A Simple and Model-Free Algorithm for Real-Time Pedestrian Detection and Tracking

Yegor Malinovskiy
Undergraduate Research Assistant
Department of Computer Science and Engineering, and
Department of Civil and Environmental Engineering
University of Washington
Seattle, WA 98195
Tel: (206) 543-7827
Email: yegorm@u.washington.edu

Jianyang Zheng, Ph.D. Candidate (Corresponding Author)
Research Assistant
Box 352700
Department of Civil and Environmental Engineering
University of Washington
Seattle, WA 98195-2700
Tel: (206) 616-6056
Email: jianyang@u.washington.edu

Yinhai Wang, Ph.D.
Assistant Professor
Box 352700
Department of Civil and Environmental Engineering
University of Washington
Seattle, WA 98195-2700
Tel: (206) 616-2696
Fax: (206) 543-5965
Email: yinhai@u.washington.edu

Word count: 4254 + 7×250 = 6004

Submitted for Presentation at the 86th Annual Meeting of the Transportation Research Board and for Publication in Transportation Research Record: the Journal of the Transportation Research Board.

Submitted on August 1, 2006

ABSTRACT:
Pedestrian monitoring is quickly becoming an avid area of interest as information regarding pedestrian flow is needed not only for developing competent access in particular urban corridors, but also system optimization scenarios, such as transit systems and intersection control optimization. In this paper, we present a simple, yet effective method for tracking pedestrian objects in a relatively large surveillance area, using ordinary CCTV cameras. Object extraction is accomplished via background subtraction, while tracking is accomplished through an inherent characteristic cost function. Composite objects are used as a means to deal with occlusions. The algorithm is implemented in a Pedestrian Tracking (PedTrack) system using Microsoft Visual C#. Tested with both synthetic and real video data, this system was proved to be reliable and about 80% pedestrian were assigned with correct IDs.

INTRODUCTION

Pedestrian detection is an important part of Intelligent Transportation Systems (ITS) applications. Automatic pedestrian detection at signalized intersections helps signal controllers to decide whether a pedestrian crossing green should be initiated. Also if pedestrians in the process of crossing can be detected, green extension can be allocated as appropriate so that pedestrian safety can be enhanced and improve efficiency of intersection signal control can be improved. Based on studies in Los Angeles, California, Rochester, New York, and Phoenix, Arizona, automated pedestrian detection can significantly improve traffic safety *(1)*. At non-signalized intersections, automated pedestrian detection can detect pedestrians in the crossing area and activate a warning device for approaching motorists. This passive system is more reliable and convenient for pedestrians, when compared with active devices, such as the push button *(2, 3)*. Pedestrian detection can also provide pedestrian volume data, which have been identified by the Bureau of Transportation Statistics as some of the most critical data for the purposes of: tracking usage trends, developing exposure measures for crash analysis, evaluating level of service, identifying and prioritizing improvements, and calibrating travel demand models *(4)*.

Recent methods for monitoring pedestrian activity have been mostly done with either infrared or video sensors. Although the prior offers the advantage of easier differentiation between pedestrians and vehicles, it is normally more expensive. Video sensors range in their parameters, but of primary interest are low-resolution surveillance-type of video cameras, as they are widely deployed for traffic operations and can be easily utilized for automated tracking of pedestrians and vehicles.

Object tracking using low-resolution video is a difficult problem. Although numerous approaches have been attempted, there are essentially three basic means of obtaining and tracking objects in a scene: feature tracking, pattern matching, and background subtraction. Feature tracking is a method that relies on tracking areas of high gradient difference, such as corners, joints or other areas of sharp curvature. However, pedestrian objects often lack such stable features, particularly in grayscale video sequences with very low resolution. Pattern matching is a method that relies on a local library of positive or negative examples of objects to look for. This approach is highly dependent on the accuracy of the library. Self-learning algorithms exist that build up a probability based library of objects, however they are computationally expensive and heavily dependent on manual initialization. Background subtraction is the most commonly used method of obtaining objects in a scene. The scene is monitored for some time and the mode or mean value is taken at every pixel, creating a background image. This background image is then subtracted from the current scene to reveal the objects that are not part of the background and thus are in motion. The extracted objects are then tracked based on proximity and other principles. Background subtraction-based approaches have a few key issues, such as camera vibration, changing environmental conditions, as well as detection misses due to similar coloring of the object and the background. However, these are outweighed by the relative simplicity and thus universality of the algorithm, as well as relatively fast computation time.

In this paper, we propose an algorithm designed to track pedestrians in low to medium flows in daytime conditions using image sequences from low-resolution grayscale cameras. The algorithm comprises of three steps: acquisition of moving objects, object tracking, and object classification. The algorithm has been implemented in a prototype Pedestrian Tracking (PedTrack) system using Microsoft Visual C#. The system was tested using live video images from several locations and the result is encouraging.

LITERATURE REVIEW

Data collection through video detection and object tracking is quickly gaining popularity. Tracking pedestrians with video is quick, automatic and inexpensive compared to the manual alternative. Numerous methods have been developed, each with its own strengths and weaknesses.

Model-based techniques are popular, such as ones suggested in *(6)*. However, not only must the model library be competent, a robust scaling algorithm must be present in order to be able to adapt to size changes, as presented in *(7)*. While the results shown by Collins *(7)* are quite promising, the creation and upkeep of the model libraries is still a problem – manual selection of desired objects is not an option.

Model-free approaches are much more attractive in terms of upkeep and universality, yet have a harder time with object classification. Model-free object classification resorts to classifying by size and/or height-to-width ratio, as done in *(5)*. Although this does restrict the position of the camera somewhat (pedestrians have to have a different height-to-width ratio than vehicles or be smaller than cars, which is true in most surveillance angles), it is not too great of a restriction for most cases. As for groups of people that may be the size of a car, as well as of similar proportions, it will be shown that such cases can be dealt with most of the time.

Pedestrian detection and tracking typically involves three steps: acquisition of moving objects, tracking, and classification *(8, 9)*. Acquisition of moving objects can be done through background subtraction, as described in *(10)*. A good quality background image is necessary for background based detection and tracking approaches. A background image can be extracted by the mode-based approach *(10)*. Compared to the commonly used median-based approach, the mode-based approach is cheap in computation. In practice, the mean-based background extraction approach is also utilized when traffic volume is low. Although this approach is less accurate, it is much simpler because the pixel values do not have to be sorted to find the mean. As mentioned above, classification consists of simply applying a threshold to the size and height-to-width ratios of the object. However, object tracking is not as simple. As noted in *(11)*, robust occlusion reasoning is the key to any successful tracking system. The key issue in occlusion reasoning with an object extraction step of background difference is the fact that object blobs (as they appear on the binary difference image) tend to leak together when objects get close to one another, making it difficult to track the individual objects. Attempts to dissect the merged blobs have been made in *(5)*, but these do not work well with a fairly common type of occlusion – a momentary total occlusion – such as one that happens when one pedestrian walks in front of another. Thus, while a method similar to what suggested in *(5)* is used to track individual objects, a different approach should be taken for occlusion reasoning. In most cases, it is not important to know where exactly in the merged blob a particular object is. Therefore Jorge et al. *(11)* suggested to track merged blobs and called these blobs "composite" objects. A composite object may split into several objects. Research efforts have been directed to discern the individual objects once they split. Direction based approaches have been used to discern pedestrians, as in *(11)*. These approaches assume that pedestrians in the scene have an origin and a destination, traveling the shortest possible distance between them, i.e. a line. That is true in many cases, but does not work well for larger surveillance areas, where pedestrians may change direction abruptly to cross the street. Pai et al. *(12)* used a pedestrian model together with the walking rhythm of a human being to conduct pedestrian recognition and tracking. Though the experimental results were favorable, the algorithm requires high computing power and high image resolution to execute.

At the current state, no single algorithm or system has been widely accepted in pedestrian detection and tracking. However, previous studies mentioned above provide valuable insights to our study of pedestrian detection and tracking.

METHODOLOGY

The pedestrian detection and tracking algorithm proposed in this study contains three steps. The first step is acquisition of moving objects. The second step is object tracking - associating the same object in different frames. This is the most challenging part because objects may merge or split in the video stream. Therefore, occlusion reasoning will be performed if necessary. The last step is object classification; only pedestrians will be assigned ID tags. Figure 1 shows a simplified flow chart of the algorithm.

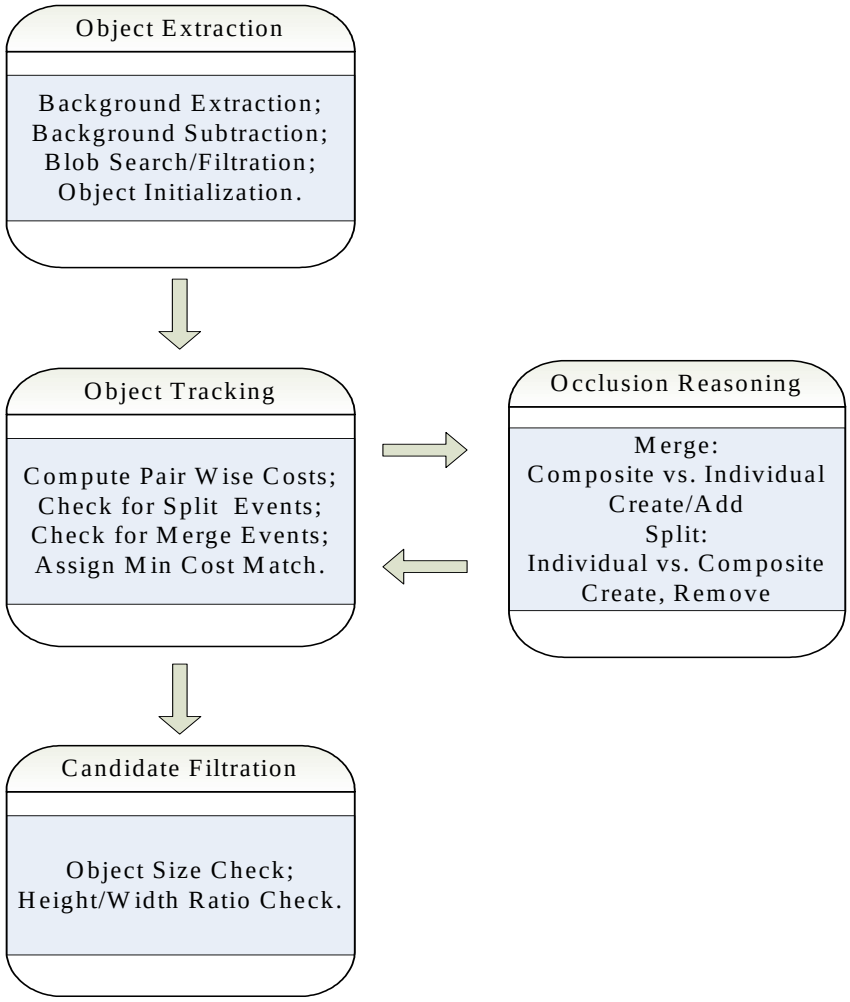


FIGURE 1 Flow Chart of the Proposed Algorithm.

Background subtraction is used to extract objects from the scene. Tracking is accomplished by matching the size, position, and a 16-bin histogram of the object of interest to one in the next frame, as done by Owens et al *(5)*. Occlusion reasoning is done by watching for merge and split

events. On a merge event, the objects merging are considered a composite object which remains until a split event happens, where the object that splits off is matched to its initial stage, the one prior to the merge event. Object filtration is then applied to distinguish pedestrians from other moving objects that may be present in the scene – usually vehicles. Pedestrians are distinguished by their size and the height-to-width ratio. Figure 2 shows an example of the proposed system's results with identified individual pedestrians.

FIGURE 2 A simple case of individual pedestrians.

The main advantages of the proposed algorithm are its simplicity, speed and robustness. Simplicity complements robustness – because the algorithm requires no libraries or manual training, it is suitable for essentially any surveillance camera placement with essentially no calibration – only one parameter has to be changed. Details of the algorithm are described step by step as follows.

Acquisition of Moving Objects

The first step of our algorithm is to identify the objects to be tracked. As mentioned before, this is accomplished through background subtraction. The background is extracted from a succession of 500 frames by calculating the mode of these frames as suggested by (*10*). For each pixel at location (x, y) in frame *i*, its color values are in vector **I**(x, y, i). In the PedTrack system, the pixel color is in the Red, Green, and Blue (RGB) color space, and **I**(x, y, i) is a vector with three elements of I^R(x, y, i), I^G (x, y, i), I^B (x, y, i). For a given pixel (x, y), a matrix for pixel (x, y) is generated for color channel red as follows:

$$M^R(x, y) = \{I^R(x, y, 20\times i), i=1,2, 3, ..., 25\} \quad (1)$$

The background image's color value in Red channel at this location is calcuated as:

$$B^R(x, y, B)=Mode(M^R(x, y)) \quad (2)$$

The color values in the other two color channels are calculated in the same way and the color value of the background image at location (x, y) is generated. Applying the same process to each location and the whole background image can be created.

This method can accurately compute the background for relatively low to fairly high traffic volumes. The background is computed quickly enough to be updated in real time, as often as needed.

After the background is obtained, it is stored and subtracted from each incoming frame. Only the grayscale value is compared during this process. A threshold *Th* is applied to the result of the subtraction to create a binary image where the white pixels represent areas of sufficient difference from the background. That is, for a given frame *i*, the binary image' value at location(x, y) is calculated as follows:

$$\delta(x, y, i) = \begin{cases} 1, for \left|I(x, y, i) - B(x, y, i)\right| > Th \\ 0, for \left|I(x, y, i) - B(x, y, i)\right| \le Th \end{cases} \quad (3)$$

Clusters of white pixels that are larger than a user specified threshold are then considered moving objects.

Object Tracking

Once the objects are obtained, they have to be tracked from frame to frame. In order to accomplish this, several parameters are kept about the each object of interest. Overall, there are 18 parameters that are kept, but most important of them are position, grayscale intensity distribution, frequency, ID tag and a list of its composing objects, if any. The key to tracking objects is to identify the objects present in the current frame in the next frame. This is done by comparing the parameters of the old objects with those of new objects, as recommended by Owens et al. *(5)*. In PedTrack, the difference vector of the two potentially matching objects *Oi* and *Oj* is calculated as follows:

$$d(Oi, Oj) = \left(\left|Ai - Aj\right|, \left|Hi - Hj\right|, \left|Wi - Wj\right|, \left|Gi - Gj\right|\right) \quad (4)$$

Where *Ai* and *Aj* are object areas, *Hi* and *Hj* are object heights, *Wi* and *Wj* are object widths, and *Gi* and *Gj* are objects' grayscale histograms. The grayscale histogram difference $\left|Gi - Gj\right|$ is calculated as follows:

$$\left|Gi - Gj\right| = \sum_{k=0}^{255} \left| f_{k,Oi} - f_{k,Oj} \right| \quad (5)$$

where $f_{k, Oi}$ and $f_{k, Oj}$ are the frequencies for grayscale value *k* in the histogram of objects *i* and *j*, respectively. In order to be able to compare the objects, a cost is computed between each current object and its potential match. The cost function is the normalized difference of the parameter vectors of two potentially matching objects, and it is calculated as follows:

$$c(Oi, Oj) = \sum_{n=1}^{4} \frac{d_n(Oi, Oj)}{R_n(Oi)} \quad (6)$$

where $d_n(Oi, Oj)$ is the n^{th} element of $d(Oi, Oj)$, and $R_n(Oi)$ is the n^{th} element of and $R(Oi)$ is the attribute vector of object *Oi*, which is calculated as follows:

$$R(Oi) = (Ai, Hi, Wi, Gi) \tag{7}$$

where

$$Gi = \sum_{k=0}^{255} f_{k,Oi} \tag{8}$$

Once the minimum cost match for a particular object pair is found, it is considered to be the same object as it progresses from frame to frame, with exceptions for newly appearing or close to disappearing objects. Therefore, the tracking of individual objects is accomplished reliably with low chance of confusion between the objects.

Occlusions are imminent in a system based on background subtraction, as the clusters of white pixels leak together when the objects they represent get close. Occlusion reasoning is the most important part of any robust tracking system. Because occlusion reasoning can be done on many levels, it is important to establish how much information needs to be extracted from occluded objects – for our purposes, we are only interested in the approximate paths of the pedestrians and thus it is not imperative to know the exact location of a particular pedestrian inside a group. This key assumption simplifies the problem to one of much more manageable size. Now, the only reasoning that needs to be done is creation of composite objects – objects that contain several pedestrians, as well as logic for splitting these composite objects into individual pedestrians.

A composite object is potentially created when two or more objects in the current frame have minimum costs associated with the same object in the next frame. If such an event occurs, the sum of the parameter vectors of the two potentially composing objects is compared to the larger single object of the next frame, if the match cost is lower than that of one of the individual object's matches to the larger object, it is considered composite. In that case, a composite object is created as one containing those objects and is tracked as a single object. A split event happens when there are two or more objects in the next scene matching one in the current scene and the sum of those two makes a better match than any individual pairing. In that case it becomes necessary to find out which of the composing objects split off. This is done by comparing the current split object to ones stored inside the composite object and finding the lowest cost match. Since most occlusions are fairly short, usually involving one pedestrian walking in front of another, it is reasonable to assume that their inherent characteristics change only slightly, thus still allowing a correct match. Very long occlusions that involve a large deformation of the composing objects as well as those that happen outside the field of view and then enter as one object are still issues to be dealt with. As mentioned before, an additional restriction is also enforced to assure better matches – objects may split off only one at a time – this reduces the amount of confusion between the composing objects. Since we use high frequency video input of 30 frames per second, the chance of having more than one object splitting off in the same frame is minimal and the overall reduction in ambiguity outweighs the potential error.

Object Classification

Object classification is an important procedure for identifying the correct object types. In this study, objects are classified based on their inherent immediate properties rather than behavioral patterns. The inherent properties in PedTrack include size and height-to-width ratio. An object *Oi* will be identified as a pedestrian when and only when:

$$\begin{cases} Ai > A_{ThL}, and\ Ai < A_{ThH}, and \\ \dfrac{Hi}{Wi} > R_{ThL}, and \dfrac{Hi}{Wi} < R_{ThH} \end{cases} \quad (9)$$

where A_{ThL}, A_{ThH}, R_{ThL},and R_{ThL} are the given thresholds of minimum area, maximum area, minimum height-to-width ratio, and maximum height-to-width ratio, respectively.

Composite objects are absolved of these constraints – as only valid pedestrian objects may combine to become composite objects, and composite object may grow in size as well as erratically change their height–to-width ratio. Thus, groups of pedestrians do not pose a problem for this simple, yet effective method of classification.

For each identified pedestrian, a unique ID is assigned to it. This ID will remain the same for a given pedestrian during the video stream, including when the tagged pedestrian is occluded by others.

TESTS AND RESULTS

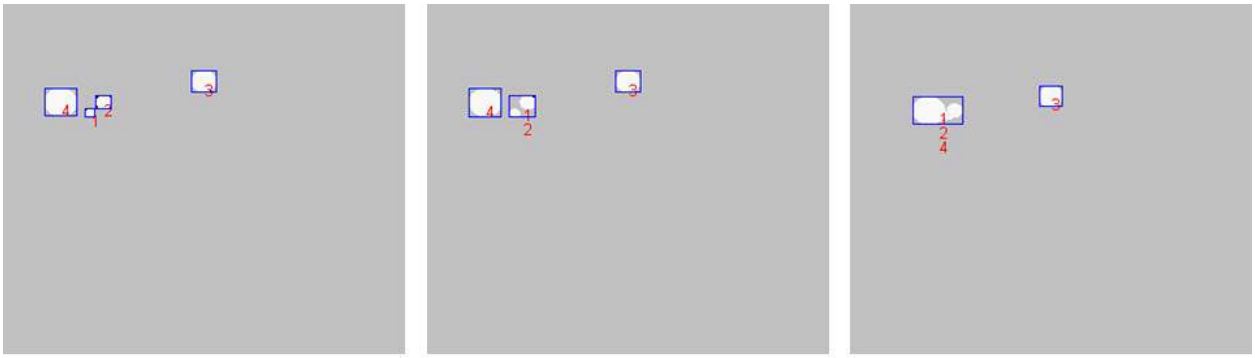
The proposed algorithm has been implemented in the prototype PedTrack system using Microsoft Visual C#. Currently the system is able to analyze AVI video streams in real time using a 1.7 GHz Centrino processor. It was evaluated with both synthetic tests and real-world video based tests.

It has been a challenge to test video tracking systems, as it is impossible to predict all types of situations and scenes that may be encountered during real-life exploitation of the tracking system. It is often the case that an algorithm is developed using a limited number of video sequences as test subjects and works well for those particular sequences, yet when it is applied to another common situation that is still in scope, the results are less encouraging. In an attempt to avoid this issue, a library of synthetic tests was created for common merging/splitting scenarios, as well as noise and distraction events. Figure 3 shows the examples of merging and splitting in synthetic tests. These types of tests are specific enough to cover most of the events that can occur and general enough to represent any camera placement situation in a large surveillance area – i.e. one where perspective changes are minimal. There are only a few types of cases that can happen between the pedestrian objects – merge/split of two single objects, merge/split of two composite objects, or a merge/split of a composite and a single object. All these cases were tested with simulated events. Other events involve noise or distractions – such as a momentary appearance/disappearance of an object. We tested our algorithm with salt and pepper noises and randomly generated distraction objects. Our algorithm performed very well in the synthetic tests conducted.

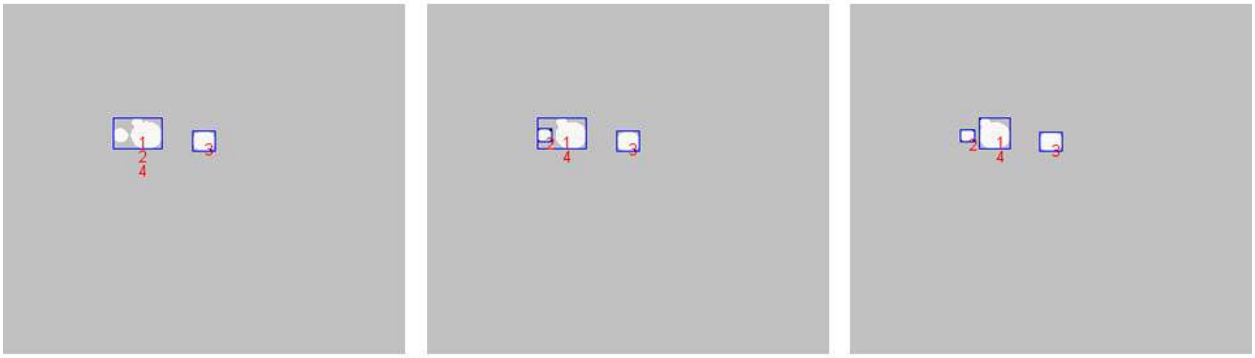
After establishing a competent library of these synthetic tests and ensuring the system's performance was adequate in the tests, real video was used. Although the algorithm performed well in synthetic tests, tests with real-world video data are still necessary due to the following two major reasons: (1) a real-world object with motion looks different from time to time in two-dimensional images; and (2) there are various disturbance types rather than the salt and paper noise simulated.

Considering that one of the potential uses of this system is proper crosswalk placement, footage of various crosswalks was used. In particular, crosswalks with and without an intersection were observed. Test video one was taken by the detection video camera shooting toward the east direction at the intersection of Wildwood Drive &164th Street SW, Snohomish

County, WA. Test video two was taken at the University of Washington campus using a research camera (Canon L2). A snapshot for each of the two test sites is shown in Figure 4. A total of thirty minutes of video, fifteen minutes from each site, were analyzed in this experiment.



(a) Merge event logic



(b) Split event logic

FIGURE 3 Merge and split event logic using synthetic test.

(a) Street Intersection (b) U W Crosswalk

FIGURE 4 Snapshots of test sites.

Figure 5 also shows an example of an intersection crosswalk where a merge event has occurred. Pedestrians 1 and 2 leak together and form a composite object that contains both 1 and 2. Because of the camera's setting angles, pedestrian occlusion is a major problem at this test site. A common occlusion event is shown in Figure 6 - pedestrian 2 passes pedestrian 1, creating a temporary occlusion. As a result of the occlusion, a composite object is created containing both pedestrians, as can be seen in Figure 6 (b) and (c). Once pedestrian 2 splits off the composite object, only pedestrian 1 is left – therefore the composite object turns into a regular pedestrian with ID 1, as can be seen in Figure 6 (d).

FIGURE 5 Sequence showing merge event.

In test site two, the camera was set about 60 ft (18 m) above the ground. Pedestrians look much smaller at this site than those in test site one. This created some challenges for pedestrian detection and tracking because the object size was small and features of objects become similar. In fact, there were some cases where the occlusion reasoning fails, as can be seen in Figure 7. In the frame on the left, a group of three pedestrians is mistaken for a group of two, as two of the pedestrians in the group walked into the scene together. The frame on the right shows that a momentary split of the group allowed tracking of the correct number of pedestrians in the group, yet the labeling is incorrect. The newly found pedestrian is labeled 5, while pedestrian 2 is lost – too much time has passed since pedestrian 2 has appeared as an individual – thus when it appears again its appearance is too different to ensure a successful match.

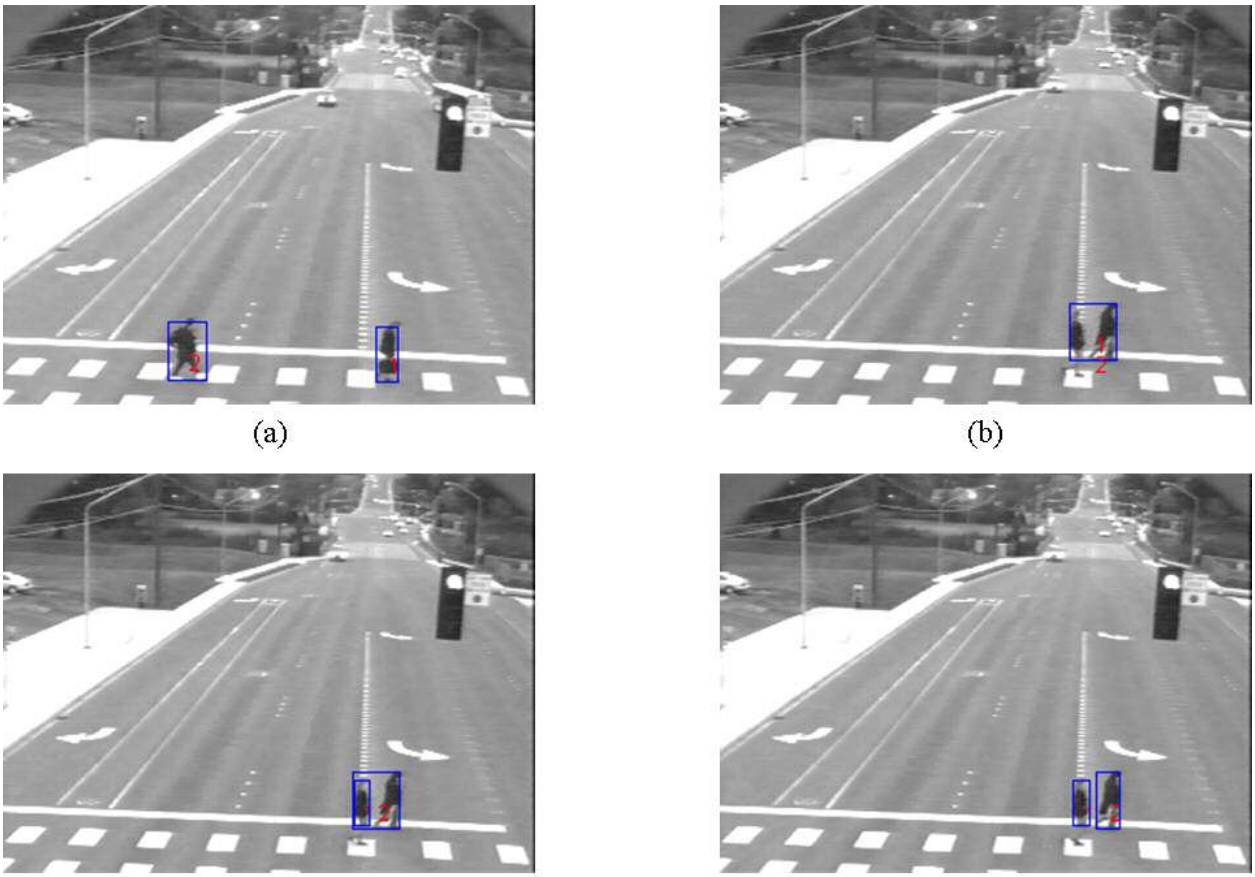


FIGURE 6 An occlusion occurs while a pedestrian passes another.

FIGURE 7 Occlusion reasoning failure.

In general, the PedTrack system performed well. The overall pedestrian detection rate was 89% for different camera positions and various occlusion scenarios involving 31 pedestrians in total. Correct ID assignment occurred about 80% of the time and the occlusion reasoning was correct about 75% of the time.

CONCLUSION

Pedestrian detection and tracking is an important issue for traffic operations and planning. It is also a very challenging issue compared to vehicle detection and tracking. Of the several technologies developed for pedestrian detection and tracking, video-based systems have attracted more attention in the past several years due to the wide deployment of video cameras for traffic surveillance and detection. The low marginal cost to add the capability of pedestrian detection and tracking to the existing video infrastructure makes the video-based systems even more attractive.

In this study, a simple, yet effective algorithm was developed for the purposes of pedestrian detection tracking. The algorithm was successfully implemented in the PedTrack system using Microsoft Visual C#. The PedTrack system was first evaluated by synthetic tests. The system performed extremely well in these tests. Then real-world video data collected from two test locations were applied to test the performance of the PedTrack system under real environment. The overall pedestrian detection rate was 89%. Correct pedestrian ID assignment occurred about 80% of the time in varying sequences involving fairly complex occlusion scenarios. Occlusion reasoning was correct about 75% of the time. Considering the overall simplicity and universality of the system, the results are encouraging.

The algorithm also provides a solid base for further refinements to improve overall accuracy. In particular, further work will be done to allow longer occlusions as well as refinements in the object classification step of the algorithm.

REFERENCES:

1. Hughes, Ronald, Herman Huang, Charles Zegeer, and Michael Cynecki. Automated Pedestrian Detection Used in Conjunction with Standard Pedestrian Push Buttons at Signalized Intersections. In *Transportation Research Record: Journal of the Transportation Research Board, No. 1705*, TRB, National Research Council, Washington, D.C., 2000, pp. 32-39.
2. Beckwith, Dana M. and Katharine M. Hunter-Zaworski. Passive Pedestrian Detection at Unsignalized Crossings. In *Transportation Research Record: Journal of the Transportation Research Board, No. 1636*, TRB, National Research Council, Washington, D.C., 1998, pp. 96-103.
3. SRF Consulting Group, Inc. and Minnesota Department of Transportation. *Bicycle and Pedestrian Detection: Final Report*. FHWA, U.S. Department of Transportation and Minnesota Department of Transportation, 2003.
4. Bureau of Transportation Statistics. *Bicycle and Pedestrian Data: Sources, Needs, and Gaps*. Report No. BTS 00-02. Bureau of Transportation Statistics, U.S. Department of Transportation, 2000.
5. Owens, Jonathan, Andrew Hunter and Eric Fletcher. A Fast Model-Free Morphology-Based Object Tracking Algorithm. In Proceedings of *British Machine Vision Conference*, Cardiff, UK, September 2002, pp. 767–776.
6. Heikkilä, Janne and Olli Silvén. A Real-Time System for Monitoring of Cyclists and Pedestrians. *Second IEEE Workshop on Visual Surveillance (VS'99)*, June 1999, pp. 74 – 81.

7. Collins, Robert T. Mean-shift Blob Tracking through Scale Space. Proceedings of *the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'03)*, Volume 2, June 2003, pp. II - 234-240.
8. Sheikh, Yaser, Yun Zhai, Khurram Shafique, and Mubarak Shah. Visual Monitoring of Railroad Grade Crossing. Proceedings of *SPIE -Int. Soc. Opt. Eng. (USA)*, vol. 5403, no. 1, April 2004, pp. 6546-60.
9. Masoud, Osama and Nikolaos P. Papanikolopoulos. A Novel Method for Tracking and Counting Pedestrians in Real-Time Using a Single Camera. *IEEE Transactions on Vehicular Technology*, Volume 50, Issue 5, September 2001, pp. 1267 – 1278.
10. Zheng, Jianyang, Yinhai Wang, Nancy L. Nihan, and Mark E. Hallenbeck. Extracting Roadway Background Image: Mode-Based Approach. *Transportation Research Record*, In press. National Research Council, Washington, D.C., 2006.
11. Jorge, Pedro Mendes, Arnaldo J. Abrantes, and Jorge S. Marques.. Automatic Tracking of Multiple Pedestrians with Group Formation and Occlusions. *IASTED International Conference on Visualization, Imaging and Image Processing*, Marbella, Spain, September, 2001, pp. 613-618.
12. Pai, Chia-Jung, Hsiao-Rong Tyan, Yu-Ming Liang, Hong-Yuan Mark Liao, and Sei-Wang chen. Pedestrian Detection and Tracking at Crossroads. *Pattern Recognition, 37(5)*. 2004, pp. 1025-1034.