Crosswalk Localization from Low Resolution Satellite Images to Assist Visually Impaired People

Marcelo Cabral Ghilardi
PUCRS, Faculdade de Informática

Julio C.S. Jacques Junior
PUCRS, Faculdade de Informática, Universitat de Barcelona, and Universitat Autònoma de Barcelona

Isabel Harb Manssour
PUCRS, Faculdade de Informática

We propose a model for crosswalk detection and localization by using satellite images captured from Google Maps, for the purpose of assisting visually impaired people. The detection is performed by an SVM classifier, which is combined with Google Road Map to speed up computation time and to eliminate some possible false alarms. Experimental results indicate that the proposed model works well in low resolution images, effectively detecting and localizing crosswalks in simulated scenarios.

Automatic localization of crosswalks (also called pedestrian crossing or zebra crossing) is a very important topic of research, with a wide range of applications. One possible application to take into consideration is to assist the mobility of visually impaired people. According to data from the World Health Organization, available at www.who.int/mediacentre/factsheets/fs282/en, there are approximately 285 million visually impaired people in the world. Among them, 246 million have low vision (less than 30%), and 39 million are blind. Blind tracks, white canes, and guide dogs are typically used to help visually impaired people to walk outside.[1] However, even with these resources, mobility autonomy in outdoor environments presents a major challenge for them.

Nowadays, as mentioned in the work of Ahmetovic and colleagues,[2] mobile devices have shown a huge potential in supporting people with disabilities. In addition, since these devices are becoming more accessible, a wide number of assistive technologies have been proposed to support people in everyday activities. Following this trend, several solutions for crosswalks' detection

and localization are based on images acquired by smartphones, which may bring some challenges, such as the need to correctly point the camera to the crosswalk[2-4] or the need to turn on a sidewalk to acquire panorama images.[1,5] Besides that, one can take advantage of microphones and other sensors to perform automatic image acquisition,[6] despite being limited to very specific scenarios/conditions (for instance, considering that there are buzzers to indicate go and stop status near traffic lights).

Another possible application for using smartphones, without using their own cameras for image acquisition, is to use their GPS (embedded in many modern smartphones—some of them with good localization accuracy) to initialize the extraction of satellite images (for example, from Google Maps, as in "Path Planning Algorithm Based on Search Algorithm, Edge Detector and GPS Data/Satellite Image for Outdoor Mobile Systems"[7]), applied to crosswalk detection and localization problems. To prevent any misunderstandings, in this article, *crosswalk localization* is considered the relation between the detected crosswalk to the user, whilst *crosswalk detection* is related to its identification/recognition in image coordinates. Such possibility eliminates the need to worry about which direction users must point out their camera/smartphone.

To this end, the main goal of this work is to present a computer vision based model for crosswalk detection and localization, designed to provide spatial guidance to visually impaired pedestrians. In a nutshell, the proposed model can provide information about the presence (or absence) and location of crosswalks near the user. The model is initialized by a GPS coordinate (given by a smartphone, for example), which is used to initialize the image acquisition step. Then, low resolution satellite and road map images are extracted from a public dataset (that is, from Google Maps) and used in the crosswalk's detection module. The detection is performed by a Support Vector Machine (SVM) framework. Finally, the position of the detected crosswalk is combined with the GPS coordinate to determine its localization in relation to the user, as well as to guide the user to the nearest crosswalk. However, we do not expect to guide the user to the nearest crosswalk with a centimeter precision (due to several factors, such as image acquisition problems, mapping from image coordinate to world coordinate systems, GPS accuracy, and so on), but give him/her an estimate about the nearest crosswalk. This could benefit his/her locomotion in many ways, for instance, showing that the nearest crosswalk is located on their right, at approximately 10 meters.

As stated above, the proposed model is limited to GPS accuracy. As we will address in the next section, some commercial applications (developed to help visually impaired people and based on GPS) also reported such limitation, while emphasizing that GPS can provide good accuracy most of the time. We believe that such technological limitation could be smoothed with dedicated hardware (making the solution more expensive), or even with the natural evolution of GPS technology. In addition, there are several models in the literature that use smartphone cameras to help visual impaired people (described in Related Works), that could be combined with the proposed model to fine-tune our final estimate, for instance, confirming the existence of a crosswalk, or even using our model as start point (since their initialization can be considered a crucial point, as explained next). Despite the GPS accuracy limitation, the proposed model was quantitatively evaluated (see the Experimental Results section) in three stages, as follows: first, we evaluated the accuracy of the crosswalk detection module using a dataset of crosswalk and non-crosswalk patches; during the second stage, the crosswalk detection was evaluated using satellite images extracted from Google Maps and simulated GPS coordinates; finally, the crosswalk localization was evaluated using the output from the previous evaluation. In these three evaluation scenarios, the proposed model demonstrated satisfactory results, effectively detecting and localizing crosswalks in different situations and conditions.

The main contributions of the proposed model are: (1) as far as we know there is no competitive approach that combines low resolution satellite images with a GPS system to help visually impaired people; (2) crosswalks are detected with high accuracy rates (about 96.9%), requiring low computation time (about 497 milliseconds per image); (3) crosswalks are localized with minor user intervention, with high accuracy rates (about 92.7%).

RELATED WORK

In recent years, several works have been done regarding pedestrian crossing detection and localization to aid visually impaired people to cross the street independently and in a safer way. Many of these works were developed for images captured on the ground (that is, by a smartphone, robot or vehicle).[1-6] As mentioned in the work of Senlet and Elgammal,[8] road, building, and vegetation's detection from aerial images has been extensively studied in remote sensing. However, the use of satellite images combined with GPS data (as in "Path Planning Algorithm Based on Search Algorithm, Edge Detector and GPS Data/Satellite Image for Outdoor Mobile Systems"[7]) seems to be little explored in this context.

Murali and Coughlan[1] presented a model to provide guidance to blind and visually impaired travelers at traffic intersections. In order to estimate the user location in relation to crosswalks in current traffic intersections, a user needs to acquire a panorama image (turning in place on a sidewalk) through an Android application. Given the panorama image, the model reconstructs the aerial view (overhead) of the intersection, creating a traffic intersection's template centered on the location where the user is standing. The generated template is matched against manually segmented traffic intersection regions (built from Google Maps satellite images) to determine the user's current location in the intersection. While in this approach, the user needs to worry about acquiring the panorama image; we propose automatic image acquisition and processing.

Shangguan and colleagues[6] developed a new smartphone application that locates zebra patterns and guides the user along the zebra crossing while crossing the road. In this work, they used the smartphone's microphone to detect when the user is approaching the crossroad, considering that there are buzzers to indicate go and stop status near traffic lights. To avoid user intervention, the smartphone was coupled to the user's white cane, and it takes pictures oriented by short stable periods during each white cane swing cycle (tip tends to rest on the ground for a while before swinging backwards). The crosswalk detection and localization is estimated by grouping parallel lines, which are detected through Hough transformation. One drawback of such approach is that it depends on buzzers that are not always available, especially in developing countries.

Ahmetovic and colleagues[4] presented ZebraRecognizer software library and ZebraLocalizer application. ZebraRecognizer library is responsible for the image processing module, zebra crossing identification, as well as computing the relative position between the observer and the zebra crossing. In this work, crosswalks are detected through a line segment detection and grouping algorithm, combined with a rectification matrix computation method (used to compensate for projections' distortions of extracted features). ZebraLocalizer application acquires images from the camera, sends them to the software library, and implements the interaction paradigm that enables blind users to identify crosswalks, aligned to the best crossing position and safety on the road. Data acquired from smartphone cameras and accelerometers is combined to reach the proposed solution. Their proposed image processing module was improved in "ZebraRecognizer: Efficient and Precise Localization of Pedestrian Crossings,"[2] in terms of performance and accuracy, as well as to cope with different users' height. One limitation of this approach is that the user is responsible for image acquisition through the smartphone.

Regarding methodologies in which satellite images are used, Herumurti and Uemura[9] presented a model for urban road network extraction and zebra crossing detection from very high resolution aerial images, combined with Digital Surface Models (DSM, which gives the elevation of land surface[10]). In this work, the zebra crossing is first detected by a template matching technique and then refined using SURF descriptor/object detector.[11] The output of their zebra crossing detection is used to initialize the urban road extraction module. It is important to mention that the authors did not give details about how the region of interest, associated to the image under analysis, is obtained. This model differs from ours because it is for road network extraction in urban areas using high resolution images (12 megapixels), while we locate crosswalks from low resolution images (0.4 megapixels).

Zidek and Rigasova[7] proposed to combine a search algorithm with satellite image data and edge detection for trajectory planning. They use GPS data to select starting and finishing points and to acquire the actual position of the device in relation to satellite maps. The context of this work is not related to crosswalk detection/localization, but it is mentioned to motivate the use of such combination: GPS data used to initialize satellite image acquisition.

In the work of Senlet and Elgammal,[8] a framework to construct sidewalk and crosswalk maps from satellite images is proposed. They used a crosswalk detection algorithm to complete the connectivity of a previously estimated sidewalk map. The crosswalk is detected by a template matching scheme, using synthetic templates with different frequencies and angle orientations, followed by a global segmentation thresholding and region labeling. In order to increase computation performance, as well as to decrease possible false alarms, the detection is performed on the road or near road regions. The authors mentioned that their model achieved about 83% of precision and 80% of recall to path segmentation, but they did not report any evaluation on crosswalk segmentation.

In addition, some commercial applications have been proposed for the purpose of helping visually impaired people, such as Nearby Explorer (US$99.00), Intersection Explorer, GetThere GPS, and Sendero GPS LookAround, which are free for download. Nearby Explorer enables independent and informative trips for visually impaired people. It not only shows surroundings and approaching streets and businesses, but also offers continually updating distance and directional information of the nearest or selected location. They mentioned that GPS provides good accuracy within a few yards from a person's actual position under optimal conditions. It means that it can be expected to achieve accuracy (most of the time) good enough to determine on which side of the street the traveler is. Intersection Explorer, offered by Google and maintained by Eyes-free group, speaks the layout of streets and intersections in neighborhoods as the user touches and drags his/her finger around the map. This helps visually impaired people to understand a neighborhood before venturing out and while on the go. In this app, as mentioned by the authors, the best accuracy achieved from the GPS was around 3 meters. Android app GetThere is offered by Lew Lasher, and it was designed specifically for blind and visually impaired people. It performs several spoken interactions with the user, providing various types of navigational information, such as the intersection that he/she will come across as he/she is walking and the distance to the next cross-street. GetThere offers navigation in most countries in the world, based on coverage from OpenStreetMap. They mentioned that the GPS reception is usually accurate enough to determine which side of the street the user is. Sendero GPS LookAround, developed for iPhone, announces the current street (with Voice Over), city, cross-street, heading and nearby points of interest. The highest quality data mentioned by the authors is in North America and Europe, but it is also available in other countries.

The aforementioned approaches have different solutions for the problem of crosswalk detection and localization: some of them require the user to take photos of the environment,[1] which can be considered a great challenge for visually impaired people; others need very high resolution aerial images[9] or are based on buzzers to indicate go and stop status[6] in traffic lights, which are usually not available in developing countries. Moreover, it is also desirable to locate crosswalks that are in the middle of the block and not on street corners. The mentioned commercial apps can provide spatial guidance for the user, combining GPS and spatial information from public maps, but the main difference when compared to the proposed model is that none of them propose to detect and locate crosswalks near the user. Furthermore, one of the limitations of these apps is that information returned to the user, such as businesses, banks, surrounding and approaching streets, must be previously set in a dataset, and even though they can identify the distance to the next cross-street, they do not identify crosswalks.

In the proposed model, low resolution satellite and road map images are combined with GPS coordinates for crosswalk detection and localization. The model does not depend on buzzers, the user does not need to take photos, and the location of crosswalks does not need to be previously set in a dataset. In addition, as reported in the work of Ahmetovic and colleagues,[2] although a universally accepted definition of crosswalk does not exist, many of currently adopted crosswalk standards are very similar. Considering that, the proposed approach could be easily adapted to other standards.

THE MODEL

This section describes in detail the proposed model for crosswalk detection and localization. It can be divided in three main parts: (1) an Android app to get GPS information and give voice

feedback; (2) CrossLib, related to the kernel of the proposed model; and (3) a web service that provides exchange of messages between the app and CrossLib. An overview of the proposed approach is illustrated in Figure 1.

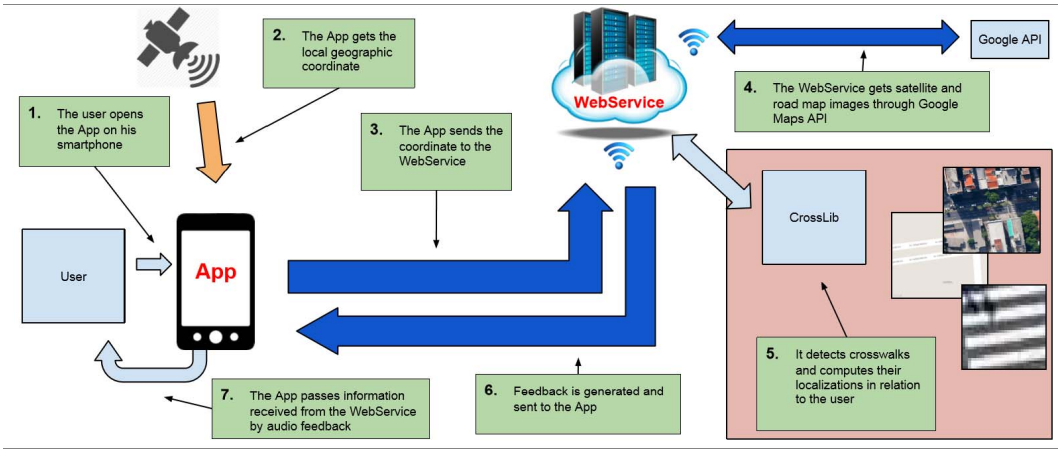


Figure 1. Overview of the proposed model.

One of the first steps of the proposed model is related to GPS coordinate acquisition. GPS coordinates are used to initialize the image extraction procedure and to make a relation between users and detected crosswalks (that is, crosswalk localization). The geographic coordinates, captured by the user's smartphone, is sent to a web service. The web service is responsible for receiving the geographic coordinates from the user (illustrated in Figure 2a by a red dot) and invokes CrossLib to initialize image extraction. To this end, Google Maps API is used to get the satellite image (Figure 2b) and the respective road map (Figure 2c). These images will be used by CrossLib during the crosswalk detection procedure (described later). The output of the crosswalk detection stage is a list of possible candidates. Such list is sent to the crosswalk localization module (described later), which gives to the user, by audio feedback, the nearest detected crosswalk location (or some other options, as described later). Next, the main steps related to CrossLib are detailed.

Figure 2. Images acquisition: (a) GPS coordinates sent by the user, illustrated by the red dot; (b) extracted satellite image from Google Maps; (c) extracted road map image from the same dataset.

Image processing module

The image processing module is responsible for extracting images from Google Maps, given a reference coordinate (illustrated in Figure 2a by a red dot), as well as guiding crosswalk detection. As we are using a free version of Google Maps API, each obtained image (satellite and road

images) is a matrix with 640×640 size of resolution, captured by a zoom of 20× (that is, the image height from the ground), corresponding to an area of 6.890m^2, approximately.

The first step of the image processing module (after image acquisition) relates to image segmentation, which consists of defining a region of interest in which crosswalks could be included (on the road or near road regions). The purpose of such approach is to improve computational performance as well as to eliminate the possibility of finding false positives in unrelated areas (for instance, tops of buildings, roofs, and so on).

Initially, both the input satellite image and road image (assigned to R) are converted to grayscale (illustrated in Figure 3a). The region of interest B is generated by a simple thresholding approach. The segmentation is obtained by thresholding each pixel (x,y) of R by λ (where $\lambda = 245$, set experimentally), as defined in Equation 1. Figure 3b illustrates the output of the thresholding approach.

$$B(x,y) = \left\{ \begin{matrix} 0, if R(x,y) \le \lambda \\ 1, otherwise \end{matrix} \right\}. \qquad (1)$$

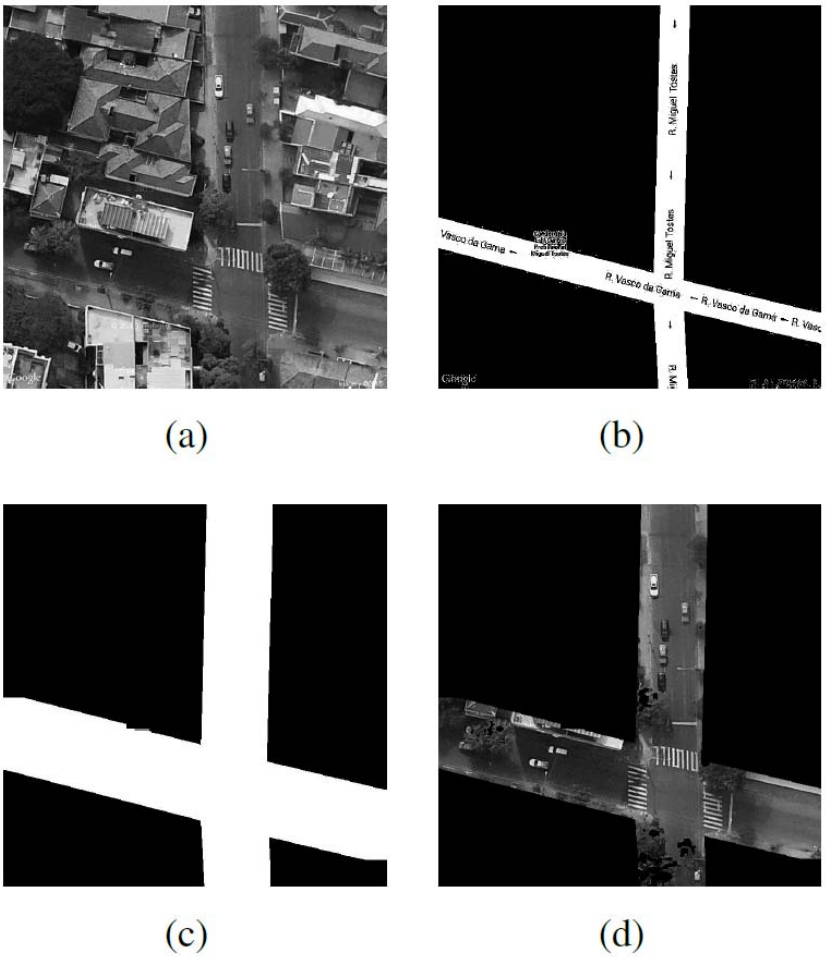


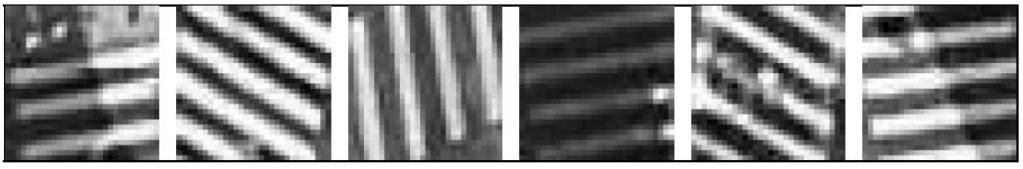
Figure 3. Image segmentation: (a) input satellite image (illustrated in Figure 2b) converted to grayscale; (b) initial segmentation of the road map (shown in Figure 2c); (c) output of morphological operations; (d) combining the satellite image with the region of interest.

As we can see in Figure 3b, there are several undesirable structures in the binary image (text, arrows, and so forth). By trying to eliminate such undesirable structures, a post-processing morphological operation was applied to this binary image (a combination of erosions and dilations). More precisely, we first applied 2 erosions in image B, with a squared structuring element with size 3×3. The resulted (eroded) image was then dilated 20 times, using the same structuring element. The goal of such procedure was to remove small artifacts found in the binary image and extend the area of interest to deal with misalignment of the road map (regarding the satellite image). The order, number, and type of morphological operations were set experimentally (these choices are strongly related to the adopted image resolution and zoom, and they can be easily changed to deal with different setups).

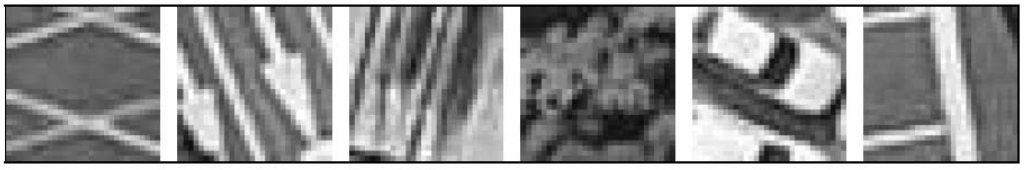
The output of the morphological operations, illustrated in Figure 3c, was then combined with the respective grayscale satellite image (Figure 3d). The procedure described above was used in the next stages of our model for crosswalk detection, as described next.

SVM classifier for crosswalk detection

In this work, we propose to train a Support Vector Machine (SVM) classifier to detect crosswalks from satellite images. To this end, we built a dataset containing several positive and negative sample patches (with size 30×30 pixels in each patch), used for cross-validation. This dataset is composed of 370 patches of crosswalks (positive samples) and 530 patches of non-crosswalks (negative samples), illustrated in Figure 4, which were manually extracted from Google Maps using its API (considering grayscale input images with 640×640 of resolution, captured by a zoom of 20×). As we can see in Figure 4, image patches vary according to illumination conditions (for example, captured at different times of the day, with or without shadows), and angle orientation and shapes, regarding the zebra pattern. In addition, some positive image patches are partially occluded by people, cars, and/or trees, while some negative image patches contain structured objects with parallel lines, such as directional arrows. The idea of including such features in the learning dataset was to deal with different situations/scenarios we usually find in real applications. Next, the dataset was divided into training and test sets, with 600 and 300 independent patches, respectively, on which a 10-fold cross-validation was used to optimize (hyper)parameters of SVM (described later).



(a)



(b)

Figure 4. Manually extracted patches of (a) positive and (b) negative samples.

We used the Local Binary Pattern (LBP) feature extraction method to train our SVM. As reported in the work of Dixit and Hegde,[12] the LBP texture operator was introduced as a complementary measure for local image contrast. It can be used to recognize a wide variety of texture types, in which statistical and structural methods have conventionally been used separately. We

have also tried to use the well-known Gray Level Co-occurrence Matrix (GLCM)[12] feature extraction, and a combination of both. Table 1 shows the evaluated methods for feature extraction under development of the proposed model.

Table 1. EVALUATED METHODS FOR FEATURE EXTRACTION.

Feature Extraction Method	Descriptor
Gray Level Co-occurrence Matrix	Contrast Energy Entropy Homogeneity
Local Binary Pattern	Black & White Symmetry Geometric Symmetry Degree of Direction

Crosswalk detection method

After training the SVM classifier, the next stage of the proposed model was crosswalk detection. In a practical situation, we consider an input satellite image (640×640 size of resolution), delimited by the region of interest, as illustrated in Figure 3d. This input image is divided into small cells, with size 15×15, as shown in Figure 5a (illustrated by a cropped region, highlighted in Figure 5d, for visualization purposes).

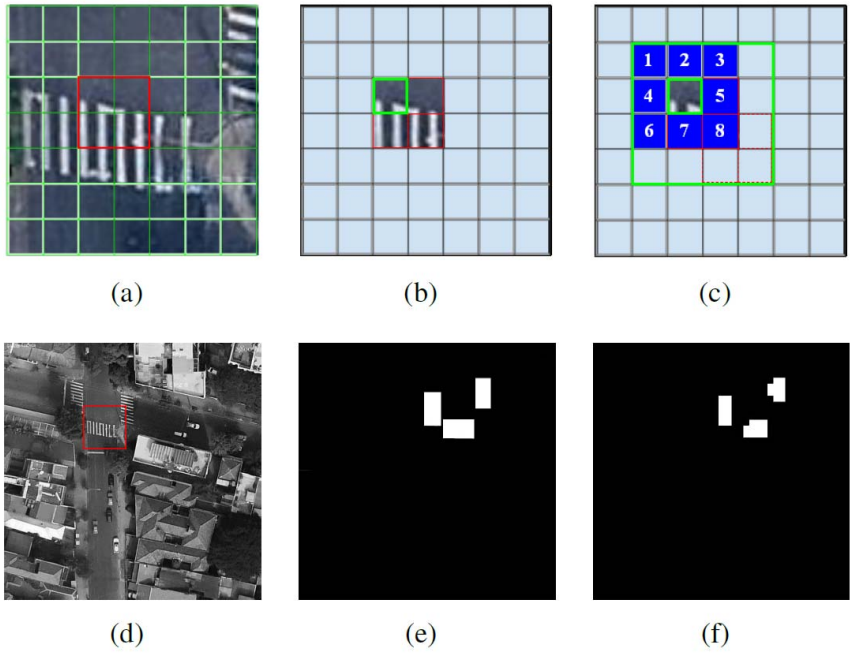


Figure 5. Crosswalk detection illustration: (a) cropped region of an image under analysis (Figure 5d) with a 30×30 reference patch marked by a red square; (b) reference patch and its 15×15 reference cell highlighted by a green square; (c) reference cells of eight patches used in the second verification; (d) input satellite image (with cropped region illustrated in the image (a) highlighted by a red square); (e) ground truth manually informed by the user; (f) final estimation, given by the proposed model.

During the classification stage, all of these 15×15 cells, which fall in the region of interest, are visited. Then, each group of four neighboring cells (that satisfies region of interest's criteria) will form a patch with size 30×30 (related to the same patch size used to train our SVM classifier), illustrated in Figure 5a by a red square. Each patch is composed by the reference cell (the upper left cell, illustrated in Figure 5b by a green square) and its neighboring cells (that is, the right cell, the bottom one, and the bottom right one). The goal of such procedure was to create an overlapping region between adjacent patches to deal with misalignment in the classification stage, as well as to perform a two-pass verification, as described next.

If a 30×30 reference patch is considered a crosswalk (for instance, the patch generated from the reference cell illustrated in Figure 5b), a second verification is performed. In this second verification, eight neighboring patches (30×30) are created around the reference patch, considering their neighboring cells (the reference cells of these eight patches are illustrated in Figure 5c by blue squares). If at least one of these eight neighboring patches is also considered a crosswalk, the reference patch is then set as a crosswalk; otherwise the reference patch is discarded.

Figure 5e illustrates regions of crosswalks manually informed by a user (used as ground truth), whilst Figure 5f illustrates the overall output for a given image (Figure 5d), in which patches are detected as crosswalk (by our two-pass verification), illustrated by white blocks.

Crosswalk localization method

Given detected crosswalks, the next stage of the proposed model relates to making a spatial relation between the user and crosswalks (that is, crosswalk localization). First, we assume the user would always send his/her GPS coordinate when facing a street and, by doing so, we can easily estimate which street he/she is by analyzing the binary image (Figure 3c) and his/her informed position (illustrated in Figure 6 by a red dot). If the user is at a corner, we assume that he/she is facing toward the corner.

Following the estimated street line in both directions (left and right), as shown in Figure 6, and also after analyzing the binary image, we can find corner intersections when facing another street (if that exists), and consequently, define the block where the user is positioned. Such information is used to discard detected crosswalks that are not connected to the block where the user is. Corner detection could also be used to give the user additional feedback (for example, when crosswalks are not detected in a block, the model could inform the user about his/her location in relation to the nearest corner).

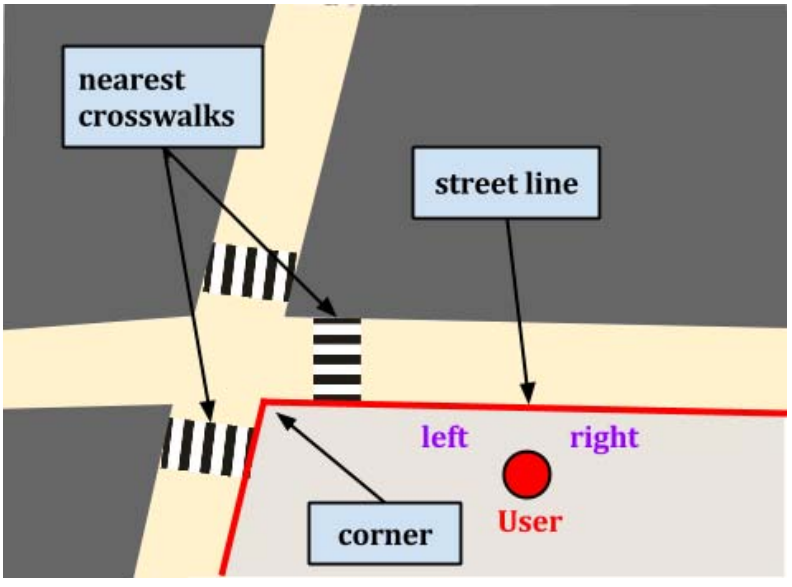


Figure 6. Illustration of street lines, corners, blocks, and crosswalks connected to the block in which the user is. All information is used in the crosswalk localization module.

Our model is developed to inform the user about the nearest crosswalk (or the two nearest in specific situations—if they are approximately at the same distance from the user). The idea is to avoid giving too much information that could confuse the user, but at the same time give him/her choices when facing two crosswalks at the same distance. To locate the nearest crosswalk (or the two nearest ones), two different situations are considered: i) the user is positioned at the corner, and ii) the user is facing the street (for example, in the middle of the block). The first case is defined just to create a reference point (that is, the corner point, or the exact street intersection point), used as a reference when informing the user the crosswalk localization (for example, there is a crosswalk on your left).

Thus, if two or more crosswalks are detected, we first compute the distance from the user to each of them, following the street line edges (to take into consideration the corners). The distance from the user to the nearest crosswalk (or corner) is computed using a very simple relation, which takes into consideration the image resolution and its real dimension, extracted from Google Maps API. Since we know that each image has 640-pixel width, related to 82.69 meters, we can assume each pixel represents 0.129203125 meters. In case two or more crosswalks have approximately the same distance in relation to the user, he/she is informed about the two nearest ones. We defined in our experiments that two crosswalks have similar distances in relation to the user when their distances differ in less than 3 meters (being a little flexible, regarding GPS accuracy). This procedure is described in Algorithm 1. Such information is used to compute the final feedback, which is sent to the user, as explained next.

```
1: N_c is the number of detected crosswalks in a block
2: U_p is the user position
3: if N_c = 1 then
4:     inform the user its location
5: end if
6: if N_c > 1 then
7:     Compute the distance from the user to each of them
8:     if they have different distances then
9:             inform the user the nearest crosswalk
10:    else
11:            inform the user the two nearest crosswalks
12:    end if
13: end if
```

Algorithm 1. Crosswalk localization.

App feedback

The proposed app provides voice feedback to the user. In order to simplify feedback, we consider the user can be facing the corner or the street, as previously mentioned. In this way, regardless of the user's position in relation to cardinal directions, feedback is provided in relation to his/her right and/or left. The provided feedback informs the distance to the nearest crosswalk (or crosswalks) on the block the user is positioned (considering only the street in front of the user and those that intersect it, which means that if the whole block is visible in the satellite image, we can discard crosswalks located on streets far away from the user). Figure 7 exemplifies the points that can generate feedback, and Table 2 describes possible feedback. The values presented in Table 2, that is, "10 meters" and "5 meters," are used for illustration purposes, since computed distances (in meters) may vary according to each situation. Information about "turning the corner" is provided only when the user is facing the street line (that is, in the middle of the block), and the nearest crosswalk is positioned after turning the corner.

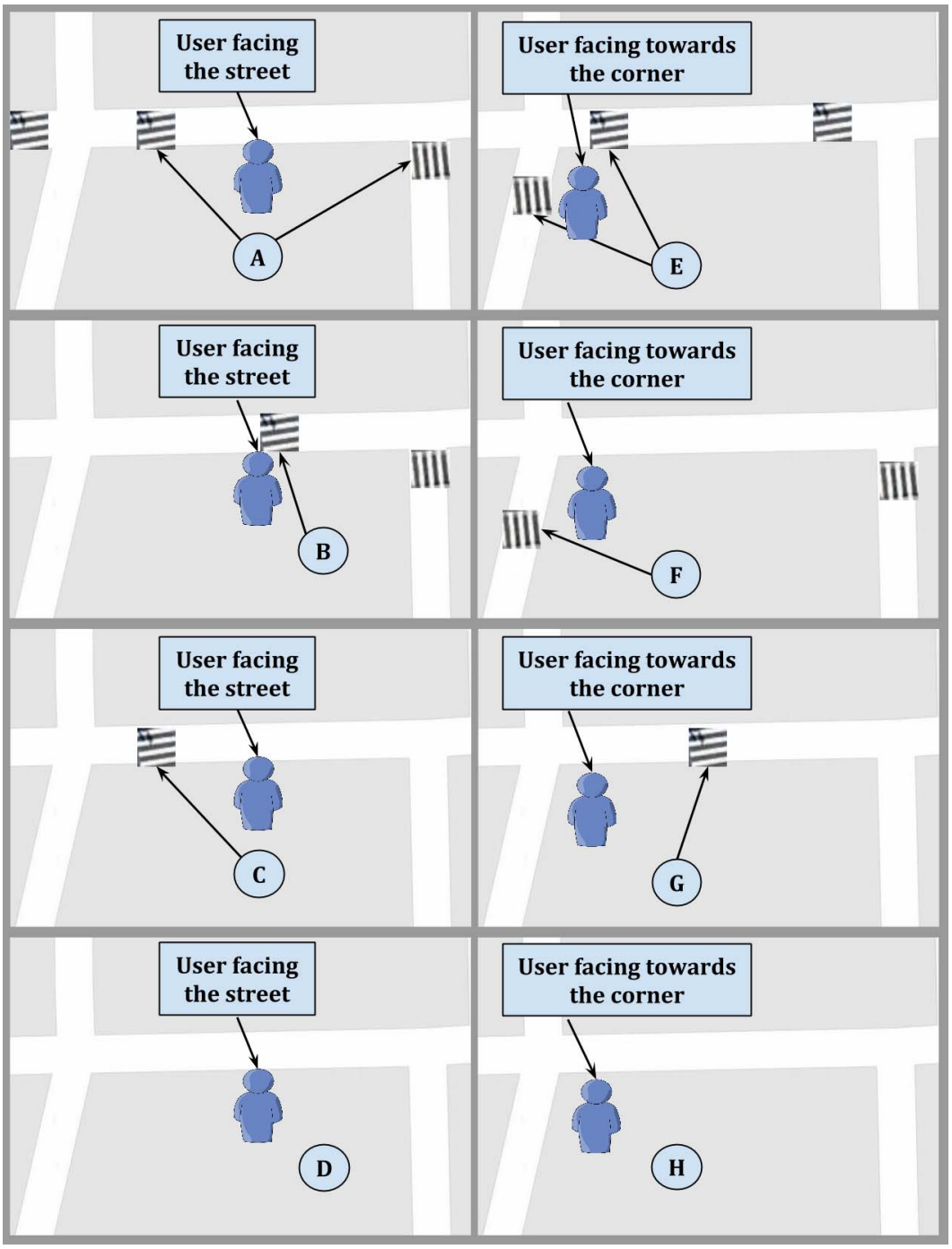


Figure 7. Examples of possible feedback.

Table 2. FEEDBACK EXAMPLES, ILLUSTRATED IN FIGURE 7.

A	There are two crosswalks at 10 meters, one on your left and another on your right, just after turning the corner.
B	There is a crosswalk in front of you.
C	There is a crosswalk at 10 meters on your left.
D	There is no crosswalk in the block you are.
E	You are at the corner and there are two crosswalks at 4 meters, one on your left and another on your right.
F	You are at the corner and there is a crosswalk at 5 meters on your left.
G	You are at the corner and there is a crosswalk at 5 meters on your right.
H	You are at the corner and there is no crosswalk in the block you are.

EXPERIMENTAL RESULTS

In this section, we illustrate some experimental results obtained by the proposed model. The experiments are divided in three case studies, as follows: i) feature evaluation, used to choose the best feature extraction method for texture analysis (regarding the evaluated ones); ii) crosswalk detection, used to evaluate the accuracy of the SVM classifier; and iii) crosswalk localization, which is used to evaluate accuracy of the application feedback.

Feature evaluation

This section describes how LBP was chosen as feature extraction method. Different methods were evaluated (as previously described in Table 2), as well as some combinations of them. The experimental results are reported below. First, our database of image patches, previously described (containing a total of 900 image patches), was randomly divided in a stratified way into a training set (containing 600 samples) and an independent test set (with 300 samples). The training set was used in a 10-fold cross-validation to assess the performance of SVM for different hyper-parameters. Once the optimal hyper-parameters were found, a new model was trained using the complete training set and applied to the independent test set. The evaluation was performed in terms of sensitivity, specificity, and accuracy, defined in Equations 2, 3 and 4, respectively.

$$Sensitivity = \frac{TP}{TP + FN'}, \qquad (2)$$

$$Specificity = \frac{TN}{TN + FP'}, \qquad (3)$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN'}, \qquad (4)$$

where TP, TN, FP, and FN refer to True Positive, True Negative, False Positive, and False Negative, respectively.

Considering the Gray Level Co-occurrence Matrix (GLCM) feature extraction method,[12] three variables must be considered: direction and distance from neighboring pixel and number of grayscale tones. Since crosswalks can assume different directions in satellite images, we carried out the experiment using four directions (0, 45, 90, and 135 degrees), ranging the distance parameter from 1 to 4, with grayscale tones from 2^1 to 2^7. As illustrated in Figure 8, the best result related to GLCM feature extraction (90.3% accuracy) was obtained when using distance=1 and grayscale tones=2.

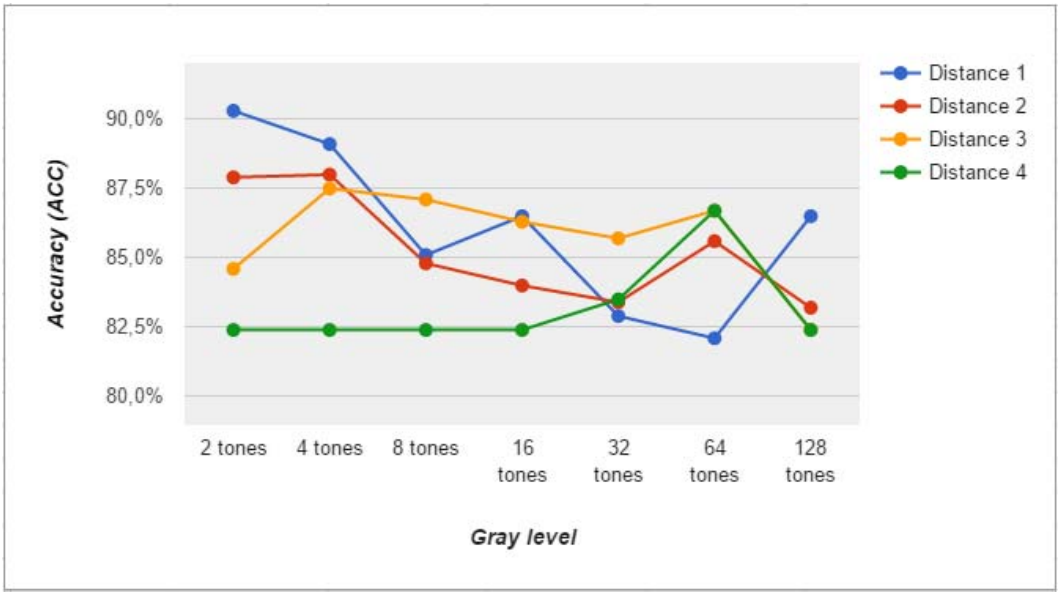


Figure 8. Accuracy evaluation of Gray Level Co-occurrence Matrix (GLCM) method.

On the other hand, as we can see in Table 3, the overall best result was obtained by LBP method, reaching about 94.6% of accuracy in crosswalk detection. In LBP method, a neighborhood of 3×3 pixels was considered, as well as the following three measures: Black & White Symmetry (BWS), which measures the symmetry between the left half and right half of grayscale's histogram; Geometric Symmetry (GS), which measures the regularity of texture form; and Degree of Direction (DD), which measures the degree of texture linearity. The other measures usually used by LBP method were not considered because they are related to texture direction extraction (and crosswalks can assume different orientations).

Table 3. COMPARISON AMONG FEATURE EXTRACTION METHODS.

Features	Sensitivity	Specificity	Accuracy
LBP	95.7%	93.9%	94.6%
GLCM + LBP	86.8%	93.6%	90.9%
GLCM	89.0%	91.1%	90.3%

Crosswalk detection evaluation

In this section, we describe the evaluation of the proposed crosswalk detection method in a practical situation. In this case study, several satellite images (to be more specific, 100 images) were extracted from Google Maps (considering grayscale input images with 640×640 of resolution, captured by a zoom of 20×), taking into account the input GPS coordinate given by the user. To simulate the GPS coordinate informed by the user, we randomly selected N (where N = 100) coordinate positions using Google Maps API (all coordinates were extracted from locations close to streets, simulating previously mentioned conditions that the user is facing the street or a corner). For each extracted satellite image, a ground truth was manually generated by the user. The ground truth (Figure 9b) of each satellite image (Figure 9a) is defined by a binary image (with the same size) with crosswalk regions delimited by almost rectangular boxes. Such information is used for quantitative evaluation. Figure 9c illustrates the output binary image of the proposed model for a given situation, and thus we can verify how accurate the model is.

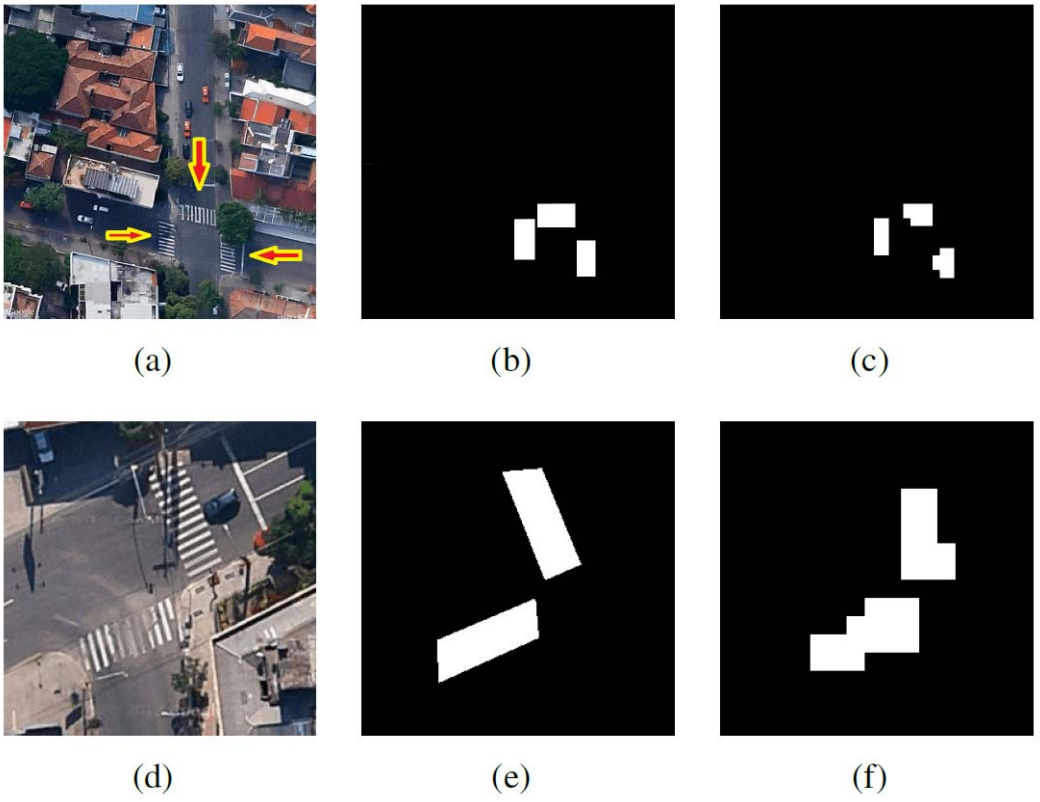


Figure 9. Illustration of extracted satellite image in (a) and a badly painted case (zoomed for visualization purposes) in (d) with their ground truth data in (b) and (e), respectively. Images (c) and (f) illustrate the respective output binary images of the proposed model for crosswalk detection. The lower row shows a case in which the crosswalk in a bad painting condition brought some additional challenges to the crosswalk detection module.

It is important to mention that, in this experiment, images were extracted from some state capitals of Brazil, such as Saõ Paulo (SP), Florianopolis (SC) , and Porto Alegre (RS). This choice brings some challenges. For instance, as Brazil is considered a developing country, the quality of some crosswalks can present some variations (as visually observed) regarding their painting quality (for instance, smoothed or with minor lacks); Figure 9d illustrates such situation. In addition, the proposed model is not developed to work only in such cities of Brazil, as the zebra crossing pattern is adopted in many cities around the world. However, we noticed that there will be variation regarding crosswalk patterns depending on where we are, affecting the performance of the proposed model. Figure 10 illustrates such problem, caused by different kinds of patterns found around the world. We argue that such problem could be easily addressed through different training sets.

Figure 10. Example of different crosswalk patterns, which could affect the classifier's performance, extracted from different cities around the world: (a) Miami, USA; (b) Phoenix, USA; (c) Tokyo, Japan.

Each ground truth image (for example, Figure 9b) is confronted with the estimation given by the proposed model (for example, Figure 9c). The comparison is made in the level of patches instead of pixels, considering that ground truth images (as well as output binary ones) are discretized by patches with 30×30 of size. We defined a True Positive patch when a ground truth patch (marked as a crosswalk region) and the output binary image patch at same location (estimated as a crosswalk region) have more than 10% of intersection. A False Positive patch is defined when the output binary image patch fails in a non-crosswalk region of the ground truth patch more than 10% of its area. True Negative and False Negative are defined similarly.

In this experiment, as shown in Table 4, the proposed model achieved an average accuracy of 96.9%, with standard deviation of 2.841, which we consider as a satisfactory accuracy rate. The average computational cost to process each image was 497 milliseconds, with standard deviation of 244, using an Intel i5 Processor, 2.27GHz and 4Gb of memory (implemented using C#.NET framework 4.5 and AForge.NET framework).

Table 4. CROSSWALK DETECTION EVALUATION.

Sensitivity	Specificity	Accuracy
87.5%	97.8%	96.9%

It is important to mention that we are not proposing a completely novel method, since SVM and LBP feature extraction are broadly used as solutions for many classification problem. However, the proposed model achieved satisfactory performance considering specificity and accuracy values, and the novelty is based on the usability compared to existing systems, since the user does not have to worry about acquiring images. In addition, the processing was carried out automatically and with low resolution images. Because of that, it was not possible to do a direct performance evaluation comparison between our approach and other models and apps.

Crosswalk localization evaluation

In this section, we evaluate the accuracy of the provided feedback related to the crosswalk localization module. To do so, we randomly chose a set of 100 geographical coordinates (simulating users' input), representing a wide number of situations, that is, areas with a crosswalk in front of the "user," on his/her right/left, as well as areas without crosswalks and areas with badly painted crosswalks or with partial occlusions. In this experiment, we considered that coordinates were always obtained from a sidewalk region, in which 75% of them were extracted simulating the user positioned facing the street line, whereas the other 25% simulated the user positioned at the corner.

In a second stage, we created a ground truth data associated to expected feedback for each extracted image (by visual inspection and according to Table 2, for example, "there is a crosswalk at 7 meters on your right"). In this case, we considered feedback as positive (true) when the model returned the corresponding feedback, and negative (false) otherwise. It is important to emphasize that the relative distance, measured from the user to the nearest crosswalk, was alleviated in this experiment (it means that small variations were allowed within 3m, if corresponding directions match). In addition, if there were two crosswalks at the same distance, and the provided feedback returns that there is only one crosswalk, this was considered a false result.

As shown in Table 5, the proposed model provided the expected feedback in 92.7% of the simulated cases with an average specificity of 95%. We consider these results very promising, since they include proper localization of corners and crosswalks. Lower accuracy values were observed in situations in which crosswalks were badly painted or under partial occlusions.

Table 5. CROSSWALK LOCALIZATION EVALUATION.

Sensitivity	Specificity	Accuracy
91.5%	95.0%	92.7%

CONCLUSION

In this work, we proposed a model for crosswalk detection and localization using satellite and road map images captured from low resolution public datasets (that is, Google Maps). The image extraction was initialized by GPS coordinates, which can be sent by the user through his/her smartphone. Crosswalks were detected by an SVM classifier and proper feedback was sent to the user based on predefined rules. The proposed model required low computational cost and had minor user intervention.

The proposed model was quantitatively evaluated (regarding crosswalks' detection and localization procedures), using a dataset containing satellite images extracted from Google Maps and simulated GPS coordinates. Experimental results indicated that the model effectively detected crosswalks in 96.9% of cases, achieving about 92.7% of accuracy in relation to their localization. We believe the proposed model can be used to improve the quality of life of visually impaired people, helping them to get around in outdoor environments in an easier and safer way.

For future work, we intend to use other resources available on most smartphones, such as the compass and/or accelerometer, in order to increase crosswalk localization accuracy, as well as to develop a case study with visually impaired people to evaluate the real applicability of the proposed model.

ACKNOWLEDGMENTS

The authors would like to thank Brazilian agencies FAPERGS/CAPES, Hewlett-Packard Brasil Ltda. and PUCRS for the financial support.

REFERENCES

1. V. Murali and J. Coughlan, "Smartphone-based crosswalk detection and localization for visually impaired pedestrians," *2013 IEEE International Conference on Multimedia and Expo Workshops* (ICMEW), 2013, pp. 1–7.

2. D. Ahmetovic et al., “ZebraRecognizer: Efficient and Precise Localization of Pedestrian Crossings,” *22nd International Conference on Pattern Recognition* (ICPR), 2014, pp. 2566–2571.
3. X. Kou, Y. Wei, and M. Lee, “Vision based guide-dog robot system for visually impaired in urban system,” *13th International Conference on Control, Automation and Systems* (ICCAS), 2013, pp. 130–135.
4. D. Ahmetovic, C. Bernareggi, and S. Mascetti, “Zebralocalizer: Identification and localization of pedestrian crossings,” *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services* (MobileHCI), 2011, pp. 275–284.
5. G. Fusco et al., “Determining a blind pedestrian’s location and orientation at traffic intersections,” *Computers Helping People with Special Needs*, Springer, 2014.
6. L. Shangguan et al., “Crossnavi: Enabling real-time crossroad navigation for the blind with commodity phones,” *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (UbiComp), 2014, pp. 787–798.
7. K. Zidek and E. Rigasova, “Path Planning Algorithm Based on Search Algorithm, Edge Detector and GPS Data/Satellite Image for Outdoor Mobile Systems,” *IEEE 10th International Symposium on Applied Machine Intelligence and Informatics* (SAMI), 2012, pp. 349–354.
8. T. Senlet and A. Elgammal, “Segmentation of occluded sidewalks in satellite images,” *21st International Conference on Pattern Recognition* (ICPR), 2012, pp. 805–808.
9. D. Herumurti et al., “Urban road network extraction based on zebra crossing detection from a very high resolution rgb aerial image and dsm data,” *2013 International Conference on Signal-Image Technology Internet-Based Systems* (SITIS), 2013, pp. 79–84.
10. T. Toutin and L. Gray, “State-of-the-art of elevation extraction from satellite SAR data,” *Journal of Photogrammetry and Remote Sensing*, vol. 55, no. 1, 2000, pp. 13–33.
11. H. Bay et al., “Speeded-up robust features (surf),,” *Computer Vision and Image Understanding*, vol. 110, no. 3, 2008, pp. 346–359.
12. A. Dixit and N. Hegde, “Image texture analysis - survey,” *Third International Conference on Advanced Computing and Communication Technologies* (ACCT), 2013, pp. 69–76.

ABOUT THE AUTHORS

Marcelo Cabral Ghilardi is a master’s student in the School of Computer Science at the Pontifical Catholic University of Rio Grande do Sul (PUCRS), Brazil. His research interests include all areas of computer vision, computer graphics, accessibility, and human-computer interaction. Ghilardi has a graduate degree in applied technologies for information systems from University Center Ritter dos Reis. Contact him at marcelo.ghilardi@acad.pucrs.br.

Julio C.S. Jacques Junior is a postdoctoral researcher in the Department of Mathematics and Informatics at the Universitat de Barcelona, Spain. His research interests include human behavior analysis, image segmentation, crowd analysis, object detection, and tracking. Jacques Junior has a PhD in computer science from Pontifical Catholic University of Rio Grande do Sul, Brazil. Contact him at julio.jacques@acad.pucrs.br.

Isabel Harb Manssour is an associate professor in the School of Computer Science at the Pontifical Catholic University of Rio Grande do Sul (PUCRS), Brazil. Her research interests include all areas of visualization, computer graphics, human-computer interaction, and computer vision. Manssour has a PhD in computer science from the Federal University of Rio Grande do Sul. Contact her at Isabel.manssour@pucrs.br.

\