

An Embedded Real-time Vision System for 24-hour Indoor/Outdoor Car-Counting Applications

Ming-Yee Chiu, Setrix Inc.

Abstract

We describe an embedded vision system, which integrates a web-cam quality CMOS imaging chip with a RISC processor, to perform real-time car-counting functions in the indoor and outdoor environment. The challenge of this application, especially for the outdoor environment, is to develop vision algorithms for day and night, and during the light-transition periods (i.e., dawn and dusk).

The vision system also needs to accommodate a tremendous range of illumination change (from sunny summer to snowy winter). Finally we report briefly the result of an outdoor system we deployed in Germany since June 2003. The entire system consists of a network of 13 embedded vision systems covering a parking facility over 1 square kilometer. The vision-network has been in daily use for the employee parking guidance.

1. Introduction

We started this vision application with the intention to offer a low-cost vision solution to customers who cannot afford a high price tag for what they need: a system to inform the car drivers where to find the free parking space among various parking houses, parking lots, or different zones of parking garages. We shall use the term: “parking place” to describe a closed parking region where the number of parking spaces can be defined clearly and a car can enter or exit this closed region only through a few number of gates or checkpoints.

By counting the number of cars passing through the checkpoints, the number of free parking spaces of the parking place can then be obtained. Traditionally the induction loop sensors, buried under the road surface, are used to detect the car passage and provide the count data.

However, there are limitations using the induction loop solution. For example, if the driveway at the checkpoint is wide, the driver can drive in the middle and the induction sensor will have problems determining the car’s direction. Furthermore, the induction sensor cannot handle high-speed cars. The embedded vision system we developed uses a web-cam quality CMOS sensor integrated with straightforward computer architecture with no dedicated hardware in order to minimize cost. This is

in contrast to [1] where the VLSI processing architecture is emphasized for real-time performance.

The software is optimized in order to achieve the 30 frames per second processing performance. This processing speed is needed because the car’s speed can be as high as 90 km/h.

2. Basic Counting Principle

The principle of car counting is quite similar to the technique of light-beam blocking. As illustrated in Figure 1, the camera is mounted at a high position on one side of the driveway, looking down on a special marker located at the other side of the driveway, near the ground.

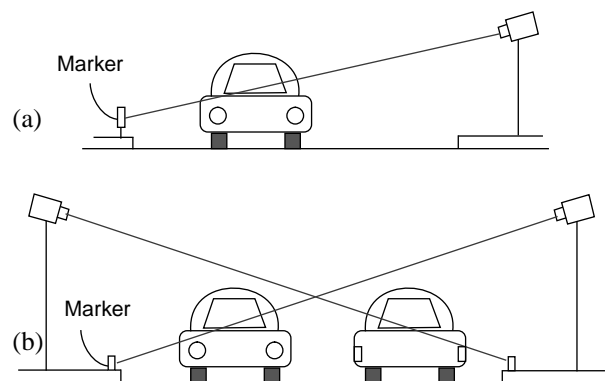


Figure 1 Geometry for car counting

The line connecting the marker and the camera defines a virtual line-of-sight. When there is no object passing, the vision system detects the presence of the marker. As soon as any object, be it a car, a truck, a pedestrian, or a bicycle, blocks the line-of-sight, the system immediately detects the absence of the marker. It then computes the motion of two consecutive frames and determines the type of the blocking object and its moving direction. For cars or trucks, the count of the cars inside the parking place is incremented or decremented depending on the moving direction. For pedestrians, bicycles, or motorcycles, they are excluded in the count since usually there are separate parking spaces for bicycles/motorcycles. The vision algorithm therefore needs to discriminate trucks or cars from pedestrians, bicycles, or motorcycles.

After the object discrimination, the system acquires a new frame and checks if the marker is visible. If not, the system repeats the motion processing and object discrimination until the marker becomes visible again. The entire process then repeats. One marker-blocked event can at most generate one vehicle-count.

In Figure 1(a), only one camera is used to count the cars from both directions.

Figure 1(b), with two cameras, is used for a wide driveway. It is possible that a tall vehicle (a van or truck) blocks the line-of-sight of both cameras. In this case, a fusion step is needed to eliminate the duplicate count from both cameras.

3. The 24-hour Marker and its Detection

As shown in Fig. 2, we use a special black and white pattern as a marker. The pattern consists of 8 radial-shape segments radiating from a center. This pattern is a special case of a more general “barcodable marker” which can be used for simultaneous identification and localization of objects [2]. The pattern can be detected reliably under cluttered environment by the use of a Hough Transform that accumulates a high peak at the center. For convenience, we shall call this black and white pattern a “sunny” pattern.

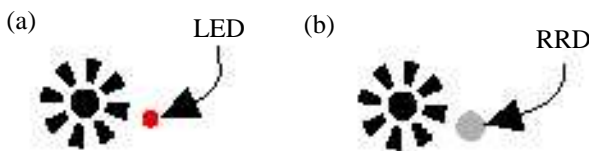


Figure 2 The 24-hour Marker

The image resolution used for the marker detection is 320x240 pixels. During the initial calibration stage, the sunny pattern is detected and localized by processing the entire image using the Hough Transform. Five parameters: the x, y location, the major, minor radius and the ellipse angle of pattern are extracted and stored. This time-consuming operation is only performed once and we will not discuss the algorithm here. (See [2] for details).

Figure 3 shows the actual marker setup we installed at the customers site. The marker is positioned near the horizontal center of the image. The images also show the real-life environment and the diverse illumination range that the vision algorithms must deal with.



Figure 3 The marker setup in real-life

For run-time operation, we use a simplified sunny detection algorithm that processes only the local region around the sunny pattern. First, we sample the intensity profile along an elliptical path that cuts through the black and white radial segments of the pattern, using the 5 sunny parameters extracted during the calibration stage. We then extract the edge locations from the 1-D intensity profile and check if there are 16 transitions with roughly equal spacing. If yes, the sunny pattern is detected.

Several sampled intensity profiles with different radii are tried for robust detection. Furthermore, the central location of the sunny pattern is also varied by +1 to -1 pixel in x- and y-direction to handle possible camera shift due to temperature change.

The edge detection for the 1-D intensity profile must be sensitive enough to detect low contrast edges. This situation is common when the sunny pattern is under the tree shadow in a bright sunny day. The detection also needs to accommodate the changing shadow cast on the marker from trees.

At night, there are two solutions we use for the marker. One is an AC or battery powered LED (Fig. 2a). Another (Fig. 2b) is a retro-reflective disc (RRD) illuminated by an array of IR LEDs located inside the camera box. The LED or RRD are placed in a fixed position with respect to the center of the sunny pattern. Usually during night the LED/RRD is bright and its size small. Therefore to detect the LED/RRD, we first search the pixel with maximum intensity near the designed LED/RRD location. If this peak intensity exceeds a threshold, we then move from the peak location outward along 8 angular directions to find the locations where the intensity drops to a fixed percentage of the peak intensity. If these points are within a short distance from the peak, then it is

an LED/RRD. This simple algorithm is fast and effective. However, occasionally there is a bright spot on the car body that happens to pass through the LED/RRD's search region and is mistaken as an LED/RRD. This creates faulty marker detection and thus splits one marker-blocked event into two (or more). To remedy this problem, we check the duration of the marker-visible state. If it is below a threshold (e.g., 0.2 sec), then we merge these two marker-blocked events into one event.

4. The Motion Processing and Object Discrimination

As soon as the marker is blocked, the system starts the motion-processing step that uses a reduced resolution of 160x120 pixels (padded to 160x128 pixels). Figure 4(a) shows two consecutive images of a passing car.

The motion processing is based on the well-known motion estimation algorithm [3] used by most video compression standards. The image is partitioned into 10x8 macroblocks (MBs) each with 16x16 pixels. The motion vector is then computed by a search based on the SAD (Sum of Absolute Differences) measure. Before the motion estimation, we perform a screening process to ignore those MBs (indicated by a dot in Figure 4(b)) that have small image change or little vertical structure (since the motion is primarily in the horizontal direction). Figure 4(b) shows the computed x-motion components of the 10x8 MBs. The numbers indicate the pixel shift between two images and is proportional to the speed.

Since the car's movement is a rigid-body motion, we can impose a motion consistency check by selecting only the MBs with roughly the same horizontal movement. These MBs are called the "consistent macroblocks".

The result is shown in Figure 4(c). As can be seen from this example, the blurring of the images due to car motion has little effect on the result.

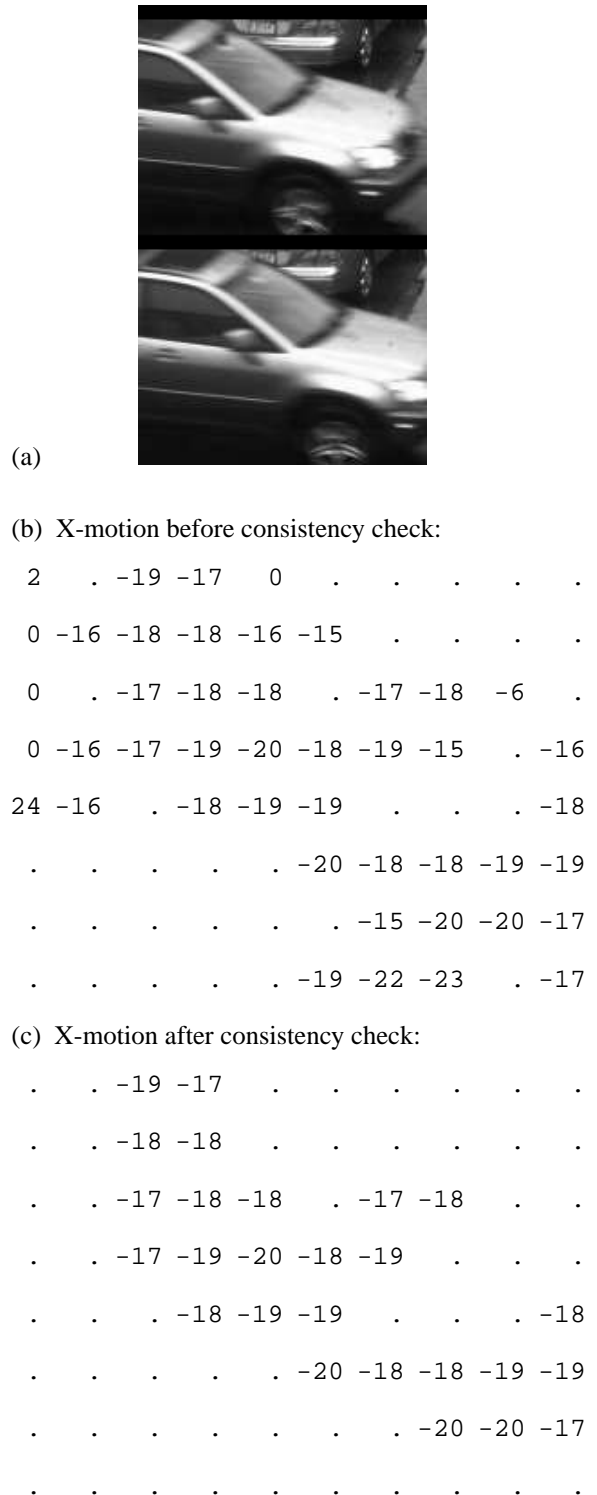


Figure 4 Motion estimation of two consecutive images

During night, we rely on the motion of the car's headlights/taillights. Figure 5 shows some images for the nighttime motion processing. As can be seen from the images, sometimes the headlights of other cars can illuminate part of the scene. The fine horizontal line pattern superimposed on the image is caused by the noise from the CMOS chip due to the high gain (7.5 times) used by the automatic gain control.



Figure 5 Night scenes for motion processing

The nighttime motion processing is different from the daytime motion processing. First the algorithm determines if the daytime processing is more appropriate by checking if the percentage of the bright pixels (i.e., pixels with intensity exceeding a high threshold) exceeds a limit (e.g. 25%). If yes, such as the case for the middle-left image of Fig.5, then the daylight processing is actually used. If not, which indicating that most area in the image is dark, we then check if the number of bright pixels is large enough for headlights/taillights. If yes, we enter into the headlight-processing mode by performing the same motion estimation and consistent motion check. Usually in this situation the number of consistent MBs is small (see the left and right images in Figure 5).

The object discrimination is based on the speed and the size of the blocked object using the 10x8 MBs. If the geometry of the setup (Figure 1) is selected properly, the cars or trucks usually have consistent MBs at the left and right portion of the image while for pedestrian or bicycles, the consistent MBs only occur in the middle of the image (Figure 6). Therefore we partition the 10x8 MBs into three regions: L, M, and R (Figure 7). L and R regions occupy 3 columns of MBs while M occupies the central 4 columns. There are 5 possible object types: (1) vehicle_to_right(V2R), (2) vehicle_to_left(V2L), (3) pedestrian_to_right(P2R), (4) pedestrian_to_left(P2L), and (5) nothing(N).



Figure 6 Classification of blocking objects

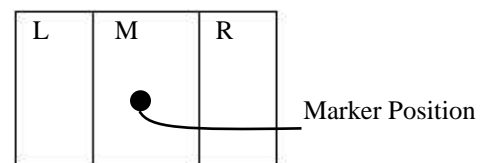


Figure 7 Partition of 10x8 motion MBs into 3 regions

The rules we use for object discrimination are:

- (1) If the speed is less than a threshold, we have type N.
- (2) For headlight processing, if the number of consistent MBs is at least 1, then we have type V2R or V2L depending on motion direction. Otherwise we have type N.
- (3) If the horizontal extend of the consistent MBs exceeds A (car's size), and the number of consistent MBs exceeds B, we have type V2R or V2L.
- (4) If the number of consistent MBs in regions R and L exceeds C and the number of consistent MBs exceeds B, we have type V2R or V2L.
- (5) Otherwise the object is type P2R or P2L, depending on motion direction.

A, B, and C are threshold numbers. The spatial discrimination rules above do not necessarily generate a correct result. For example, a black car under poor illumination can have a small number of consistent MBs (less than B) and thus it can be misclassified as a pedestrian.

The spatial object discrimination is performed for every frame during the entire marker-blocked period. This means we can refine the classification by performing a temporal processing by combining the spatial object types over several frames. We use a voting mechanism that produces a very reliable object type, especially

for classifying bicycles or motorcycles to the pedestrian type instead of the vehicle type.

5. Hand-over of Algorithms During Light-Transition Periods

Figure 8 shows images during the light-transition periods (dawn or dusk) that last roughly 0.5-1 hours. During this time the illumination changes dramatically.



Figure 8 Images during the light-transition periods

For the marker detection, the LED/RRD detection method is activated only when the scene is about to get dark. This time point is determined by monitoring the parameters from the automatic exposure control of the CMOS sensor. When activated, the LED/RRD detection is tried first. If it fails, the sunny pattern detection is tried next. This overlapping detection works extremely well to accommodate the gradual disappearance of LED/RRD during dawn.

For the handover of motion processing, the system uses a time point when the scene is darker. For the cases in Fig. 8, only the left-most event is processed by the nighttime algorithm.

6. Real-time performance

All real-time vision systems need to face the issue of uneven needs for processing power. This application is no exception. During the marker-blocked period, the computation is heavy since it needs to perform motion estimation, object discrimination and marker detection for every frame. We buffer the images so there are more time to process marker-blocked events at a later time. The algorithms are also arranged properly so that the computation is spread out evenly. At the end, we are able to achieve the 30 f/s processing speed even at high car passing frequency (30–40 cars/minute during the morning rush hour).

7. Deployed Systems

We installed the first indoor park-garage guidance system with 7 networked vision systems. This system is in daily use since May 2003. The outdoor system with 13 vision systems was installed in June 2003.

The outdoor vision system (Fig. 9) has a size of 20x12x10 cm. The hardware includes a camera, a RISC CPU, 16MB flash and 16MB RAM. The system consumes a maximum of 2W power. All vision systems are connected to a parking guidance controller via the RS-485 interface. The length of the serial cables alone is about 4.2 km. The total number of parking spaces managed by the vision systems is about 2,300. Two busiest vision systems count about 2,000 cars each way daily.

The system has an error rate of about 2–5%, which is satisfactory for the current operation.



Figure 9 The outdoor embedded vision system

8. References

- [1] W. Wolf, B. Ozer, and T. Lv, "Smart cameras as embedded systems", *IEEE Computer*, 35(9), 2002, pp. 48-53.
- [2] M.Y. Chiu, T. Spindler, "Hough Transform for Aggregate Shapes and the Design of a High-Density Barcodable Marker", submitted to CVPR 2004.
- [3] J.R. Jain and A.K. Jain, "Displacement measurement and its application in interframe image coding", *IEEE Trans. Commu. COM_29*, 1981, pp1799-1808.