE and FCT), 8005-139 Faro, Portugal
{*jrodrig, dubuf*}*@ualg.pt*

Keywords: Sidewalk border detection; obstacle avoidance; path tracking; visually impaired.

Abstract: We present part of a vision system for blind and visually impaired people. It detects obstacles on sidewalks and provides guidance to avoid them. Obstacles are trees, light poles, trash cans, holes, branches, stones and other objects at a distance of 3 to 5 meters from the camera position. The system first detects the sidewalk borders, using edge information in combination with a tracking mask, to obtain straight lines with their slopes and the vanishing point. Once the borders are found, a rectangular window is defined within which two obstacle detection methods are applied. The first determines the variation of the maxima and minima of the gray levels of the pixels. The second uses the binary edge image and searches in the vertical and horizontal histograms for discrepancies of the number of edge points. Together, these methods allow to detect possible obstacles with their position and size, such that the user can be alerted and informed about the best way to avoid them. The system works in realtime and complements normal navigation with the cane.

1 INTRODUCTION

Every car and bicycle can be equipped with a GPS/GIS-based navigation system that may cost a few hundreds of euros. By contrast, blind and visually impaired persons need to navigate using the stick or, at best, an ultrasonic obstacle detector. This asymmetry needs to be solved, because there are an estimated 180 million persons with severe impairments of which 40-50 million are completely blind, and every year 2 million more become blind. The Portuguese project "SmartVision: active vision for the blind" aims at developing a portable GIS-based navigation aid for the blind, for both outdoor and indoor navigation, with obstacle avoidance and object recognition based on active vision modules.

There are a few recent systems for visually impaired users which may assist them in navigation, with and without obstacle detection and avoidance, e.g., Lee and Kang (2008) who developed a system which integrates outdoor navigation and obstacle detection. Kim et al. (2009) presented an electronic travel aid called iSONIC. It complements the conventional cane by detecting obstacles at head-height.

The work presented here concerns one of the modules of the SmartVision project. This module serves to detect sidewalk borders and assists the user in centering on the sidewalk, thereby avoiding any obstacles. Typical obstacles are light poles, trash cans and tree branches, also imperfections as holes and loose stones, at a distance of 3 to 5 meters from the user. The system automatically adapts to different types of sidewalks and paths, and it works in realtime on a normal portable computer.

2 SYSTEM SETUP

In the SmartVision project, a stereo camera (Bumblebee 2 from Point Grey Research Inc.) is fixed to the chest of the blind, at a height of about 1.5m from the ground. Results presented here were obtained by using only the right-side camera, but the system performs equally well using a normal, inexpensive webcam with about the same resolution. The resolution must be sufficient to resolve textures of the pavements related to possible obstacles like holes and loose stones with a minimum size of about 10 centimeters at a distance of 3 to 5 meters from the cam-

era (the first meters are not covered because of the height of the camera; this area is covered by the cane swayed by the user). Figure 1 (top) shows a typical frame; Fig. 4 shows one of our test sequences.

The system is composed of three processing steps: (1) Sidewalk border detection and the definition of the obstacle detection window (ODW). (2) The detection of obstacles in the ODW using two complementary processes for tracking irregularities: (i) the number of local maxima and minima of pixel values, and (ii) histograms of binary edge information. (3) Tracking of obstacles in subsequent frames for alerting the user and obstacle avoidance.

2.1 Sidewalk border detection

There are some methods to detect the borders of sidewalks (Kayama et al., 2007). Here we detect them by using a simple edge detector in combination with a tracking mask to obtain straight lines, from the bottom of each frame to the top, characterized by slope, length and proximity to the left and right boundaries of the frame. The detected borders will define the horizontal position and width of the ODW.

We apply the Canny edge algorithm (Heath et al., 1997) with three parameters: σ defines the size of the Gaussian filter, and T_h and T_l are the high and low thresholds for hysteresis edge tracking. We always use $\sigma = 1.5$, $T_h = 0.95$ and $T_l = 0.5$. Figure 1 (2nd image from top) shows edges detected in the case of the frame shown above. Other edge detectors may perform better, see Rodrigues and du Buf (2009), but most require more CPU time which is critical in this application.

In order to detect potential sidewalk borders, several horizontal test lines are defined in the binary edge image, $I_i(x,y)$ with i the frame number. Figure 1 (2nd image) shows these on the left (in red). On the first test line "1TL" there are many edge pixels which may be part of potential border lines. They are starting points which are labelled differently for the testing process. A line-tracking mask of size 5x3 (Fig. 1, bottom-right) is applied to track connected pixels upwards, vertically with an opening angle $\alpha \approx 120$ deg, from "1TL" to "2TL", 20 pixels higher, and connected pixels are attributed the same label. Then, starting at "2TL", the process is repeated for finding more potential border lines, complementing the first search. New labels are generated for pixels on "2TL" which are not connected to pixels on the first test line. This second search continues until the last testing line "last TL", 200 pixels above "1TL". Figure 1 (3rd row, at left) shows the mask tracking edge pixels with label 43 and (at right) tracked edge pixels with a small gap which will be filled.

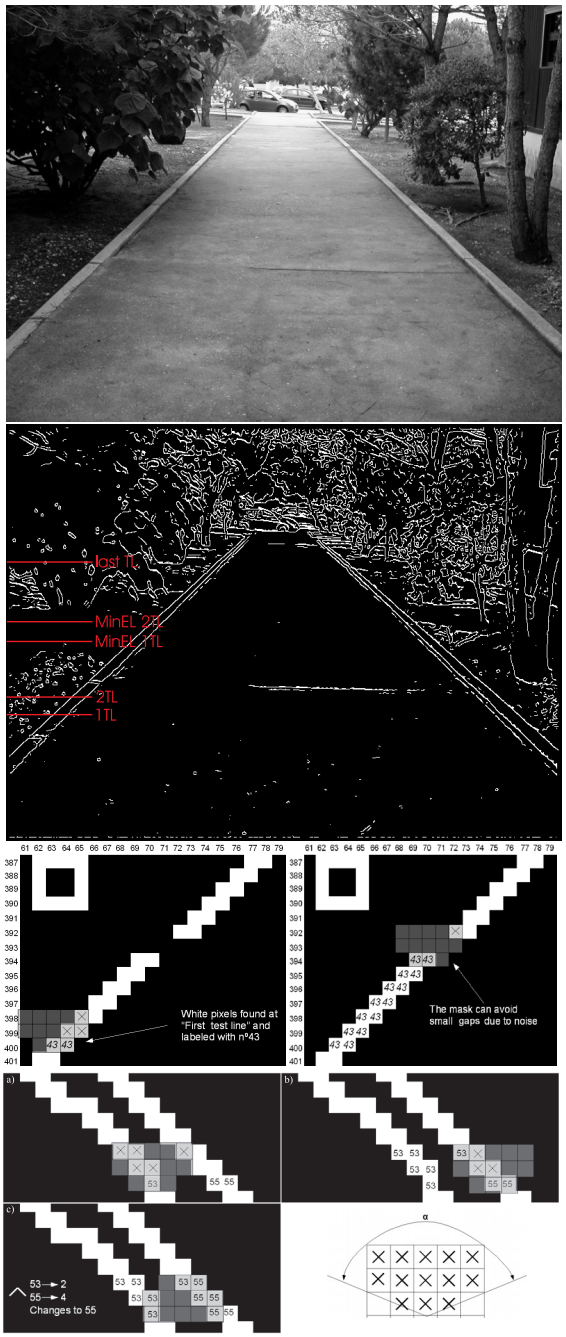

Figure 1: Sidewalk border detection. Top to bottom: an input frame; edge detection with testing lines indicated (in red) at the left; start of tracking edge pixels at the first test line (1 TL) with label 43 (at left) plus the filling of a small gap (at right); the three steps a, b and c for coping with very close lines; and the line tracking mask with angle α.

Occasionally, there are two or more labeled edge pixels which are very close. These could correspond to a single or to several border lines. In those cases, when the mask is applied to a new edge pixel, the already labelled pixels below are checked by using

the vertically mirrored mask. If there are many pixels with a label not equal to the label of the mask's central pixel, the label of the central pixel changes to the one of the majority of the pixels in the mask. The 4th and 5th rows in Fig. 1 show an example of this process with steps a, b and c. Final images with connected and labelled edge pixels are denoted by $L_i(x,y)$, but these still contain *potential* border lines. To be considered sidewalk borders, detected lines must satisfy the following three requirements:

(1) Connected edge pixels must have a minimum length (MinEL) covering at least 80 vertical positions (MinEL 1TL or MinEL 2TL, depending on the line starting on 1TL or on 2TL, see Fig. 1, 2nd image from top). Shorter series are removed.

(2) Connected edge pixels must be almost linear, i.e., with correlaton $r > 0.9$, with a slope $|b|$ between 0.5 and 10. The slope also provides information about the sidewalk's width: the higher the slope, the narrower the sidewalk. In order to speed up the correlation/slope process, only eight equidistant points of each potential line are processed. The correlation r of edge pixels with the same label is given by $r = \sigma_{xy}^2/\sqrt{\sigma_x^2 \cdot \sigma_y^2}$, where σ_{xy}^2 is the covariance $\sigma_{xy}^2 = \sum xy/n - (\sum x \cdot \sum y)/n^2$ and σ_x^2 and σ_y^2 are the variances of x and y

$$\sigma_x^2 = \frac{\sum x^2}{n} - \left(\frac{\sum x}{n}\right)^2; \sigma_y^2 = \frac{\sum y^2}{n} - \left(\frac{\sum y}{n}\right)^2. \quad (1)$$

The regression line is given by $L = a + bx$, with

$$b = \frac{\sum x \cdot \sum y - n\sum xy}{(\sum x)^2 - n\sum x^2}; a = \frac{\sum x \cdot \sum xy - \sum y \cdot \sum x^2}{(\sum x)^2 - n\sum x^2}. \quad (2)$$

(3) Occasionally, more than two lines remain as potential sidewalk borders, or none at all. Depending on the number of lines n_l, the following is done: (i) If no lines are found, $n_l = 0$, the last two borders found in a previous frame will be used. (ii) If a single line is found, $n_l = 1$, the second line will be automatically generated, symmetrically with respect to the vertical line that passes through the intersection of the line found and the horizontal line through the vanishing point. The latter is updated dynamically for each new frame, as explained below. (iii) If two lines are found, $n_l = 2$, they are accepted as sidewalk borders, but with the following exception: If the signs of the slopes of the two lines are not different, this means that we do not have the right lines. The outermost one is ignored, the innermost one is used, and a new line is generated as in case $n_l = 1$. Here, innermost means closest to the center of the frame and outermost closest to the left or right frame borders. (iv) In the case of more lines, $n_l > 2$, the most symmetrical and inner pair of lines is selected.

Above, the vanishing point is used to generate symmetrical line pairs. At the start of a sequence of frames, or when no acceptable sidewalk borders can be detected, the height of the vanishing point will be initialized at $3/4$ of the frame height. Then, when two correct sidewalk borders are found, the vanishing point is determined by the intersection of the two borders, and the point is dynamically updated by averaging the points of the previous frame and the new frame.

After obtaining two valid sidewalk borders, the **obstacle detection window** (ODW) is defined for detecting and locating possible obstacles. This window has predefined upper and a lower limits with a height of $N_{v,ODW} = 100$ pixels (v is vertical); see Section 2. The left and right limits are defined by taking 80% of the distance between the two borders found at the upper limit, which gives $N_{h,ODW}$ pixels (h is horizontal).

3 OBSTACLE DETECTION AND AVOIDANCE

For obstacle detection, two different methods are applied to the ODW. The first one counts variations of gray values, i.e., local maxima and minima, on each horizontal line inside the ODW. Then, outliers are reduced by averaging counted variations over groups of lines, and variations over these groups are combined into a single value which indicates whether the frame has a possible obstacle or not. Final confirmation is obtained by combining the results of a few subsequent frames. The second method is based on irregularities in vertical and horizontal histograms of the binary edge image I_i. An obstacle can lead to two different signatures: if the pavement is smooth, an obstacle may appear as a local excess of edge points, but if it has a strong texture there will be a huge amount of edge points and an obstacle may appear as a local deficiency (lack or gap) of edge points. The second method is used to confirm the result of the first one, but it also serves to detect the size and position of an obstacle in order to guide the user away from it.

3.1 Local maxima and minima

(a) A small lowpass filter (averaging block filter of size 3x3; $LP(x,y)$) is applied twice to the graylevel ODW of frame i ($F_{ODW,i}$), so high frequencies are suppressed and a less noisy window can be processed. If $*$ denotes convolution, then $\tilde{F}_{ODW,i}(x,y) = F_{ODW,i}(x,y) * LP(x,y) * LP(x,y)$.

(b) Then, the variations of gray values on each horizontal line of the window are computed by applying the first derivative $F'_{ODW,i}(x) = \partial(\tilde{F}_{ODW,i}(x))/\partial x$.

(c) In order to keep significant variations, a threshold $T_d = \pm 2$ is applied to the derivative. This suppresses small transitions and maxima and minima can now easily be found: where the derivative changes its sign (zero crossing or *ZC*), $+/-$ for a maximum and $-/+$ for a minimum.
(d) The next step consists of counting on each horizontal line y the number of maxima and minima $MM(y)$ over the ODW window (100 horizontal lines), $MM_i(y) = \sum ZC[T_d[F'_{ODW,i}(x)]]$.
(e) The result is stabilized by removing outliers. This is done by taking the average of *MM* over triplets of lines, i.e., over three consecutive horizontal lines, which results in only 33 values for each ODW. With $k = 0,1,2$ and line counting starting at $y = 0$, $MM_i(y/3) = \sum MM_i(y+k)/3$.
(f) For calculating variations over the ODW's lines, the first derivative is applied to $MM_i(y/3)$, $MM'_i(y/3) = \partial(MM_i(y/3)/\partial y$.
(g) The last processing step of the ODW of frame i consists of determining the maximum value (max) of the absolute value $([\cdot]^+)$ of the derivative from step (f), $max_i = \max[MM'_i(y/3)]^+$. This value indicates a possible obstacle in the ODW.
(h) In order to detect and confirm obstacles, a dynamic threshold is used to alert the user. The dynamic threshold is initialized by computing the average of max_i over the first five frames, $\overline{max} = \sum_{i=1}^{5} max_i/5$, and the average of the deviation, i.e., the difference between $\overline{max}$ and each max_i, $\overline{dev} = \left[\sum_{i=1}^{5}(max_i - \overline{max})/5\right]^+$. The first threshold, $T_6 = \overline{max} + \overline{dev}$, is going to be tested against max_6 obtained from frame 6, after this frame has been processed from step (a) to (g). The same processing is done with max_i, dev_i and T_i. Two conditions can occur for $i > 5$:
(h.1) If max_i does not exceed threshold T_i, a new threshold for the next frame will be calculated by $T_{i+1} = 4/5 \cdot T_i + 1/5 \cdot (max_i + dev_i)$.
(h.2) Otherwise, a warning-level counter is activated. If max_i of the next two frames continue exceeding the threshold, an obstacle warning will be issued. The same happens when, after the warning level has been activated, there is only one frame which does not exceed the threshold. If more than two consecutive frames do not exceed the threshold, the warning counter will be reset and the threshold will continue to be adapted dynamically.

The processing described above detects big variations of the number of local maxima and minima, first in the horizontal lines and then over the lines in the ODW. This allows to detect the appearance and the disappearance of an obstacle in the window, because these coincide with the first and last max_i which exceed the dynamic threshold. For detecting the position and size of an obstacle, we apply an analysis of edge histograms.

3.2 Edge histograms

This method exploits the already available edge maps $I_i(x,y)$, see Section 2.1, but only inside the ODW $I_{ODW,i}(x,y)$. Depending on the smoothness (texture) of a sidewalk's pavement, different characteristics are expected. If r_w is the fraction of the number of "white" (edge) pixels $N_{w,ODW}$ in $I_{ODW,i}(x,y)$, with $N_{h,ODW}$ and $N_{v,ODW}$ the window's dimensions, $r_w = N_{w,ODW}/(N_{h,ODW} \times N_{vODW})$.
Extensive tests with real pavements, but without obstacles, revealed that $r_w > 0.1$ indicates rough surfaces, for example with cobblestones like in Portuguese-style "calçada," whereas smaller values indicate smooth surfaces. In both cases, vertical and horizontal edge histograms are computed, i.e., for each line and for each column in the ODW the number of "white" pixels are summed. Then, the two histograms are smoothed by applying a simple 1D averaging filter.

Two thresholds, T_c and T_l, are computed for the histograms of the lines and columns. For the column histogram, T_c is the ratio between the number of white pixels in the ODW and the number of columns of the window, $T_c = (N_{w,ODW}/N_{h,ODW}) \times K$, and, similarly, T_l is the ratio between number of white pixels and the number of lines, $T_l = (N_{w,ODW}/N_{v,ODW}) \times K$, where K is a normalization factor.

In the case of a smooth pavement $(r_w < 0.1)$, an obstacle will appear as an excess of white points, see Fig. 2, and we apply $K = 0.8$. An obstacle will be detected if at least two neighboring values in both the line and column histograms exceed the thresholds, T_l and T_c, respectively.

In the case of a textured pavement $(r_w \geq 0.1)$, an obstacle will appear as a lack of white points, see Fig. 3, and we apply $K = 0.4$. Now at least two neighboring values in both histograms must be lower than the corresponding thresholds.

3.3 Obstacle avoidance

In order to detect an obstacle, both methods described above must detect something, but in order to save CPU time the histogram method is only applied to frames in which the first method has detected something. Once the histogram method has confirmed the detection, a sound is generated in order to alert the user. This sound is modulated such that it indicates the approximate position and the best way to avoid the obstacle.

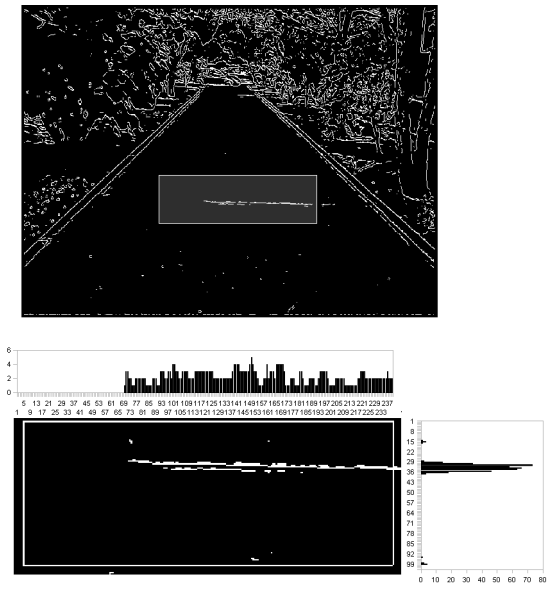

Figure 2: Smooth pavement with a possible obstacle and corresponding edge histograms.

The first method indicates an obstacle somewhere in the entire ODW. Often, but not always because it depends on the textures of the pavement and the object and the latter's size, the histogram method can narrow the approximate position to the object's bounding box. Blind people prefer to walk near walls or the façades of buildings along streets, swaying the cane in front and keeping contact with the wall or façade. Since a wall- and façade-detection algorithm has not yet been implemented, we illustrate walking close to the centerline of paths and sidewalks in this paper. This scenario is also quite realistic, and available information about the path's borders and the vanishing point can be used to inform the user about the centerline. In addition, having the position and dimensions (in pixels) of an obstacle's bounding box, and the dimensions (in pixels and in meters) of the ODW, see Section 2, it is easy to convert the bounding box to meters and provide information about the approximate position and size of the obstacle. If the obstacle is not centered on the path, the user can be informed about the left or right side which has the largest distance between the bounding box and the path's left and right borders. In any case, the user will check the obstacle using the cane.

It should be stressed that the camera is tightly fixed to the person's breast such that it points straight forward, also that the blind have been trained not to sway much with the body while swaying the cane. The obstacle-avoidance module requires initial calibration in order to obtain correct distances, for these depend on each user's length and posture. In the future, a disparity module which uses both cameras

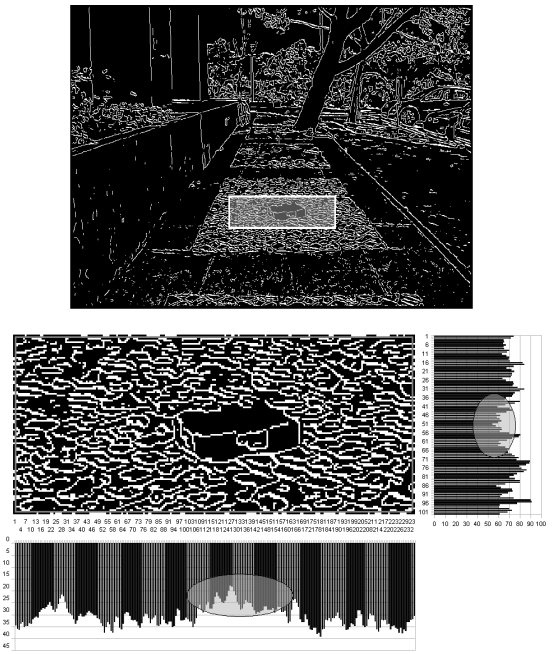

Figure 3: Textured pavement with a possible obstacle and corresponding edge histograms.

of the stereo camera will be integrated. This module will complement the obstacle-detection methods and it will also provide calibrated distance information, such that the calibration mentioned above will no longer be required.

4 CONCLUSIONS AND RESULTS

Various test sequences have been captured on several paths and sidewalks of Gambelas Campus at the University of the Algarve, with different pavements and obstacles. Figure 4 shows one sequence with, apart from the frame number given in the upper-left corner, the following frame annotation: Type 0 concern the first 5 initial calibration frames with no obstacles. Type 1 frames are those in which the variation of maxima and minima in the ODW does not exceed the dynamic threshold. Type 2 frames do exceed the threshold and the first one activates an alert counter which counts, from 1 to N, the number of subsequent Type 2 frames. The third alert ($N = 3$) activates "obstacle warning" which remains until the first new frame of Type 1 is encountered.

The first results are very encouraging. The numbers of false positives and negatives of the sequences tested were quite small, with more false positives than false negatives. False positives were mainly caused by tree leaves and litter, whereas false negatives were mainly cobblestones pushed up a few centimeters by tree roots, such that the irregularity of the calçada's texture is not detectable.

On a portable computer with an Intel Pentium clocked at 1.6 GHz, elapsed time to process individual frames is about 0.5 second. This is already fast enough for realtime application where the user walks at normal speed. By using a new portable with a multi-core processor, more than two frames per second can be processed, but the disparity module will also consume CPU time (although disparity processing might be limited to the ODW), as will other modules for using GPS in combination with a dedicated GIS for autonomous navigation.

In general, good results were obtained in the detection of the borders of paths and sidewalks, and many more paths and sidewalks are now being tested. Difficulties mainly arise when the color difference or contrast of a path's curbs is small, when the curbs are partly hidden by plants and long grass, or when a path has no curbs but is delimited by grass or low shrubs. Similar problems arise in the case of obstacle detection, when the contrast and the texture of an obstacle and those of the pavement are too similar. However, most obstacles, including missing cobblestones in Portuguese-style "calçada," the smallest but most frequent problem, can be detected, but not yet elevated cobblestones which are being pushed up by long tree roots. Therefore, detection algorithms must be improved, even at the cost of more CPU time.

A specific problem is the detection of single and multiple steps, for which no dedicated algorithm has been included yet. Occasionally, a wrong obstacle detection window is caused by a wrong detection of the borders. However, normally this happens in a single frame and the problem can be solved by keeping the alert counter counting such that at the next Type 2 frame an obstacle warning will be issued. This solution, i.e., tracking information over multiple frames, can be applied in many more cases. For example, positions of borders detected in previous frames can be extrapolated to new frames in order to narrow the search area and to confirm the new borders, although sudden and unpredictable movements of the user cannot be excluded unless they are detected by big changes of the global optical flow of entire frames. Such aspects require more research because of the CPU times which are involved. Also being developed is a dynamic adaptation of the parameters of the Canny edge detector as a function of the type of pavement, for resolving finer textures of obstacles like elevated cobblestones, but also for the detection of often minute differences between textures of horizontal and vertical surfaces of steps when the contrast between them is too low.

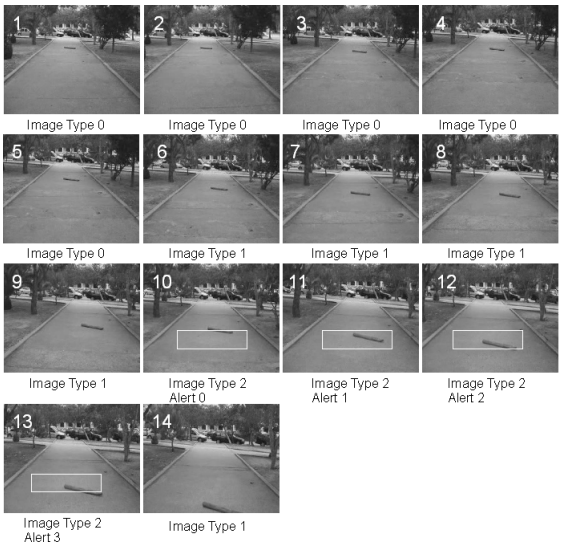

Figure 4: One test sequence with annotation.

Acknowledgements: Portuguese Foundation for Science and Technology (FCT) through the pluriannual funding of the Inst. for Systems and Robotics (ISR/IST), the POS_Conhecimento Program with FEDER funds, and FCT project SmartVision (PTDC/EIA/73633/2006).

REFERENCES

Heath, M., Sarkar, S., Sanocki, T., and Bowyer, K. (1997). A robust visual method for assessing the relative performance of edge-detection algorithms. *IEEE Trans. PAMI*, 19(12):1338–1359.

Kayama, K., Yairi, I., and Igi, S. (2007). Detection of sidewalk border using camera on low-speed buggy. *Proc. IASTED Int. Conf. on Artificial Intelligence and Applications,13-15 February, Innsbruck, Austria*, pages 262–267.

Kim, L., Park, S., Lee, S., and Ha, S. (2009). An electronic traveler aid for the blind using multiple range sensors. *IEICE Electronics Express*, 11(6):794–799.

Lee, S. and Kang, S. (2008). A walking guidance system for the visually impaired. *Int. J. Pattern Recogn. Artif. Intell.*, 22(6):1171 – 1186.

Rodrigues, J. and du Buf, J. (2009). Multi-scale lines and edges in V1 and beyond: Brightness, object categorization and recognition, and consciousness. *BioSystems*, 95:206–226.