



Pedestrian Detection System

Mahak Chittoda¹, Neelesh Vaishnav², Palash Jain³, Pranav Bhardwaj⁴

^{1,2,3}Students, Department of Computer Science and Engineering, Acropolis Institute of Technology and Research, Indore, M.P., India

ABSTRACT: -

In environments where a camera is installed on a freely moving platform, e.g., a vehicle or a robot, object detection and tracking becomes much more difficult. In this paper, we present a real time system for human detection, tracking, and verification in such challenging environments. To deliver a robust performance, the system integrates several computer vision algorithms to perform its function: a human detection algorithm, an object tracking algorithm, and a motion analysis algorithm. To utilize the available computing resources to the maximum possible extent, each of the system components is designed to work in a separate thread that communicates with the other threads through shared data structures. The focus of this paper is more on the implementation issues than on the algorithmic issues of the system. Object oriented design was adopted to abstract logarithmic details away from the system structure.

Introduction

The problem of object detection and tracking in video sequences becomes much harder when the camera is allowed to move uncontrollably. If the object of interest is a deformable object, like a human, the problem becomes even more challenging. Nevertheless, several interesting applications are waiting for solutions to this problem. A modern car prepared with a digital camera and a computer vision software for human detection can automatically avoid running over pedestrians. A military unmanned vehicle equipped with similar technology can automatically detect and deal with an enemy before being attacked.

In this paper, a real-time computer vision system for human detection and tracking in uncontrolled moving camera platforms is presented. The contribution presented in this paper is not in the algorithmic aspect of the system. Rather, our focus is on the system design and implementation aspects. Namely, our design was made to achieve two main goals: robustness and efficiency. Robustness was achieved through integration of algorithms, for human detection, tracking, and motion analysis, in one framework so that the final decision is based on the agreement of more than one algorithm. Efficiency was achieved through multi-threaded design and usage of a high-performance library. A final merit of our system is its object-oriented design. Object orientation was adopted to abstract the algorithmic details away from the system design so that we can easily experiment with different algorithms. Therefore, our system can be regarded as a testbed in testing different algorithms for human detection, tracking, and motion analysis.

The rest of the paper is organized as follows: Section 2 explores some of the related works. Section 3 explains our system design in detail. Section 4 briefly introduces the algorithms used in our implementation. Section 5 presents some experimental results. Finally, Sec. 6 concludes our paper.

Problem Formulation

The objective of this project is to address the different approaches and methods to reduce the pedestrian fatal injuries or accidents by integrating the safety feature on vehicle front end or as separate unit assembled to the vehicle and using advanced sensors to detect human movement on autonomous vehicles.

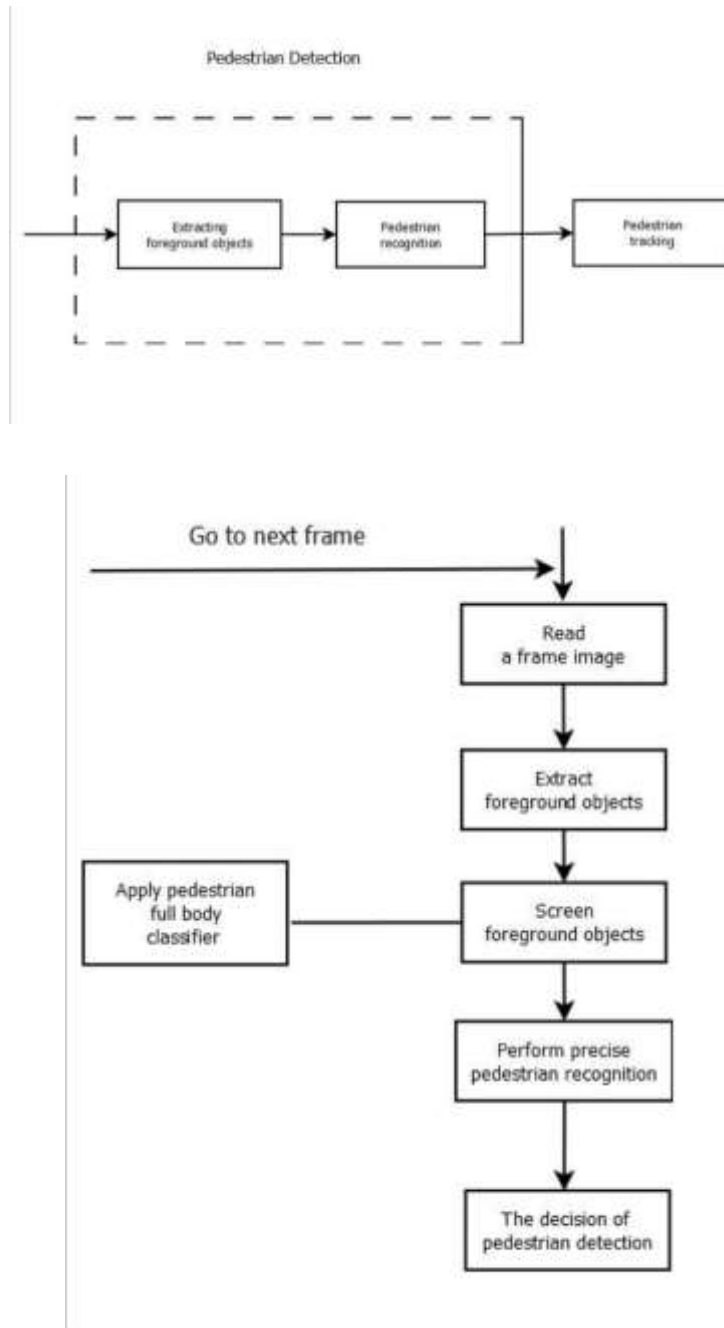
I. Literature Review

Since our focus in this paper is on the implementation issues of our system, a review of technical approaches in video surveillance will not be much relevant. For a global overview on automated surveillance systems, the reader is referred to [4]. Readers interested in broader and deeper coverage of technical details are referred to any recent survey on video surveillance, such as [8], [13], or [7]. For completeness of the discussion, we selected few systems that are relevant to ours in function to describe briefly here. Perhaps, a closely related system to ours is the DETER system [9]. DETER can detect and track humans, and can analyze the trajectory of their motion for threat evaluation. The W^4 system [6] detects humans and their body parts and analyzes some simple interactions between humans and objects. In [11], the idea of combining several cues together to enhance the robustness of tracking was utilized. Our system utilizes a similar idea with one step further. It uses different algorithms with different cues instead of one algorithm with different cues. IBM S3-R1 [5] can detect, track, and classify moving objects in a scene, encodes detected activities, and stores all the information on a database server to enable a broad range of queries on it. We have to make it clear that all these systems, though close to ours in function, work only in a static camera environment, but, our system works in an uncontrolled moving camera environment

II. Methodology

Figure 1 depicts the architecture of our system. The main modules of our system, as they are shown in Fig. 1, from left to right, are: the frame grabbing module, the human detection module, the object tracking module, the motion analysis module, and finally the output module. Input of the system is video frames obtained either online from a video camera, or off-line from a video file or individual video frame files. The output can be shown online or stored to the disk for further analysis.

Each of the modules runs in a separate thread and data is passed from one module to another via a shared data structure. Details of inter-thread communication are given in Sec. 3.4.



1.1 System Components

In this section, we describe the operation of each module in the system.

1.1.1 Frame Grabbing Module

The frame grabbing module is responsible for dealing with the input device. The input device can be a digital video camera connected to the computer, or a storage device on which a video file or individual video frames are stored. This module abstracts the nature of the input device away from the rest of the system so that changing the input device does not affect the rest of the system.

1.1.2 Human Detection Module

This module is responsible for invoking the detection algorithm. Ideally, the detection algorithm is to be run on each input frame. However, this will inhibit the system from meeting its real time requirements. Instead, the detection algorithm in our implementation is invoked every two seconds. The location of the human targets in the remaining time is determined by tracking the detected humans using the tracking algorithm. To further speed up the process, the detection algorithm does not look for humans in the entire frame. Instead, it looks for humans in the regions determined to be foreground regions. To determine the foreground regions, a stabilization algorithm is used to align the current frame with a preceding frame and with a succeeding frame. After alignment, the current frame is subtracted from the two other frames. The result of each subtraction is thresholded to form a binary image that represents the locations of foreground objects in the two subtracted frames.

1.1.3 Object Tracking Module

This module processes frames and detections received from the human detection module, and retains information about all the existing tracks. When a new frame is received, the already existing tracks are extended by locating the new bounding box's locations for each track in this frame. If the frame is received accompanied with new detections, the new detections are compared to the already existing tracks. If a new detection significantly overlaps with one of the existing tracks, it is ignored. Otherwise, a new track is created for this new detection. A track is discontinued if the tracking algorithm fails to extend it in a newly coming frame.

1.1.4 Motion Analysis Module

When the length of a track exceeds some specific length, typically, two seconds, the motion analysis module is invoked. The motion analysis module analyzes the periodicity encountered in the track. Based on the result of this analysis, it decides whether the tracked object is indeed a human or not. This way, the detection results are double checked by the motion analysis. In our experiments, that results in a reduction in the false positives produced by the detector, as shown in Sec. 5.

1.1.5 Output Module

When each track in a frame has been either analyzed by the motion analysis module, or removed because of being too short to be analyzed, this frame is ready for output and passed to the output module. The output module marks the detected human locations in the frame and sends it to the output device, which can be the display monitor or a storage device.

1.2 Inter-Thread Communications

Multi-threading in our system is implemented using the open threads open-source threading library [1]. Two threads communicate with one another through the shared data structure that both can access. To prevent race conditions between any two threads sharing a data structure, a template class is designed that automatically defines the Open Threads objects that are necessary for mutual exclusion enforcement for any given data type. In this section, the details of the data structures at the interface between each pair of communicating threads are explained. Figure 1 illustrates the various data structures used.

1.2.1 Frame Grabbing and Human Detection Modules

The shared data structure between the frame grabbing module and the human detection module is just a queue of pointers to frames.

1.2.2 Human Detection and Object Tracking Modules

This interface is a queue of structure elements. Each element contains a pointer to the frame along with a list of detections found in it. If the list of detections is empty, then, either the detection algorithm was not run on this frame, or the detection algorithm did not find any human in it.

1.2.3 Object Tracking and Motion Analysis Modules

In the shared data structure of this interface, two lists are maintained: a *frames queue*, which is a queue of *frames queue elements*, and a *tracks list* which is a list of pointers to *track element lists*. Each *frames queue element* contains a pointer to a frame and a counter that holds the number of tracked objects in that frame. Each *track element list* is, in turn, a list of *track elements*. A *track element* represents the location of the tracked object in one of the frames

pointed to by an element in the frames queue. A track element contains two items: a bounding box that specifies the location of this track's object in the corresponding frame, and a pointer to the entry of this frame in the frames queue.

1.2.4 Motion Analysis and Output Modules

The interface between the motion analysis module and the output module is simply a queue of pointers to frames that has become ready for output. The Motion Analyzer Module is responsible for sending frames ready for output to the output module, along with the bounding boxes that identify the targets that have been verified to be humans.

4.1 Stabilization Algorithm

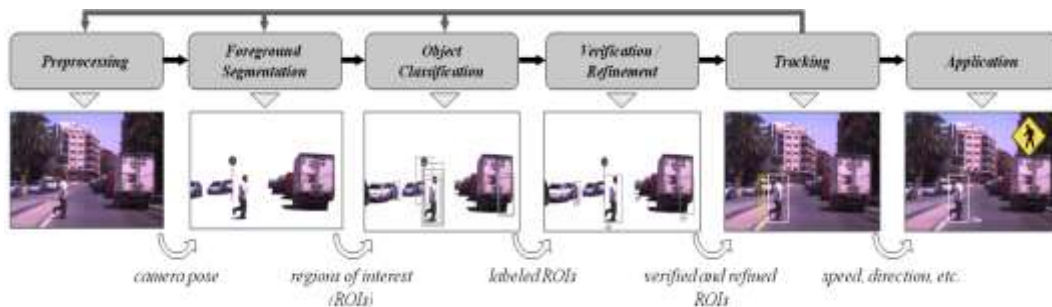
Wolberg and Zokai [14] presented an image registration algorithm which recovers affine motion between a pair of images. The algorithm proposed in [14] uses a log-polar transformation of the image pair to recover translational, rotational and scale misalignments. For a detailed discussion of the stabilization algorithm, the reader is referred to [14]. Briefly, assume that we need to register the image pair I_1 and I_2 . A rotation between the two images in the Cartesian space is indeed a translation in the polar space. Therefore, the rotation can be recovered by transforming the image pair to the polar space and location of the maximum cross correlation.

4.2 Human Detection Algorithm

The human detection algorithm used in our implementation was introduced in [3]. This algorithm searches for humans in the image by matching its edge features to a database of templates of human silhouettes. Examples of these templates are shown in Fig. 2. The matching is done by computing the average Chamfer distance [2] between the template and the edge map of the target image area. The image area under consideration must be of the same size as the template. Let the template T be a binary image that is 0 everywhere except for the silhouette pixels where the value is 1, and let the Chamfer distance transform of the edge map of the target image area be denoted by C . The distance between a template T and the target image area I can be computed by

4.3 Object Tracking Algorithm

The tracking algorithm used in our system is the one in [15]. The intensity value of each pixel is modeled as a time varying mixture of three Gaussian components, F , S , and W . The F component is the fixed component, which represents the appearance that is expected to be encountered most of the time. The S component represents the stable structure of the object's appearance observed over time. The W component, also called the wandering component, represents the transient changes in the appearance between two successive frames. Let the appearance model at time t be denoted by A_t . A_t is composed of three components F_t , S_t , W_t . Let $\mu_{f,t}$, $\mu_{s,t}$, and $\mu_{w,t}$, and $\sigma_{f,t}$, $\sigma_{s,t}$, and $\sigma_{w,t}$, and $m_{f,t}$, $m_{s,t}$, and $m_{w,t}$ denote the means, the standard deviations, and the mixing probabilities for the F , S and W



Frames ready for output to the output module, along with the bounding boxes that identify the targets that have been verified to be humans.

4.4 Stabilization Algorithm

Wolberg and Zokai [14] presented an image registration algorithm which recovers affine motion between a pair of images. The algorithm proposed in [14] uses a log-polar transformation of the image pair to recover translational, rotational and scale misalignments. For a detailed discussion of the stabilization algorithm, the reader is referred to [14]. Briefly, assume that we need to register the image pair I_1 and I_2 . A rotation between the two images in the Cartesian space is indeed a translation in the polar space. Therefore the rotation can be recovered by transforming the image pair to the polar space and location of the maximum cross correlation.

4.5 Human Detection Algorithm

The human detection algorithm used in our implementation was introduced in [3]. This algorithm searches for humans in the image by matching its edge features to a database of templates of human silhouettes. Examples of these templates are shown in Fig. 2. The matching is done by computing the average Chamfer distance [2] between the template and the edge map of the target image area. The image area under consideration must be of the same size as

thetemplate. Let the template T be a binary image that is 0 everywhere except for the silhouette pixels where the value is 1, and let the Chamfer distance transform of the edge map of the target image area be denoted by C . The distance between a template T and the target image area I can be computed by

4.6 Object Tracking Algorithm

The tracking algorithm used in our system is the one in [15]. The intensity value of each pixel is modeled as a time varying mixture of three Gaussian components, F , S , and W . The F component is the fixed component, which represents the appearance that is expected to be encountered most of the time. The S component represents the stable structure of the object's appearance observed over time. The W component, also called the wandering component, represents the transient changes in the appearance between two successive frames. Let the appearance model at time t be denoted by A_t . A_t is composed of three components F_t , S_t , W_t . Let $\mu_{f,t}$, $\mu_{s,t}$, and $\mu_{w,t}$, and $\sigma_{f,t}$, $\sigma_{s,t}$, and $\sigma_{w,t}$, and $m_{f,t}$, $m_{s,t}$, and $m_{w,t}$.

4.7 Motion Analysis Algorithm

Since the human motion is naturally different than other types of motions, the sequence of bounding boxes can be further analyzed to verify whether or not the tracked subject is indeed a human. We use the human motion analyzer proposed by Rental. [10]. The algorithm proposed in [10] tests the periodicity of the temporal pixels in order to prove or disprove the null hypothesis that the signal being tested is periodic. Briefly, for a given pixel location (x, y) in the bounding box, the algorithm computes the periodogram of the color value of (x, y) across all bounding boxes of subject. A peak in the periodogram proves that the spatiotemporal signal is periodic. On the other hand, a flat periodogram means that the spatiotemporal signal is a white noise. The periodicity test is repeated for all pixel locations in the bounding box. If the number of periodic pixels is higher than a certain threshold, then the subject undergoing the test is a human. Otherwise, the null hypothesis that the subject is a human is rejected. For more details on the algorithm, the reader is referred to [10].

III. Result Discussions

Our system was experimented on a set of challenging video sequences. It has succeeded to demonstrate robustness and close to real time performance (around 15 frames per second.) In this section, we will present the results of two sequences. In the figures presented, rectangular bounding boxes are the output of the detection algorithm. Green boxes are the detections that are verified by the motion analysis algorithm, and the red boxes are the ones that are rejected by it. The reader is referred to the electronic version for clarity of results.

In the sequence shown in Fig. 3, there was only one false detection and it was caught by the motion analysis. On the other hand, the sequence shown in Fig. 4 clearly shows the advantages of the verification step. The shape-based detection algorithm produced many false alarms due to high edge density on the left part of the scene. After tracking these false detections for a period of time, the motion analysis algorithm decided that they did not exhibit the periodic motion of a human.

IV. Conclusion

In this paper, a real-time computer vision system for human detection, tracking, and verification in uncontrolled camera motion environment has been presented. The key features of our system are robustness and efficiency. Robustness was achieved via integration of more than one algorithm, each of which uses a different visual cue to identify humans. The efficiency was achieved via a multi-threaded design with efficient inter-thread communication, and the usage of a highly optimized software library. The system has demonstrated a satisfactory performance on highly challenging video sequences. Our short-term plan is to further optimize our system and experiment with other algorithms. Our long-term plan is to extend the system to analyze human activities and evaluate threats.

Acknowledgment

We are profoundly grateful to Bharti Bhattad ma'am for her expert guidance and continuous encouragement throughout the process. We would also like to express our heartfelt gratitude to all those who helped directly or indirectly during this course of work.

References

- [1] Openthreads. <http://openthreads.sourceforge.net/>.
- [2] H. G. Barrow. Parametric correspondence and chamfer matching: two new techniques for image matching. In *International Joint Conference on Artificial Intelligence*, pages 659–663, 1977.
- [3] D. Gavrila. Pedestrian detection from a moving vehicle. In *ECCV '00: Proceedings of the 6th European Conference on Computer Vision-Part II*, pages 37–49, London, UK, 2000. Springer-Verlag.
- [4] A. Hampapur, L. Brown, J. Connell, A. Ekin, N. Hass, M. Lu, H. Merkl, S. Pankanti, A. Senior, C.-F. Shu, and Y. L. Tian. Smart video surveillance. *IEEE Signal Processing Magazine*, pages 38–51, March 2005.
- [5] A. Hampapur, L. Brown, J. Connell, N. Hass, M. Lu, H. Merkl, S. Pankanti, A. Senior, C.-F. Shu, and Y. L. Tian. S3-r1: The ibm smart surveillance system- release 1. In *ACM SIGMM workshop on Effective telepresence*, 2004.

-
- [6] I. Haritaoglu, D. Harwood, and L. Davis. w^4 : Real-time surveillance of people and their activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):809–830, 2000.
- [7] W. Hu, T. Tan, L. Wang, and S. Maybank. A survey on visual surveillance of object motion and behaviors. *IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews*, 34:334–352, 2004.
- [8] T. B. Moeslund and E. Granum. A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 81:231–268, 2001.
- [9] V. Morellas, I. Pavlidis, and P. Tsiamyrtzis. Deter: Detection of events for threat evaluation and recognition. *Machine Vision and Applications*, 15:29–45, 2003.
- [10] Y. Ran, I. Weiss, Q. Zheng, and L. Davis. Pedestrian detection via periodic motion analysis. *To Appear, International Journal on Computer Vision*.
- [11] M. Spengler and B. Schiele. Towards robust multi-cue integration for visual tracking. *Machine Vision and Applications*, 14:50–58, 2003.
- [12] S. Taylor. *Intel Integrated Performance Primitives: How to Optimize Software Applications Using Intel IPP*. Intel Press, 2003.
- [13] L. Wang, W. Hu, and T. Tan. Recent developments in human motion analysis. *Pattern Recognition*, 36(3):585–601, 2003.
- [14] G. Wolberg and S. Zokai. Robust Image Registration Using Log-Polar Transform. In *International Conference on Image Processing*, 2000.
- [15] S. K. Zhou, R. Chellappa, and B. Moghaddam. Visual tracking and recognition using appearance-adaptive model in particle filters. *IEEE Transactions on Image Processing*, 13(11):1491–1506, November 2004.