Cooperative Visual Surveillance Network with Embedded Content Analysis Engine

Shao-Yi Chien and Wei-Kai Chan
Media IC and System Lab, Graduate Institute of Electronics Engineering and Department of Electrical Engineering, National Taiwan University
Taiwan

1. Introduction

Visual surveillance plays an important role in security systems of digital home and enterprise Regazzoni et al. (2001). Evolving from CCTV video surveillance, the IP camera surveillance system with Internet as the connection backbone is a trend in recent years. A typical IP camera surveillance system is shown in Fig. 1. IP camera systems have the advantages of easy setup and universal access ability; however, several issues in network transmission are introduced Foresti & Regazzoni (2001), which become more and more important when the number of camera grows. Since the surveillance systems share the same network with other applications and devices of digital home and enterprise, the congestion of network caused by transmission of large surveillance contents may degrade the service quality of the these applications, including the surveillance application itself. Besides, the control server can only afford the content storage from a limited number of video channels, which limits the system extension in camera number.
Further evolving from IP camera systems, the maturity of visual content analysis technology makes it feasible to be integrated into the next-generation surveillance systems to achieve intelligent visual surveillance network Mozef et al. (2001) Hu et al. (2004) Stauffer & Grimson (1999) Elgammal et al. (2000) Comaniciu et al. (2003) Cavallaro et al. (2005) Maggio et al. (2007), where high-level events can be automatically detected, and multiple cameras can cooperate with each other, including different types of fixed cameras and mobile cameras hold by robots, as shown in Fig. 1.
In the next-generation system in Fig. 1, namely cooperative visual surveillance network, new design challenges are introduced. First of all, the system configuration should be carefully designed since the distribution of the computations for these analysis functions will significantly affect the performances of these visual content analysis algorithms, and it will also affect the utilization efficiency of network resources. Moreover, the large computation of content analysis algorithms will increase the loading of servers, which will further limit the scale of camera number. Thanks to the advanced silicon manufacturing technology, which makes the transistor count in single chip increase dramatically, more and more functions can be considered to be integrated as a System-on-a-Chip (SoC) to cover more and more tasks for

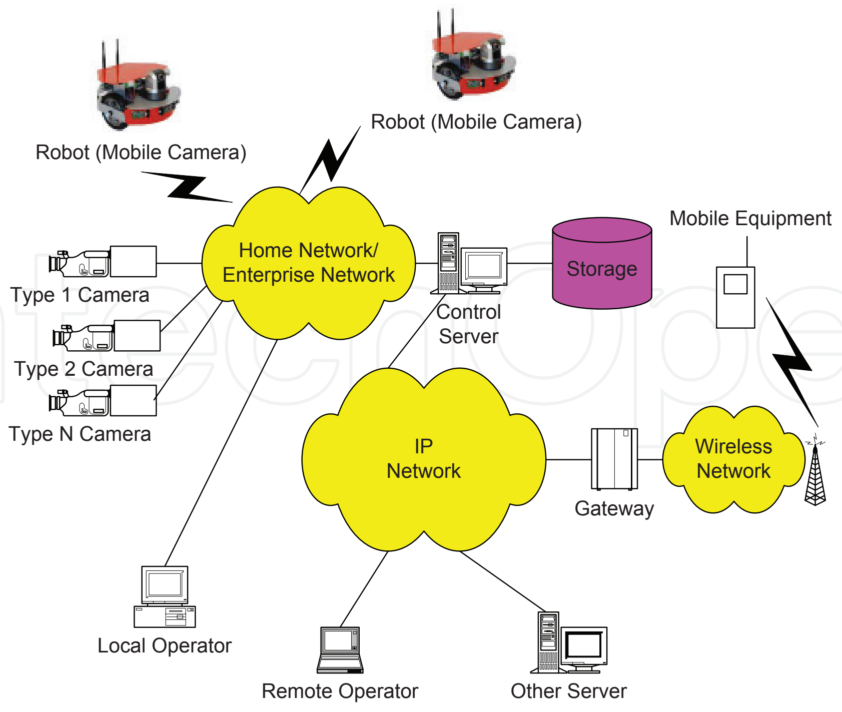


Fig. 1. Illustration of a cooperative video surveillance network.

surveillance applications Wolf et al. (2002). An efficient smart camera SoC is required to help reducing the deployment space and cost and solving the above issues.
This chapter is organized as follows. In Section 2, the issues in conventional IP camera surveillance systems are discussed, and our solution for the next-generation surveillance system is introduced. The surveillance system pipeline in system level, and an five-layered surveillance visual content abstraction hierarchy are presented. In Section 3, the proposed algorithms to be embedded in each surveillance camera are introduced, including video segmentation in complex and dynamic background, user-friendly video object description, efficient multiple video object tracking with split- and merge-handling, and efficient face detection and scoring. Next, in Section 4, our proposed visual content analysis hardware engine is introduced, where the proposed content analysis algorithms are implemented, and the overall hardware architecture for smart camera SoC is also provided. Two design examples are then demonstrated. The first one is a multi-fixed-camera surveillance system, which will be shown in Section 5; the second one is a surveillance network with several fixed cameras, one robot (mobile camera), and in-door localization system using Zigbee, which will be shown in Section 6. Finally, we will conclude this chapter and introduce some future directions in Section 7.

2. Proposed system configuration and data abstraction hierarchy

In this section, the system configuration of the next-generation surveillance systems are discussed with considering the issues mentioned previously, and a better system configuration is analyzed based on the surveillance pipeline model shown in Fig. 2. After that, the new ability of abstraction hierarchy of the next-generation surveillance systems is introduced as shown in Fig. 3. With content analysis ability, the scalability in IP camera surveillance systems can be extended across all of the five layers according to

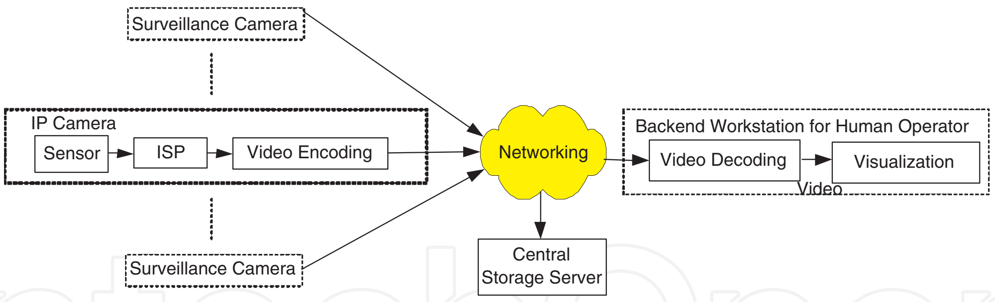


Fig. 2. Surveillance pipeline model of the conventional IP camera surveillance system.

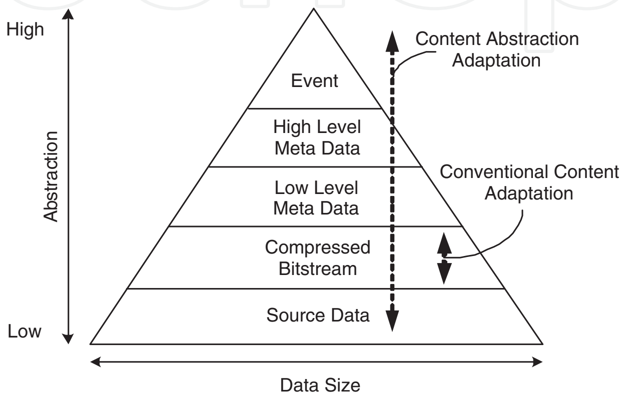


Fig. 3. Abstraction hierarchy of visual content in a surveillance network.

operation conditions, network conditions, and storage space conditions. The most important information in semantics will be transmitted under restricted communication and storage resources.
A conventional IP cameras surveillance system is modeled with the pipeline in Fig. 2. In this pipeline, there are seven important tasks: Video Sensing (*Sensor*), *Image Signal Processing (ISP)*, *Video Encoding*, *Network*, *Video Decoding*, *Content Storage*, and *Visualization*. In the *Back-end Workstation*, which is placed at the other side of the *Network* from the *IP camera*, the transmitted video stream is decoded in *Video Decoding*. The video data is then displayed with *Visualization* for the human operators. The coded video data is also received and stored in the *Central Storage Server*, and the *Back-end Workstation* and *Central Storage Server* can be integrated in the same machine in some cases. Note that the detailed tasks in *Network* are ignored in the pipeline since they are beyond the scope of this chapter. From Fig. 2, we can see that all of the compressed bitstreams from every IP camera should be transmitted through *Network* for inspection by the human operators. When the number of cameras increases, a network congestion problem will occur, and the surveillance system's performance will degrade significantly.

2.1 Content abstraction hierarchy

When visual content analysis tools are employed in surveillance systems, the essential change is the introduction of a content abstraction hierarchy, which is described in the first place before the system configuration discussion. As shown in Fig. 3, there are five different layers to represent the visual surveillance contents: *Source Data Layer*, *Compressed Bitstream Layer*, *Low Level Meta Data Layer*, *High Level Meta Data Layer*, and *Event Layer*. The *Low Level Meta*

Data represents the low-level features of the source video, such as color and shape features, and the *High Level Meta Data* is obtained through further analysis of the *Low Level Meta Data*, where the concept of "object" is considered, such as face locations, object trajectories, poses, and the gaits of human objects. *Event* represents the results of an event detection, where the event is defined in advance according to different application scenarios. In a conventional IP camera surveillance system, the content scalability only occurs in the *Compressed Bitstream Layer* to adjust the coding bitrate according to different network conditions or storage space conditions, which is indicated as the *Conventional Content Adaptation* in Fig. 3. With content analysis ability, the scalability in IP camera surveillance systems can be extended across all of the five layers in Fig. 3 according to operation conditions (results from event detection...etc), network conditions (network congestion...etc), and storage space conditions. That is, the most important information in semantics will be transmitted under restricted communication and storage resources.

2.2 System configuration discussion

In order to discuss the system configuration, surveillance pipeline models similar to the one in Fig. 2 are employed. Three representative surveillance pipelines are provided in Fig. 4. In these pipelines, *Content Analysis* tasks are added. With the minimum modification to the conventional surveillance pipeline in Fig. 2, the *Back-End Content Analysis* task in Fig. 4(a) utilizes visual content analysis tools in the back-end workstation after the video bitstream is received from the network and decoded. In such a surveillance pipeline, the delay of event detection due to network transmission delay could be expected. Moreover, the coding process and the packet loss in transmission may also degrade the video quality for analysis. Besides, when the number of IP cameras becomes large, more *Back-End Workstations* are required to provide sufficient computing power, which will dramatically increase the system cost and deployment space. In addition, the *Central Storage Server* cannot afford to perform all the storage tasks for all the surveillance cameras, and the network loading also increases for the purpose of storage. To deal with the above-mentioned problems, the *Back-End Content Analysis* can be moved to the front-end as distributed *Front-End Content Analysis Workstations*, and distributed *Local Storage* can be used to replace the *Central Storage Server*. The resultant pipeline from these two modification is shown in Fig. 4(b). In this pipeline, the network congestion problem can be solved by using a content abstraction hierarchy to transmit only the semantically meaningful information for visualization, while the *Local Storage* stores the complete video bitstream for off-line historical inspection and review. However, the number of *Front-End Content Analysis Workstations* will still increase as the number of cameras increases, which leads to large cost and deployment space. Consequently, a choice is made to replace the *Front-End Content Analysis Workstation* with a smart camera SoC with visual content analysis functions. For the issues mentioned here, the surveillance pipeline that features distributed local storage and smart camera SoCs in Fig. 4(c) is our chosen solution. It is the best solution among these three pipelines in terms of system cost, deployment space, network loading, and system scalability. The comparisons among these three pipelines are summarized in Table 1.

2.3 Proposed visulization strategy

A visualization strategy is also proposed with the five-layered visual content abstraction scalability in Fig. 3 and the selected pipeline in Fig. 4(c), where human objects is the focus in our system. This strategy consists of three condition levels: normal level, alert level,

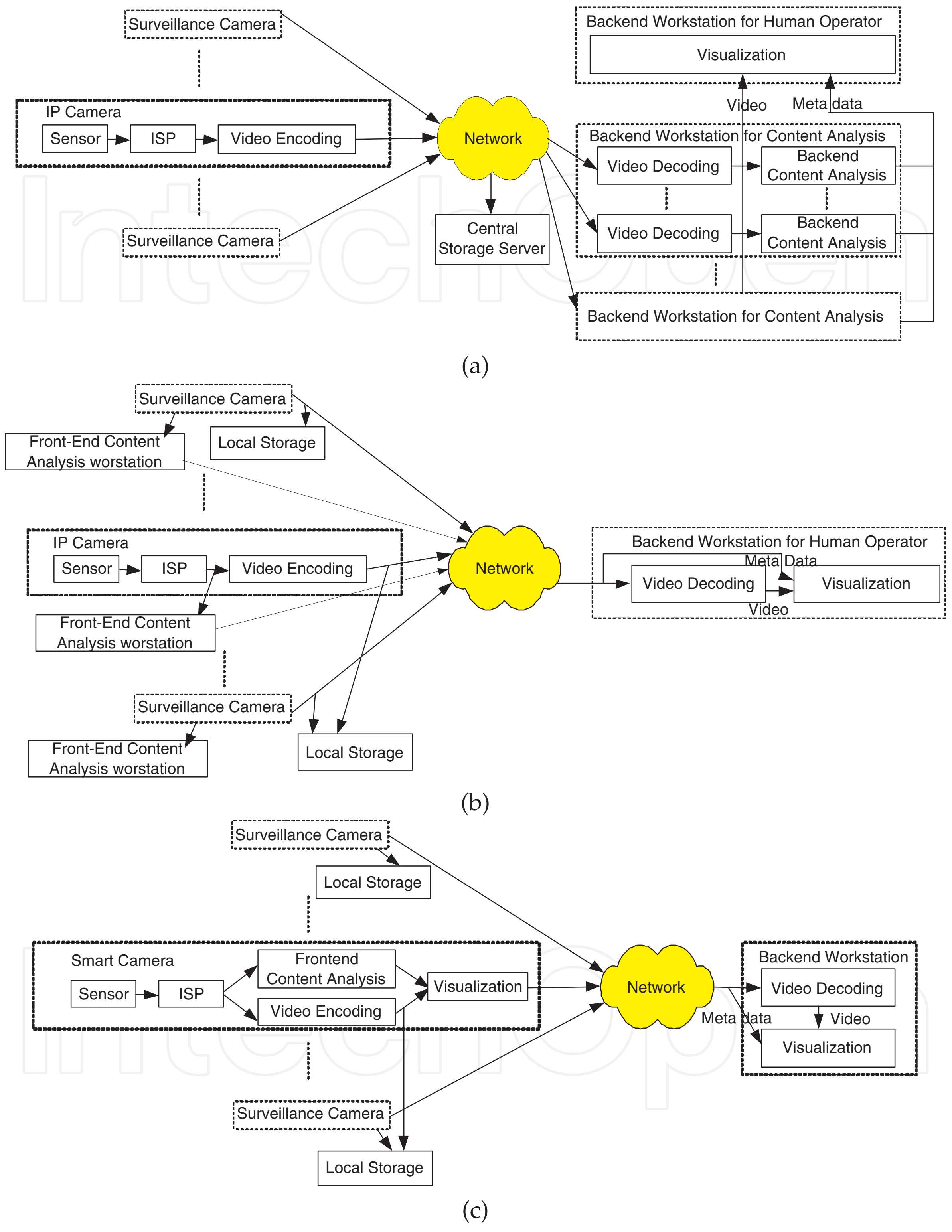


Fig. 4. Surveillance pipelines with different configurations. (a) Surveillance pipeline with a central storage server and back-end content analysis. (b) Surveillance pipeline with local storage servers and front-end content analysis workstations. (c) Next-generation surveillance pipeline with local storage servers and front-end content analysis in a smart camera SoC.

Surveillance Pipeline	Fig. 4(a)	Fig. 4(b)	Fig. 4(c)
Network Congestion[1]	More possible	Less possible	Less possible
Limitation from Storage Space[2]	More limited	Less limited	Less limited
Video Quality for Analysis[3]	Low	High	High
Event Detection Delay[4]	More possible	Less possible	Less possible
Deployment Space and Cost[5]	High	High	Low
System Scalability[6]	Lower	Middle	Highest

Table 1. Comparison between Surveillance Pipelines in Fig. 4

1. In Fig. 4(a), the video bitstreams should be transmitted to the back-end workstation through network for analysis. This will cause network congestion problem more frequently.
2. In Fig. 4(a), one central storage server can not afford the storage of video from all of the cameras.
3. The quality of the video for analysis in Fig. 4(a) may be degraded due to network transmission loss, such as packet loss, and video coding distortion.
4. The event detection delay in Fig. 4(a) is due to the delay of video streaming.
5. The deployment space and cost are largely reduced with smart camera SoCs in Fig. 4(c) .
6. The highest system scalability in Fig. 4(c) is due to its lowest requirements on the network bandwidth, the processing capacity of a single storage server, and the system deployment space and cost.

and operator interaction level. In the normal level, only the data in event layer and meta data layers in Fig. 3 will be transmitted to the back-end workstations for human operators. Meta data here includes a clear, representative face for each human object, and all objects' current positions and their color features. Once an event has been detected at a front-end smart camera, the current condition level will be switched to alert level. Note that the event definition can be defined according to different application scenarios by human operators. In alert level, a small-sized video stream from the cameras that detect this event will be sent to human operators along with the meta data. The human operators can judge the true condition by inspecting the small-sized video and all the meta data. Once the human operators want to see what happens clearly, the current condition level can be switched to operator interaction level on human operators' commands at any time. In operator interaction level, the upper four layers of data, which includes a large-sized video, will be transmitted to the back-end for further checking.

Two examples are described here to show the data-size-reduction ability of the proposed visualization strategy. The sizes of the data to be transmitted for each frame are measured and compared with those of conventional IP camera surveillance systems. In the first example, the sequence Hall Monitor is tested, and the required data sizes to be transmitted are shown in Fig. 5(a). For the proposed visualization strategy, the condition level will be switched to alert level if there is any object appearing from the left door in the scene. At the begging, only the meta-data is transmitted, and almost no data needs to be transmitted. At frame number 20, these is a human object appeared from left door, so the alert level is triggered. The video of size 176x144 is transmitted to the operator. At frame number 132, when the human operator finds that the man is putting something near the wall, he requires the corresponding camera to transmit the large-sized video in 352x288 frame size for detailed inspection. On the other hand, for a conventional IP camera surveillance system, the video with frame size 352x288 is always transmitted over the network even when there is no event of interests. As shown by the dashed curve in Fig. 5(a), it is obvious that the size of transmitted data is larger. In the

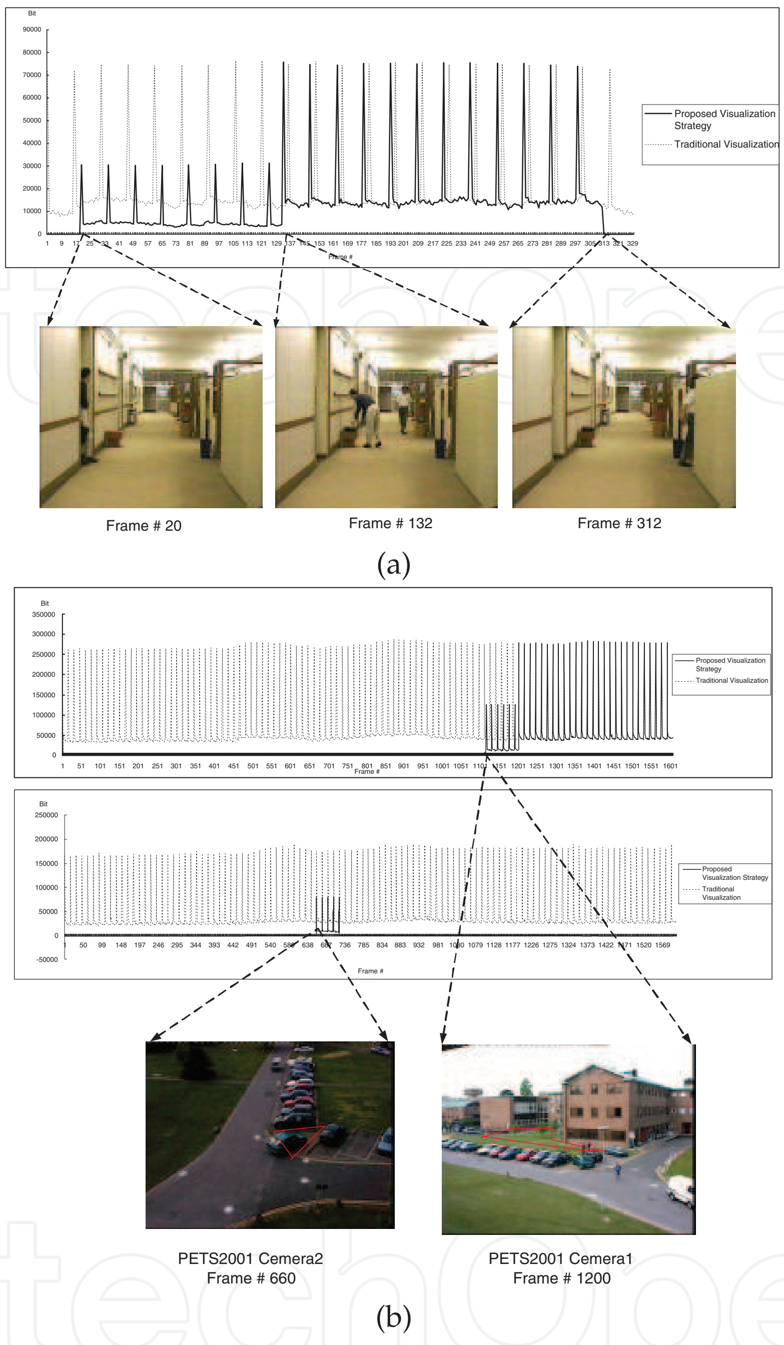


Fig. 5. Comparisons of data size to be transmitted.

second example, the PETS2001 multi-camera sequences PETS (2007) are tested as shown in Fig. 5(b). In this case, there are two cameras monitoring the same area. We set that camera 1 is used to detect illegal treading on grass, and camera 2 is used to detect illegal parking. At frame number 660, an illegal parking event is detected. At frame number 1116, a treading on grass event is detected. It can be seen that compared with conventional approach, the data size is greatly reduced in Fig. 5(b) with our proposed visualization strategy because only the semantically meaningful information for visualization is transmitted, which will greatly relief the network loading.

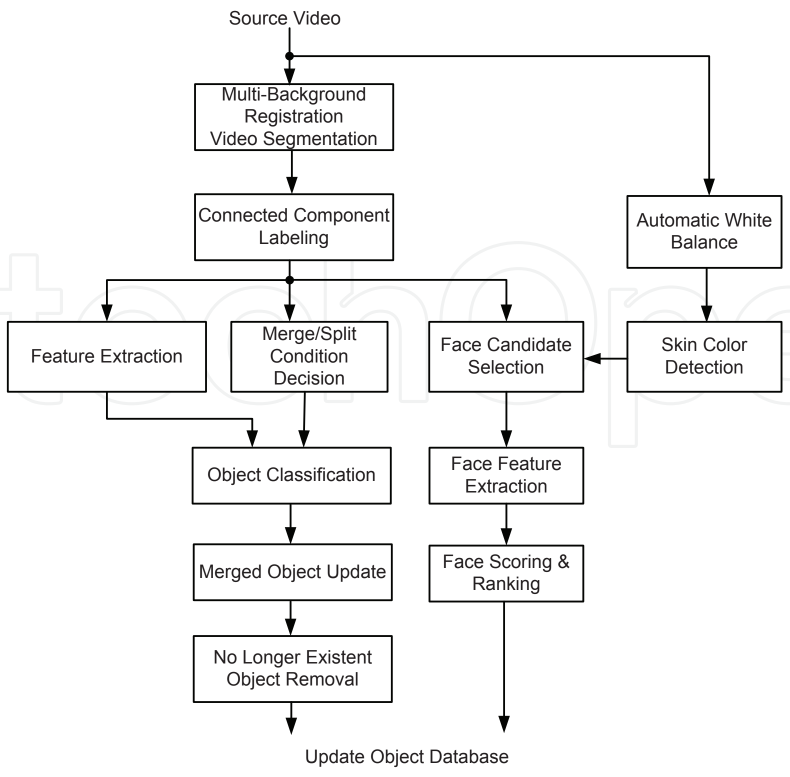


Fig. 6. The block diagram of the proposed front-end content analysis algorithm in a single smart camera.

3. Single camera operation

In this section, the algorithms proposed to be embedded in each smart camera in the next-generation surveillance pipeline for the front-end content analysis are introduced. The proposed algorithms can be shown with the block diagram in Fig. 6. There are three major components, which are video object segmentation Chan & Chien (2007), video object description and tracking Chien, Chan, Cherng & Chang (2006), and face detection and scoring Chen et al. (2007). These three components are described as follows.

3.1 Video object segmentation

As shown in Fig. 6, the video object segmentation is the first step. The segmentation algorithm employs a multiple layer background modeling technique named Multi-Background Registration. It models the background with N layers of images to contain up to N possible values of the background at every pixel Chan & Chien (2007). The object mask is produced with comparing the current frame with these background images (background subtraction), while the thresholds are decided with our threshold decision technique Chien et al. (2004). The object mask is further de-noised with morphological operations.

3.2 Video object description and tracking

In this subsection, the video object description and tracking component of our algorithm is introduced. We designed this component with the considerations below. First, the style of video object description should be close to what people are used to describe video objects, since the users would like such a description style for their inspection convenience. Second, the tracking algorithm should be capable of tracking multiple video objects with mutual merging and splitting. Finally, since there are usually similar operations in segmentation, description, and tracking, these operations should be further combined as a single system without redundant computations.

This component is based on Chien, Chan, Cherng & Chang (2006) combined with a mechanism to handle the object merging and splitting Kumar et al. (2006). As shown in Fig. 6, this component is a segmentation-and-description based video object tracking algorithm. It receives the segmentation results from our proposed *Multi-Background Registration Video Segmentation* introduced in the previous subsection. The *Connected Component Labeling* step in Fig. 6 is used to give each blob on the object mask a unique label. After that, several features/descriptors are extracted for each blob, which corresponds to the *Feature Extraction* step in Fig. 6. Our proposed Human Color Structure Descriptor (HCSD) Chien, Chan, Cherng & Chang (2006) is used as one of the descriptors to be extracted. The HCSD requires a skeletonization process to decompose the blobs. A decomposition example is shown in Fig. 7. We can see that the human object's blob is decomposed into a Body part and Limbs part in Figs. 7(d) and (e), respectively. After skeletonization decomposition, we can easily extract the color features in each individual part. Especially for human objects, they can be described in terms of their shirt color and pants color, which is usually how people describe strangers. Beside HCSD, four other features are also extracted. These are the overlapped area sizes of the blobs in the current frame with the video objects in the previous frame (Object- Blob Overlapped Area), the area of each blob (Blob Area), the center of each blob (Blob Center), and the color histogram of each blob in YUV color space (Blob Color Histogram).

On the other hand, the merging and splitting conditions are judged in *Merge/Split Condition Decision* step in Fig. 6 in advance before *Object Classi□cation*, where the correspondences between the blobs on the object mask and the video objects in the history database will be built. Here we use the reasoning method based on blob-object overlapping condition Kumar et al. (2006) to judge the merging and splitting conditions. However, we do not use the Kalman filter to predict video objects' motions Kumar et al. (2006). The overlapping conditions between the blobs in the current frame and the video objects in the history are employed instead considering that Kalman filter may fail to predict random motions.

After *Feature Extraction* and *Merge/Split Condition Decision*, the correspondences between the blobs and the video objects in the history database are built in *Object Classi□cation*. The correspondences involved with merging objects or splitting objects are built according to the merging or splitting conditions obtained from the *Merge/Split Condition Decision* step, while the correspondences involved with only single objects (non-merging and non-splitting) are built with selecting the closest object in the history database for each blob. The closest object here is obtained with the comparison based on the four features extracted previously.

In *Merged Object Update* step, the single objects split from some merged objects (judged with the *Merge/Split Condition Decision* step) can be removed from the merged object lists. Meanwhile, objects that are no longer observed for a predefined length of time will be removed from the object list for object-blob matching in the *No Longer Existing Object Removal* step.

3.3 Face detection and scoring

The face detection and scoring algorithm is based on segmentation and feature based face scoring Chen et al. (2007). Firstly, as shown in Fig. 6, the algorithm detects skin color regions Chai & Ngan (1999) on the surveillance video after automatic white balance Ramanath et al.

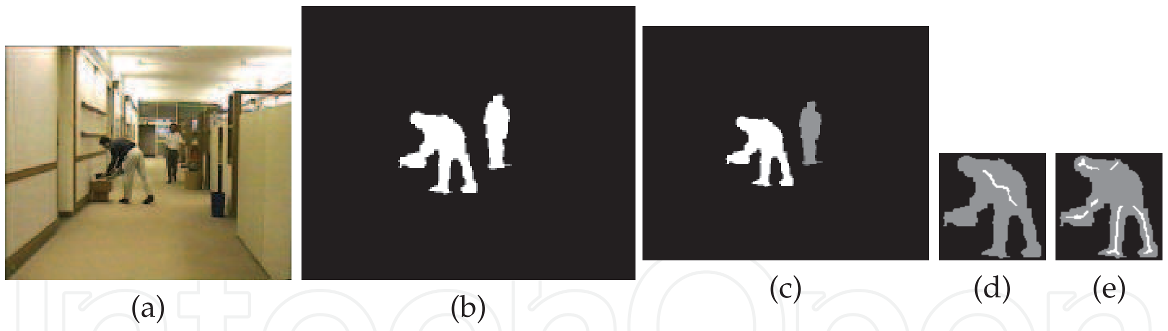


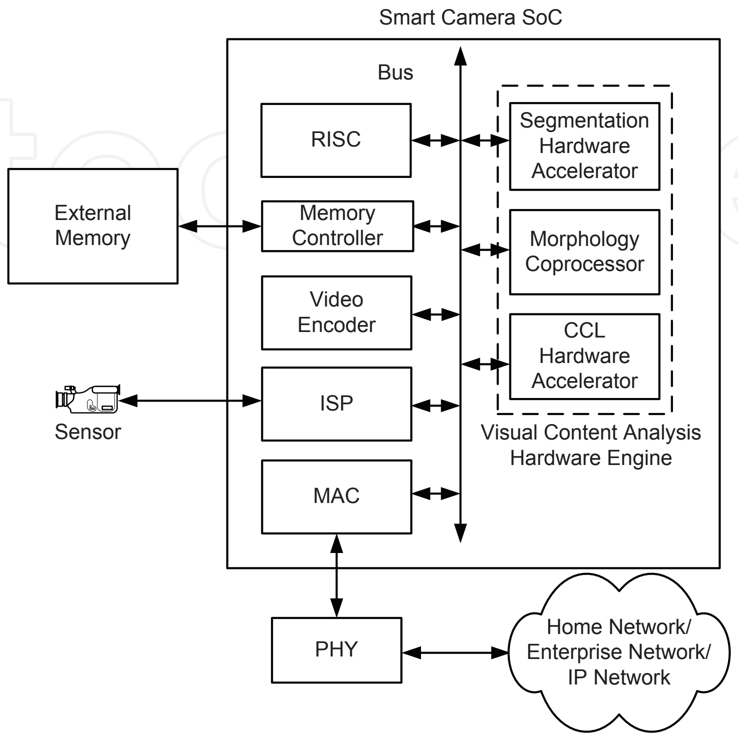
Fig. 7. An example from segmentation step to skeletonization step, where the test sequence is *Hall Monitor*. (a) Original sequence of *Hall Monitor*; (b) result after segmentation; (c) result after connected component labeling; (d) result after skeletonization: Body part; (e) result after skeletonization: Limb part.

(2005). Then, in *Face Candidate Selection* step, the possible locations of faces on the overlapped area between the skin color regions and the foreground regions in the object mask are found via a mean shift process Comaniciu et al. (2003). After finding the possible locations of faces, these face candidates are scored with four face feature scores, which are Skin Color Coverage Score, Luminance Variation Score, Circularity Measurement Score, and Eye-pixel Histogram Score Chen et al. (2007). These four scores are linearly combined to form the final score function. A neural network is employed to train the weighting of each score based on Least-mean-square (LMS) delta rule. The faces with higher final scores are better ranked and are selected as the detected faces.

4. Smart camera hardware architecture with embedded content analysis engine

Before hardware architecture design and mapping, the whole content analysis algorithm is first analyzed with execution time profiling and data flow graph. We find that there are three computationally intensive operations that dominate the computation time, which are multi-background registration video segmentation, morphological operations (to de-noise the object mask), and the connected component labeling. Based on this analysis result, the hardware architecture of the smart camera SoC is proposed as shown in Fig. 8(a). We can see that there are three hardware accelerators, *Segmentation Hardware Accelerator*, *Morphology Coprocessor*, *CCL Hardware Accelerator* in the proposed *Visual Content Analysis Hardware Engine*. These three accelerators are used to accelerate the processing of the three computation intensive operations in our profiling.

There are several design issues for these hardware content analysis accelerators. First, the operations involved in a single camera have very different requirements. For example, some of the algorithms should be implemented to have high throughput, and dedicated hardware accelerators may be a better solution for this case. Some of the algorithms are adaptive according to different situations, and programmable hardware accelerators could be used. In addition, morphology operations Serra (1982) are widely used in the proposed content analysis algorithms. Hardware sharing between different algorithms should be considered. Moreover, most of the operations involved are basically frame-level operations, which means that the next operation can be executed only when the whole frame is completely scanned or processed by the current operation. To achieve high throughput for such operations, frame-level pipeline technique should be employed, and a frame buffer is required. For



(a)

Process	TSMC 0.13μm	
Working Frequency		
CPU (ARM926EJ-S)	266MHz	
Visual Content Analysis Hardware Engine	62.5MHz	
System Bus (64-bit)	133MHz	
Hardware Cost		
	Gate Count	On-Chip Memory (Kb)
Segmentation	23,216	10.00
CCL	6,111	15.63
Morphology Coprocessor	276,480	30.00
Total Hardware Cost	305,807	55.63
Processing Speed	30 640x480 frames/s	

(b)

Fig. 8. (a) Block diagram of smart camera SoC hardware architecture with heterogeneous content analysis hardware engine. (b) System specifications and implementation results.

the requirement of large frame size, the frame buffer is not feasible to be implemented on-chip and should be located in the off-chip memory, which will introduce high memory bandwidth requirement. Furthermore, when these hardware accelerators are integrated in an SoC, bit-width mismatch may sometimes lower the performance of the hardware. That is, for the various algorithms, the data formats are quite different. Some are 8-bit, some are binary, and some are 16-bit; however, the bit-width of the system bus is fixed, which is decided to be 64-bit in this paper after system analysis. To efficiently utilize the system bus bandwidth, the hardware should be carefully designed.

For the above reasons, firstly, the visual content analysis hardware engine is proposed to be designed with heterogeneous processing units, which includes CPU, dedicated and programmable hardware accelerators here. Besides, these algorithms can be separated into special operations and morphological operations. Therefore, in our proposed visual content analysis hardware engine, there are three modules: *Segmentation Hardware Accelerator*, *Connected Component Labeling (CCL) hardware accelerator*, and *Morphology Co-Processor*. The *Segmentation Hardware Accelerator* and *CCL Hardware Accelerator* are dedicated hardware accelerators for video object segmentation and connected component labeling operation, respectively. The major design techniques employed are delay-line and partial-result-reuse technique Chien, Hsieh, Huang, Ma & Chen (2006). The *Morphology Co-Processor* is a programmable hardware accelerator for binary morphology operations, which can be used to accelerate the processes of Video Segmentation, Skeletonization, and Face Detection and Scoring Hartenstein (2001) Chan & Chien (2006a) Serra (1982). As for the bit-width mismatch problem, the design concept of subword level parallelism (SLP) Chan & Chien (2006b) is considered in our design to efficiently utilize the system bus bandwidth.

The implementation results are shown in Fig. 8(b), which demonstrates a nice trade-off between hardware cost and performance. It can achieve the processing speed of 30 VGA frames/s with a reasonable hardware cost.

5. Case study I: Surveillance system with multiple fixed cameras

The first case study is a surveillance system with multiple fixed cameras. We will introduce two key techniques for the spatial consistency labeling in multi-camera surveillance systems: homography transformation Semple & Kneebone (1979) and earth movers distance (EMD) Rubner et al. (1998). The PETS2001 multi-camera sequences PETS (2007) are employed as the test sequences.

Spatial consistency labeling (SCL) algorithm is used to find the object correspondences between different camera views Chang et al. (2008). With the assumption that the views should have a common ground plane, the algorithm for SCL is based on ground plane homography transformation Semple & Kneebone (1979) Bradshaw et al. (1997). The homography transformation matrix is defined as follows.

$$\begin{bmatrix} \lambda_i X_i \\ \lambda_i Y_i \\ \lambda_i \end{bmatrix} = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \quad (1)$$

In this equation, $\mathbf{M}$ is the ground plane warping matrix between two views. (X_i, Y_i) and (x_i, y_i) represent the corresponding ground plane positions. Homography transformation converts the coordinates of a ground point in one view to the coordinates in the other view. An example is shown in Fig. 9. Given four or more matching pairs, the ground plane warping matrix can be derived with least square errors.

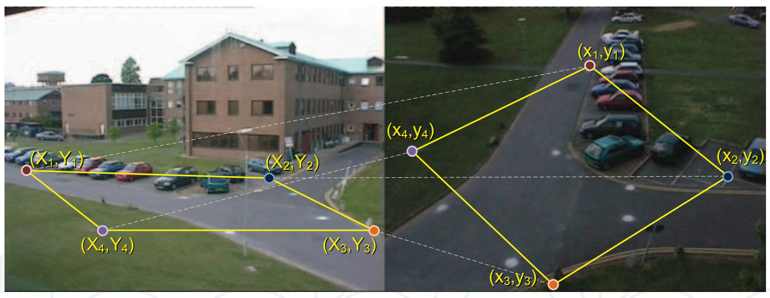


Fig. 9. Warping a ground plane from one view to another using homography transformation.

The bottom center of a bounding box of a mask is assumed to be the object ground point. However, for a mask of a person, the shadow in the mask may shift the ground point from the true position, and for a mask of a vehicle, the same object in different views may have totally different ground points. For instance, for the same car, the left rear tire is regarded as its ground point in one view, and the right rear tire may be regarded as its ground point in another view. For this issue, two concepts, earth mover's distance (EMD) Rubner et al. (1998) and trusting-former-pairs-more, are employed to prevent the generation of wrong matching pairs in this type of case.

EMD is originally used as a difference measure between two distributions. The distance measurement is modeled as a transportation problem, where a flow of goods with minimum transportation cost from suppliers to customers is required to be found and the minimum transportation cost per unit of good flow is defined as the earth mover's distance. In our matching approach, let $P = \{(\mathbf{p}_1, w_{\mathbf{p}_1}), ..., (\mathbf{p}_m, w_{\mathbf{p}_m})\}$ be the ground-point distribution with m converted coordinates of points from the first view to the second view with homography transform, where $\mathbf{p}_i$ is the converted coordinate of a point and $w_{\mathbf{p}_i}$ is the weighting of the point. Here we set the weighting to 1 since each point needs to be paired to only one point at most in the other view. Then let $Q = \{(\mathbf{q}_1, w_{\mathbf{q}_1}), ..., (\mathbf{q}_n, w_{\mathbf{q}_n})\}$ be the ground-point distribution with n points in the second view. We also let $\mathbf{D} = \{d_{ij}\}$ be the distance matrix where d_{ij} is the distance between $\mathbf{p}_i$ and $\mathbf{q}_j$. $\mathbf{F} = \{f_{ij}\}$ is the flow matrix where f_{ij} is 1 if and only if $\mathbf{p}_i$ and $\mathbf{q}_j$ are considered as a pair otherwise f_{ij} would be 0. Fig. 10 shows an example of notation representation. The objective is finding an $\mathbf{F}$ that will minimize the total cost function $C(P, Q, \mathbf{F})$

$$C(P, Q, \mathbf{F}) = \sum_{i=1}^{m} \sum_{j=1}^{n} d_{ij} f_{ij}, \tag{2}$$

and then the EMD(P, Q) is calculated as

$$\text{EMD}(P, Q) = \frac{C(P, Q, \mathbf{F})}{\sum_i^m \sum_j^n f_{ij}}. \tag{3}$$

EMD tries to find the matching pairs that will minimize the overall cost. It means that if some ground point deviations exist, EMD still works correctly because it finds pairs with a minimum global cost but does not pursue a minimum matching distance one point by one point.

The other concept, trusting-former-pairs-more, is performed for stable and reliable consistency labeling. To work with satisfactory performance, we have to make an assumption that a new incoming object should keep its distance from other objects at the start. In this case,

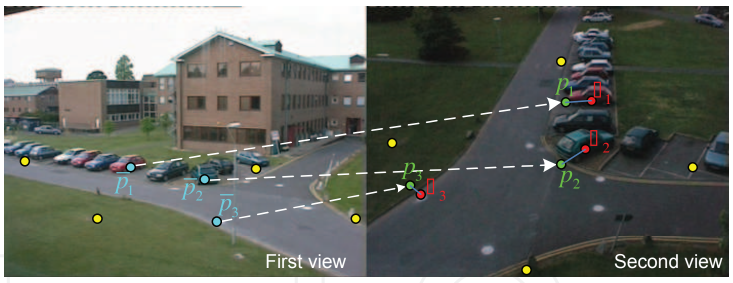


Fig. 10. Illustration of earth mover's distance. The ground point addresses $\bar{p}_i$ in the first view are converted to the addresses $\mathbf{p}_i$ in the second view. $\mathbf{q}_j$ are the ground point addresses in the second view. Yellow points are referenced points for generating a homography transformation matrix.

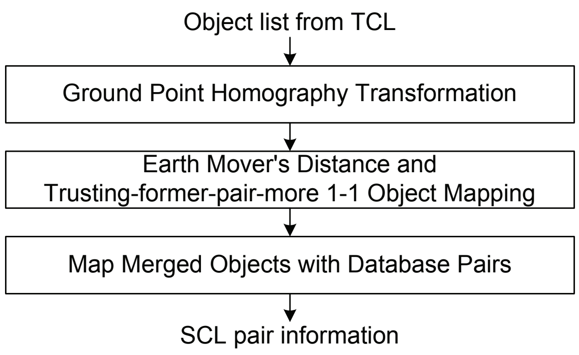

Object list from TCL
Ground Point Homography Transformation
Earth Mover's Distance and Trusting-former-pair-more 1-1 Object Mapping
Map Merged Objects with Database Pairs
SCL pair information

Fig. 11. Spatial consistency labeling flow.

if a new mask starts to appear in two cameras, the corresponding mask found at this time has higher confidence than the other corresponding masks found in the future, since the mask has relatively less occluding problems, less interactions with other masks, and fewer matching choices. Therefore, even if it has a large ground point deviation, the matching algorithm can still produce a correct pair. The pair still has chances to be renewed if the absolute point distance goes too far. The large absolute point distance reveals that a wrong pair has been generated, and the pair should be discarded from the database.

The overall flow of SCL is shown in Fig. 11. The database generated by single camera tracking, named as Temporal Consistent Labeling (TCL), is used as the input for SCL. The database provides the ground point information of single objects to EMD object matching. The concept of trusting-former-pairs-more is realized in the *one-to-one object mapping*. Note that the merged object masks are not matching in this stage because the ground point of a merged object is meaningless. Finally, the merged objects get the pair information from every single object before they are merged. This stage depends totally on the pairs in the history database.

One SCL result is shown in Fig. 12. Blue, green, red, and yellow grid points in two views are the given match points for homography matrix generation. The bounding boxes of objects with same color indicate they are matched pairs. A merged object will present alternately all color tags come from single objects paired before they are merged. Fig. 13 shows two cases that the pairs are renewed when their distances are larger than the predefined outlier distance. Fig. 13(a) shows that a man is leaving a car, which lead to an object splitting case. Fig. 13(b) shows that a car enters the left view but it is merged with the bike at the boundary. These errors are fixed at the SCL stage. In order to present the objective results of SCL, the concepts in multi-object tracking evaluation Smith et al. (2005) are employed and modified for SCL here. We define the ratio of the false pair number to the total ground true pair number in

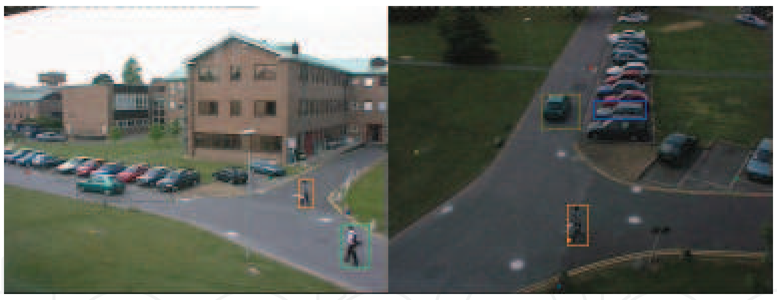


Fig. 12. Spatial consistency labeling result. The merged object in left view contains green, red, and blue tag which represents the driver, the green car, and the dark blue car in turn. If an object in the left view is a single object, its transformed ground point is shown in the right view.

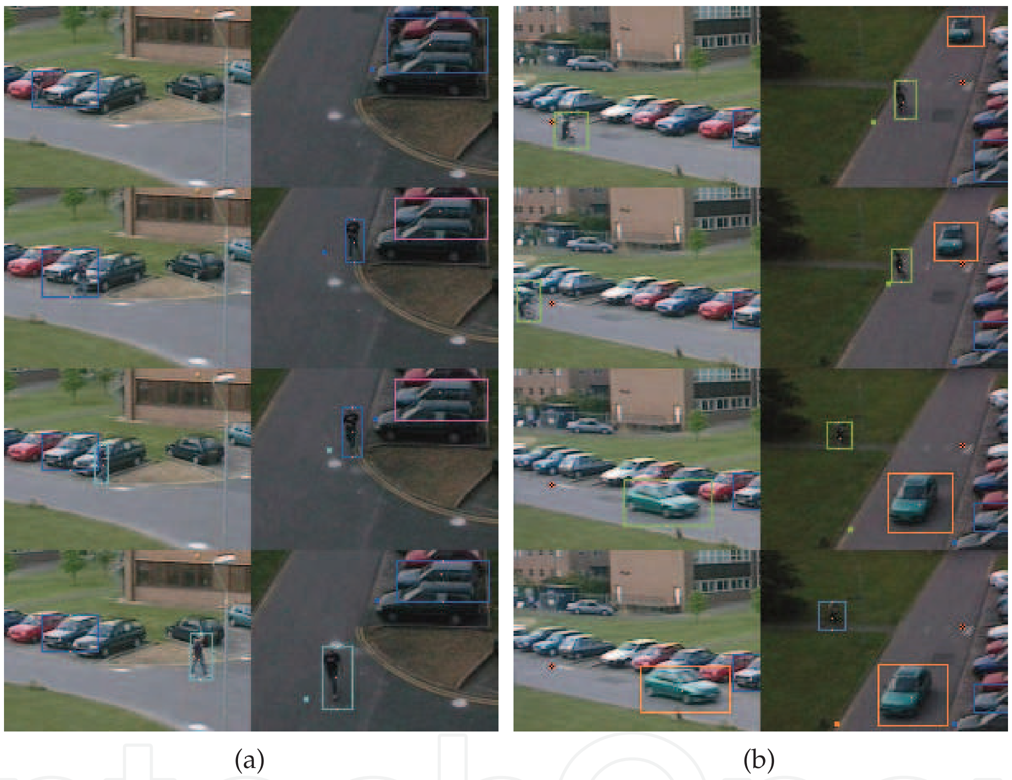


Fig. 13. Fig. 13(a) shows that a man is leaving a car. The system has no prior information about the man in the car and the car is considered as a single object. Due to the warping point deviation, the car in the left view matches the man in the right view. Later, the distance between them is large enough for fixing their pairs. Fig. 13(b) shows that a car enters the left view but it is merged with the bike at the boundary. That mask is considered as same objects before and after the switching during object tracking stage. The error is fixed at the SCL stage.

an entire sequence as the averaged falsely identified pair ($\overline{FIP}$). The $\overline{FIP}$ of the PETS2001 sequence is 0.22, which means that there are 0.22 false pair per ground true pair. Most errors are generated from bad foreground masks and the situation when objects enter the view but occluded by others.

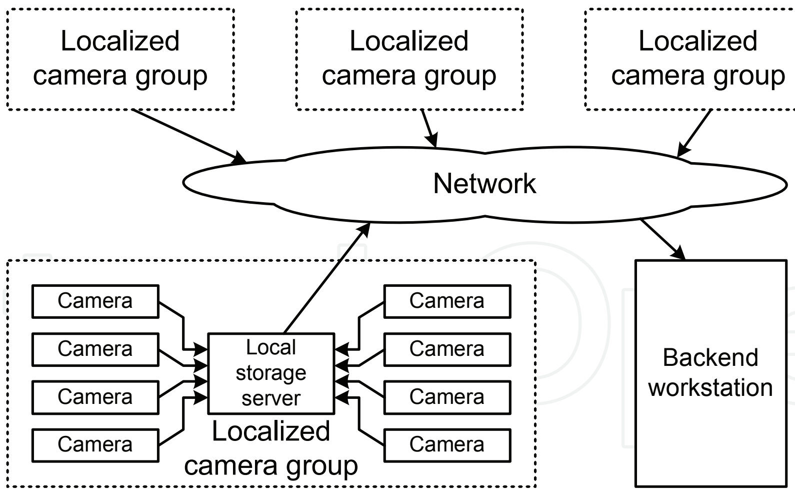


Fig. 14. Hierarchical surveillance system. In a localized area, bitstreams and object coordinates generated by each camera should be transmitted to a local storage server.

Running with on an Intel Core 2 Duo processor at 2.8 GHz, the SCL processing speed is about 4,638 fps per one pair of CIF-sized video channels. To carry out SCL, object positions from different cameras are needed to be transmitted to a server. Besides, all recorded bitstreams in a localized area should be stored for future inspection. These two reasons make the hierarchical surveillance system shown in Fig. 14 become reasonable. The local storage server stores all the data in this area, and the proposed low-complexity SCL makes it easy to be realized in this server.

6. Case study II: Cooperative surveillance system with fixed camera localization and mobile robot target tracking

The second case study is a cooperative surveillance system with fixed-camera localization and mobile-robot target tracking Chia et al. (2009). As shown in Fig. 15, the fixed cameras detect the objects with background subtraction and locate the objects on a map with homography transform with the techniques described in Section 3 and 5. At the same time, the information of the target to track, including the position and the appearance, is transmitted to the mobile robot. After breadth-first search in a map of boolean array, the mobile robot finds the target in its view by use of a stochastic scheme with the given information, then it will track the target and keep it in the robot's view wherever the intruder goes. With this system, the dead spot problem in typical surveillance systems with only fixed cameras is considered and resolved.

6.1 Motivation and introduction to the cooperative system

Recent approaches in surveillance systems typically include the use of static cameras along with the content analysis algorithms Regazzoni et al. (2001) Stauffer & Grimson (1999). The drawback is that blind spots cannot be covered, and intruders can try to avoid the fixed camera's sight, which results in less robustness for the surveillance systems. Systems employing pan-tilt-zoom (PTZ) cameras or omni-directional camera system can increase the covering range Micheloni et al. (2005) Foresti et al. (2005) Iwata et al. (2006). However, there may still exist blind spots and the covering area still depends on the cameras' positions decided in the deployment phase. Besides, several object tracking algorithms that are capable of tracking targets with a mobile camera are developed in recent years Maggio et al. (2007)

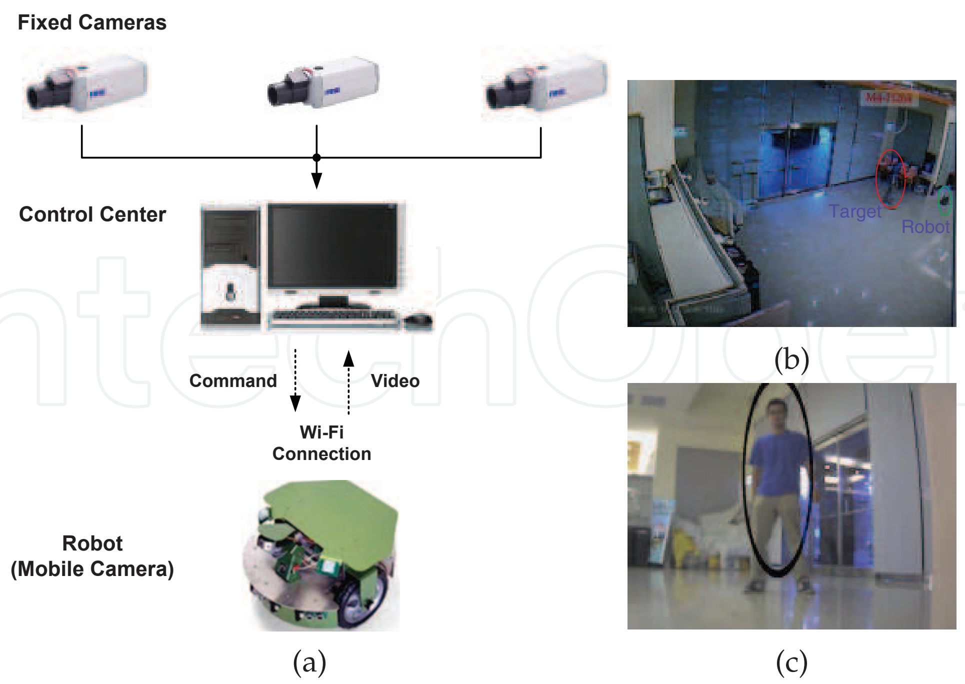


Fig. 15. (a) The proposed cooperative surveillance system. (b) Video captured by one fixed camera. (c) Video captured by the mobile robot.

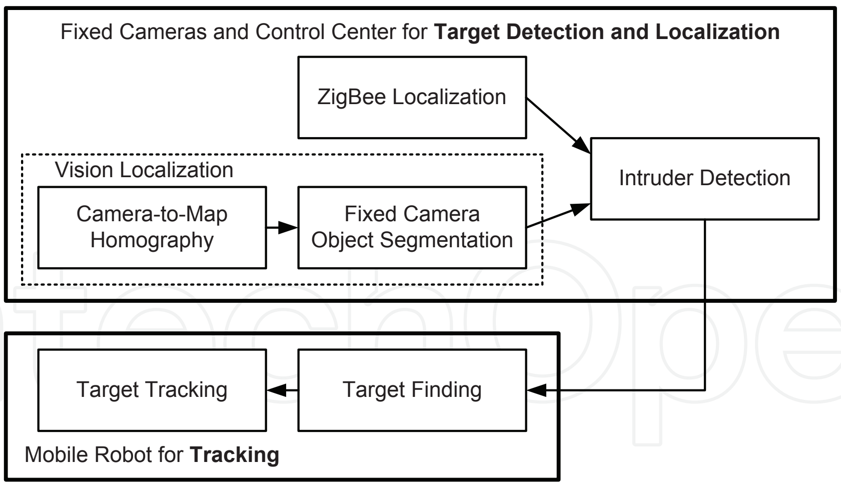
Comaniciu et al. (2003). The critical issue in the use of these algorithms is the initialization of the tracker, which is usually manually selected without automatic initialization scheme.
We now introduce a prototype system which consists of fixed cameras and a mobile robot as shown in Fig. 15 and the overall operation flowchart is shown in Fig. 16. In this system, we propose a cooperation scheme between ZigBee localization system, fixed cameras and a mobile robot. The fixed cameras can do object detection and feature extraction automatically as described in Section 3. Then the object is localized on a map of the environment via homographic relations Bradshaw et al. (1997) between the fixed cameras and a global map, which is constructed in the camera calibration phase. After these fixed camera operations, the information of the object's location in the map and its appearance are provided to the mobile robot, in which a target finding algorithm and a target tracking algorithm are implemented. The robot will follow the target throughout the entire environment and keep it in the center of the robot's view.

6.2 Intruder detection and localization

In this section, the *Target Detection and Localization* subsystem in Fig. 16 is introduced. This subsystem integrates two localization mechanisms: *ZigBee Localization* and *Vision Localization*.

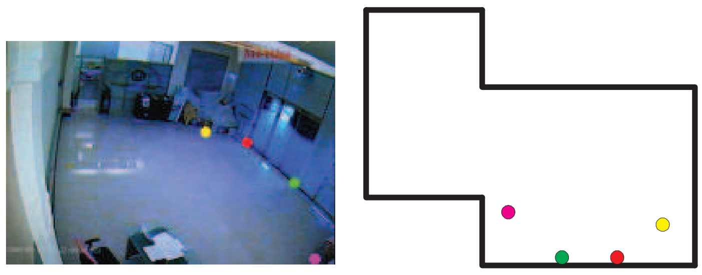
6.2.1 ZigBee localization

ZigBee is a specification for a suite of high level communication protocols using small, low-power digital radios based on the IEEE 802.15.4 standard for wireless personal area networks (WPANs). It is generally targeted at radio-frequency applications that require a low data rate, long battery life, and secure networking.
Being widely deployed in wireless monitoring applications with high-reliability and larger range, ZigBee transmitters are spread around the environment. We assume that all authorized in-comers should wear a ZigBee receiver so that their locations can always be monitored. The


Fixed Cameras and Control Center for **Target Detection and Localization**
ZigBee Localization
Intruder Detection
Vision Localization
Camera-to-Map Homography
Fixed Camera Object Segmentation
Target Tracking
Target Finding
Mobile Robot for **Tracking**

(a)

Fig. 16. the proposed cooperation scheme.



(a) (b)

Fig. 17. (a) Shot of the test environment taken by one of the fixed cameras and (b) a simple global map. The colored points are the four corresponding pairs of points chosen.

control center can receive information about the number of authorized visitors and their rough locations Lorincz & Welsh (2005).

6.2.2 Vision localization

Vision Localization is based on homography transform Bradshaw et al. (1997). After background subtraction and appropriate denoising, the segmentation result can be used to detect and localize the objects in the view of each fixed camera. Here, as described in Section 5, the bottom-centroid of each segmented object blob is treated as the object's location in the view of each fixed camera. In order to localize the object in a global coordinate system, a homography transform is employed to build the correspondences between the coordinates of fixed camera views and the coordinate of a global map, as shown in Fig. 17(b). The vision localization gives location information about all objects, including authorized and unauthorized, when they are detected with background subtraction.

6.2.3 Integration for identification

Identification (Intruder detection) can be easily done by comparing the results from *ZigBee Localization* in Section 6.2.1 and those from *Vision Localization* in Section 6.2.2. From the *ZigBee Localization*, the system receives information about the number of authorized visitors and their rough locations. At the same time, fixed cameras can find the objects (people), including authorized and unauthorized, and locate them in terms of global coordinates. Intruders, who have no authority to come in, are then detected by comparing the object information from the fixed cameras with those from the ZigBee system.

6.3 Mobile robot for tracking

From Section 6.2, the coordinate of the intruder can be inferred. A template of the intruder can also be obtained from the segmentation results in fixed cameras. Information about the object's location and appearance will then be transmitted immediately to the mobile robot to start tracking. In this section, we will introduce our *Tracking* subsystem in Fig. 16.

6.3.1 Target modeling and similarity measurement

Since the camera on the robot may be different from the fixed cameras, specific camera calibration techniques, including those for cameras with different lighting conditions and orientations, are useful for the processing and analysis Haralick & Shapiro (1992). The template of the intruder will be modeled with a color histogram, which is a viewing-angle-invariant feature. The object is represented by an ellipse. The sample points (pixels) of the model image are denoted by x_i and $h(x_i)$, where x_i is the 2D coordinates and $h(x_i)$ is the corresponding color bin index of the histogram. The number of color bin indexes used is denoted as β. The object's color histogram is constructed as follows.

$$p(u_j) = \sum_{i=1}^{I} k(\| \frac{x_i - c}{\sigma} \|) \delta[h(x_i) - u_j], 0 \le j \le \beta \tag{4}$$

where I is the number of pixels in the region, u_j is the color bin index in the histogram. σ is the bandwidth in the spatial space and c is the center of this object. δ is the Kronecker delta function. To increase the reliability of the color distribution, smaller weights are assigned to pixels farther away from the center (denoted as c), which are more likely to belong to the background. The chosen weighting function here is the Epanechnikov kernel : $k(u) = \frac{3}{4}(1 - u^2), |u| \le 1$.

Here Bhattacharyya coefficient is adopted to measure the distribution similarity as shown with the following equations Comaniciu et al. (2003).

$$B(I_x, I_y) = \sqrt{1 - \rho(p_x p_y)} \tag{5}$$

where function ρ is defined as

$$\rho(p_x, p_y) = \int \sqrt{p_x(u) p_y(u)} du \tag{6}$$

6.3.2 Target finding with mobile robot

The mobile robot then receives information about the intruder, including its coordinate and appearance (color distribution). The initial location and the initial direction of the robot can be decided according to different environments in advance with respect to any target location. The mobile robot can go to any location, including the initial location, by breadth-first search with a 2D boolean array, in which the array element stores 1 or 0 indicating whether the location is reachable or not Cormen et al. (2001). After arriving at the initial location and turning to the initial direction with the help of a compass module, the robot finds the target within its view with a stochastic scheme. First, different hypotheses are made randomly about the target's location and size. Each hypothesis is again represented by an ellipse, with randomly chosen center, and the ratio of the two axis is fixed into 3.5:1 (which approximately stands for a person's ratio in height and width). The Bhattacharyya distance between the color histogram of each hypothesis and that of the target model is then calculated, and those hyposthsis with smaller disntances will be selected. Finally, the estimated target's location and size can be derived with the average of these selected hypothesis.

6.3.3 Target tracking with mobile robot

Particle filter with color-based features is employed for target tracking with mobile robot. Particle filter, also known as Sequence Monte Carlo method or Sampling-Importance-Resampling (SIR) filter, is a state estimation technique based on simulation Ristic et al. (2004) Nummiaro et al. (2003). Here, the state of target is described by the center of the ellipse and a scaling factor representing the length of axis, since the ratio of the two axes is fixed.

The idea of particle filter is to evaluate the probability of all the particles and thus estimate the location of our target. We use a particle sample set $S = \{s^{(n)}|n = 1 \ldots N\}$, each sample s is a hypothetical state of the target.

After successfully locating the target according to Section 6.3.2, we can construct a sample set with all the samples equivalent to the target just found and then start evolution. Evolution of the particles is described by propagating each particle according to a Gaussian noise added to its center. Note that, many previous approaches propagated the particles according to a system model including moving direction and speed; however, it is not considered in our case since the tracker robot is also moving

After the particle propagation, weighting of the sample set can be computed by estimating the Bhattacharyya coefficients. We would give larger weights to samples whose color distributions are more similar to the target model. The weighting of each sample is given as follows:

$$\pi'^{(n)} = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1-\rho(p_{(n)},q)}{2\sigma^2}} \tag{7}$$

Normalizing $\pi'^{(n)}$ with the following equation, we obtain $\pi^{(n)}$.

$$\pi^{(n)} = \pi'^{(n)} / \sum \pi'^{(n)} \tag{8}$$

We can then estimate the location of our target as:

$$E[S] = \sum_{n=1}^{N} \pi^{(n)} s^{(n)} \tag{9}$$

In the last step, namely resampling, samples with higher weights will be reproduced more times than the samples with lower weights Ristic et al. (2004). When the next frame comes, the whole process can be repeated again to continue tracking.

After locating the target in each frame, the robot will judge if the target is in the left-side or the right-side of its view in order to decide which way it should turn. At the same time, it make a decision on going forward or backward according to the change in the target's scale, for example, going forward if the target's scale becomes smaller and going backward if the target's scale becomes bigger.

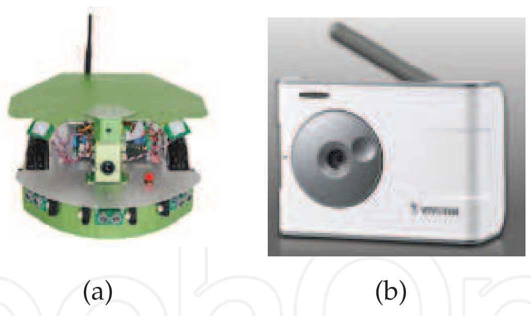


Fig. 18. (a) Dr.Robot X80; (b) Vivotek WLAN Network Camera IP7137.

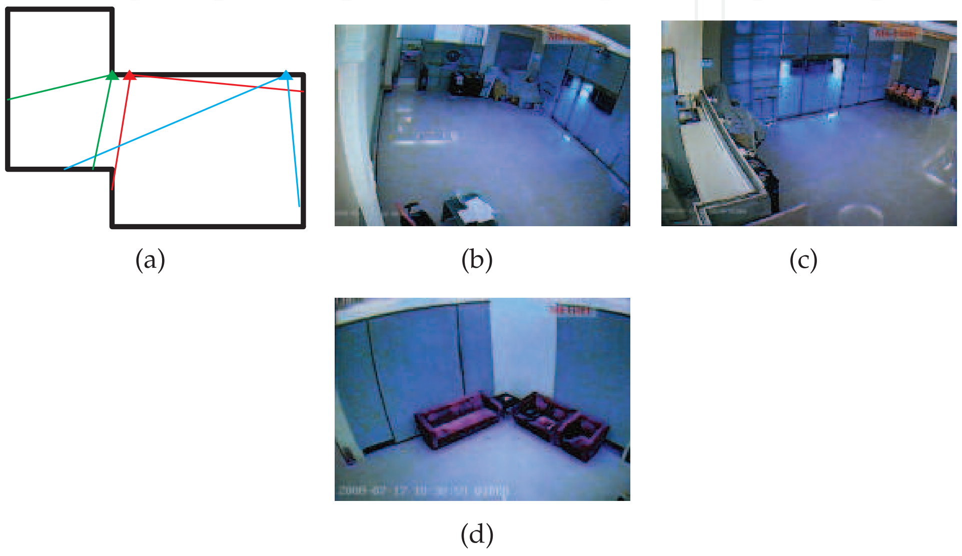


Fig. 19. The environment of Ming-Da Building: (a) A simple map of the environment with the three fixed cameras and their view range marked; (b) the image of camera colored in red; (c) the image of camera colored in blue; (d) the image of camera colored in green.

6.4 System implementation and experiment

6.4.1 System implementation

The control center is a PC with an Intel Core 2 Quad 2.4GHz CPU (1066MHz FSB), and all of the processing tasks are implemented with C♯ in Visual Studio 2005. The mobile robot we used is Dr.Robot X80. Instead of the using the built-in camera of the robot, Vivotek Network Camera IP 7137, a wireless camera with video streaming function, is equiped on the robot to provides the high-qulity video for analysis. The robot and the camera are shown in Fig.18. The overall system can track the target with the robot at 3–5 fps while the code is not optimized. Our test environment is the Technology Exhibition Center at the Ming-Da Building 1st floor of our university with three fixed cameras. The respective views of these three cameras are shown in Fig.19.

6.4.2 Experimental result

In this section, we present some results of this cooperative surveillance system. In the first place, two different scenarios are setup for testing this system. The first one (Fig.20) is that an intruder comes in and goes into the blind spots that fixed cameras cannot cover. The second scenario (Fig.21) is that an intruder is going out of the building, where the fixed

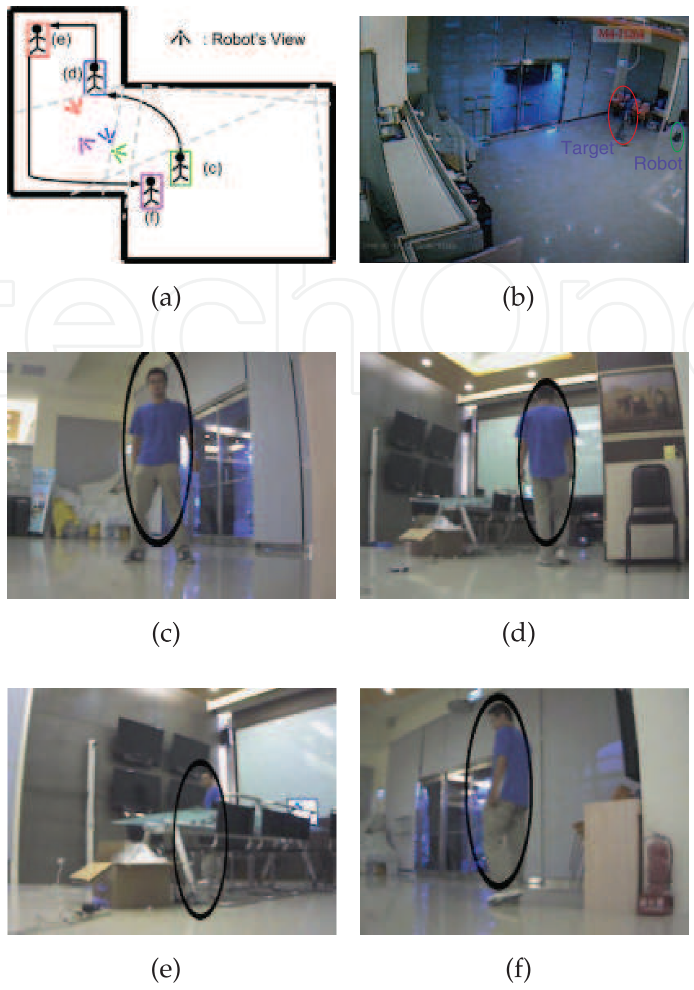
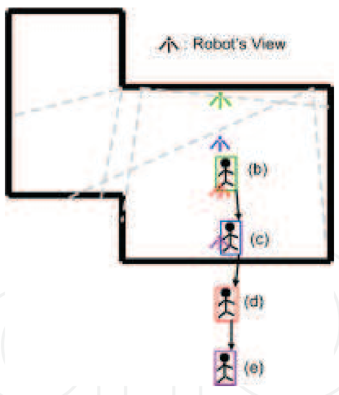


Fig. 20. Scenario one — blind spot experiment: in image (d) and (e), the intruder goes into a blind spot where the fixed cameras cannot see.

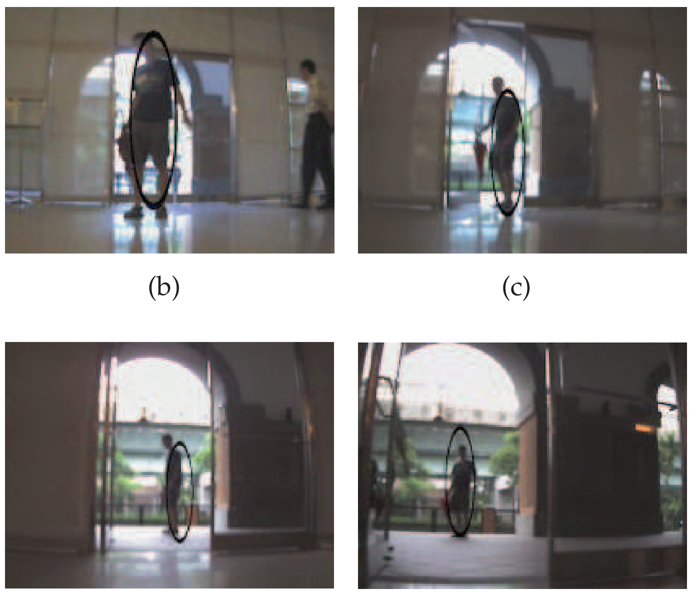
camera obviously cannot cover. In these two scenarios, our system have shown robustness in successfully locating and tracking the intruder.
In Fig.20 and Fig.21, each figure contains one map with four images. The map shows the intruder and the robot's route, where the small black man represents the intruder and a mark (viewpoint) represents the robot and its viewing direction. The four images are taken from the moving camera (i.e. the robot's view), and in each image, the target being tracked is shown with an ellipse surrounding it. In the map, the position of the green mark (robot) and the green-framed small man (target) shows where the first image is taken, while blue for the second image, red for the third image, and purple for the last image.

7. Conclusion

In this chapter, we discuss the data abstraction hierarchy and the system configuration of the next-generation surveillance systems. A conclusion has been made that each camera should be embedded with content analysis ability to become a smart camera instead of just an IP camera. The requirements of network condition, data storage, deployment space and cost are largely reduced, and even the content analysis accuracy can be enhanced with better input video quality. A simple example of smart camera SoC for the smart surveillance camera has



(a)



(b) (c)

(d) (e)

Fig. 21. Scenario two — out of building experiment, in image (d) and (e), the intruder escaped out of the door where the fixed camera cannot continue tracking.

been shown as well to show the feasiblity of the proposed concept. Finally, we provide two examples of cooperative surveillance systems, one is composed of multiple fixed cameras to jointly track objects across different camera views, and the other example is a cooperative surveillance system composed of fixed cameras and a mobile robot to resolve the blind-spot problem and the track initialization problem.
To construct robust and efficient next-generation smart surveillance systems, it is suggested that more robust content analysis algorithms should be developed. The hard conditions, such as occlusions and bad lighting conditions, should be considered and handled. Real-time performance is critical for visual surveillance camera network. More flexible programmable hardware accelerators should be designed and proposed in the future, especially for supporting more complex algorithms, such as particle filter for object tracking. Moreover, the cooperations between different cameras and even between different modalities, such as video, audio and wireless localization, should be a good way to further enhanced the performance

and ability of futher systems. All the above-mentioned issues or topics would be interesting research directions for the researchers in this area.

8. Acknowledgements

The authors would like to thank Prof. Liang-Gee Chen, Dr. Jing-Ying Chang, and Dr. Tse-Wei Chen in National Taiwan University for discussion and information sharing. This work is partially supported by National Science Council, Taiwan (R.O.C.) under Grant NSC97-2220-E-002-012 and NSC97-2221-E-002-243-MY3. This work is also partially supported by VIVOTEK Inc. EDA tools are supported by Chip Implementation Center (CIC).

9. References

Bradshaw, K., Reid, I. & Murray, D. (1997). The active recovery of 3d motion trajectories and their use in prediction, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19(3): 219–234.

Cavallaro, A., Steiger, O. & Ebrahimi, T. (2005). Tracking video objects in cluttered background, *IEEE Trans. Circuits Syst. Video Technol.* 15(4): 575–584.

Chai, D. & Ngan, K. N. (1999). Face segmentation using skin-color map in videophone applications, *IEEE Trans. Circuits Syst. Video Technol.* 9(4).

Chan, W.-K. & Chien, S.-Y. (2006a). High performance low cost video analysis core for smart camera chips in distributed surveillance network, *Proc. IEEE Multimedia Signal Processing Workshop*, pp. 170 – 175.

Chan, W.-K. & Chien, S.-Y. (2006b). Subword parallel architecture for connected component labeling and morphological operations, *Proc. IEEE Asia Paci□c Conference on Circuits and Systems*, pp. 936 – 939.

Chan, W.-K. & Chien, S.-Y. (2007). Real-time memory-efficient video object segmentation in dynamic background with multi-background registration technique, *Proc. IEEE Multimedia Signal Processing Workshop*.

Chang, J.-Y., Wang, T.-H., Chien, S.-Y. & Chen, L.-G. (2008). Spatial-temporal consistent labeling for multi-camera multi-object surveillance systems, *Proc. IEEE International Symposium on Circuits and Systems(ISCAS'08)*, pp. 3530–3533.

Chen, T.-W., Chan, W.-K. & Chien, S.-Y. (2007). Efficient face detection with segmentation and feature-based face scoring in surveillance systems, *Proc. IEEE Multimedia Signal Processing Workshop*.

Chia, C.-C., Chan, W.-K. & Chien, S.-Y. (2009). Cooperative surveillance system with fixed camera object localization and mobile robot target tracking, *Proc. Paci□c Rim Symposium on Advances in Image and Video Technology*, pp. 886–897.

Chien, S.-Y., Chan, W.-K., Cherng, D.-C. & Chang, J.-Y. (2006). human object tracking algorithm with human color structure descriptor for video surveillance systems, *Proc. IEEE International Conference on Multimedia and Expo*, pp. 2097 – 2100.

Chien, S.-Y., Hsieh, B.-Y., Huang, Y.-W., Ma, S.-Y. & Chen, L.-G. (2006). Hybrid morphology processing unit architecture for moving object segmentation systems, *Journal of VLSI Signal Processing* 42(3): 241-255.

Chien, S.-Y., Huang, Y.-W., Hsieh, B.-Y., Ma, S.-Y. & Chen, L.-G. (2004). Fast video segmentation algorithm with shadow cancellation, global motion compensation, and adaptive threshold techniques, *IEEE Trans. Multimedia* 6(5): 732–748.

Comaniciu, D., Ramesh, V. & Meer, P. (2003). Kernel-based object tracking, *IEEE Trans. Pattern Anal. Machine Intell.* 25(5): 564–577.

Cormen, T. H., Leiserson, C. E., Rivest, R. L. & Stein, C. (2001). *Introduction to Algorithms, 2nd Ed.*, McGraw Hill/The MIT Press.

Elgammal, A., Harwood, D. & Davis, L. (2000). Non-parametric model for background subtraction, *Proc. European Conference on Computer Vision*, pp. 751–767.

Foresti, G. L. & Regazzoni, C. S. (2001). Video processing and communications in real-time surveillance systems, *J. Real-Time Imaging* 7(3).

Foresti, G., Micheloni, C., Snidaro, L., Remagnino, P. & Ellis, T. (2005). Active video-based surveillance systems: the low-level image and video processing techniques needed for implementation, *IEEE Signal Processing Mag.* 22(2): 25–37.

Haralick, R. M. & Shapiro, L. G. (1992). *Computer and Robot Vision*, Addison Wesley, Reading, MA.

Hartenstein, R. (2001). Coarse grain reconfigurable architectures, *Proc. Asia and South Paci□c Design Automation Conference*, pp. 564–569.

Hu, W., Tan, T., Wang, L. & Maybank, S. (2004). A survey on visual surveillance of object motion and behaviors, *IEEE Trans. Syst., Man, Cybern.* 34(3): 334 – 352.

Iwata, K., Satoh, Y., Yoda, I. & Sakaue, K. (2006). Hybrid camera surveillance system by using stereo omni-directional system and robust human detection, *Lecture Notes in Computer Science, Advances in Image and Video Technology, IEEE Paci□c-Rim Symposium on Image and Video Technology (PSIVT'06)*.

Kumar, P., Ranganath, S., Sengupta, K. & Huang, W. (2006). Cooperative multi-target tracking with efficient split and merge handling, *IEEE Trans. Circuits Syst. Video Technol.* 16(12): 1477–1490.

Lorincz, K. & Welsh, M. (2005). MoteTrack: A robust, decentralized approach to RF-based location tracking, *Proceedings of the International Workshop on Location and Context-Awareness (LoCA 2005) at Pervasive*.

Maggio, E., Smerladi, F. & Cavallaro, A. (2007a). Adaptive multifeature tracking in a particle filtering framework, *IEEE Trans. Circuits Syst. Video Technol.* 17(10): 1348–1359.

Micheloni, C., Foresti, G. & Snidaro, L. (2005). A network of cooperative cameras for visual-surveillance, *IEE Proc. on Visual, Image and Signal Processing* 152(2): 205–212.

Mozef, E., Weber, S., Jaber, J. & Tisserand, E. (2001). Urban surveillance systems: From the laboratory to the commercial world, *Proc. IEEE* 89(10).

Nummiaro, K., Koller-Meier, E. & Gool, L. V. (2003). An adaptive color-based particle filter, *Image and Vision Computing* 21(1): 99–110.

PETS (2007). PETS: Performance evaluation of tracking and surveillance.
URL: *http://www.cvg.cs.rdg.ac.uk/slides/pets.html*

Ramanath, R., Snyder, W. E., Yoo, Y. & Drew, M. S. (2005). Color image processing pipeline: a general survey of digital still camera processing, *IEEE Signal Processing Mag.* 22(1): 34–43.

Regazzoni, C., Ramesh, V. & Foresti, G. L. (2001b). Special issue on video communications, processing, and understanding for third generation surveillance systems, *Proc. IEEE* 89(10): 1355–1367.

Ristic, B., Arulampalam, S. & Gordon, N. (2004). *Beyond the Kalman Filter: Particle Filters for Tracking Applications*, Artech House.

Rubner, Y., Tomasi, C. & Guibas, L. J. (1998). The earth mover's distance as a metric for image retrieval, *Technical Report STAN-CS-TN-98-86*, Stanford University, CA.

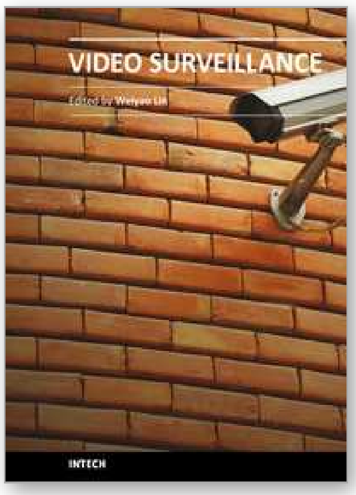
Semple, J. G. & Kneebone, G. T. (1979). *Algebraic Projective Geometry*, Oxford University Press.

Serra, J. (1982). *Image Analysis and Mathematical Morphology*, London: Academic Press.

Smith, K., Gatica-Perez, D., Odobez, J.-M. & Ba, S. (2005). Evaluating multi-object tracking, *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 36–43.

Stauffer, C. & Grimson, W. E. L. (1999). Adaptive background mixture models for real-time tracking, *Proc. IEEE International Conference on Computer Vision and Pattern Recognition*, p. 246ąV252.

Wolf, W., Ozer, B. & Lv, T. (2002). Smart cameras as embedded systems, *IEEE Computer* 35(9): 48–53.



Video Surveillance
Edited by Prof. Weiyao Lin

ISBN 978-953-307-436-8
Hard cover, 486 pages
Publisher InTech
Published online 03, February, 2011
Published in print edition February, 2011

This book presents the latest achievements and developments in the field of video surveillance. The chapters selected for this book comprise a cross-section of topics that reflect a variety of perspectives and disciplinary backgrounds. Besides the introduction of new achievements in video surveillance, this book also presents some good overviews of the state-of-the-art technologies as well as some interesting advanced topics related to video surveillance. Summing up the wide range of issues presented in the book, it can be addressed to a quite broad audience, including both academic researchers and practitioners in halls of industries interested in scheduling theory and its applications. I believe this book can provide a clear picture of the current research status in the area of video surveillance and can also encourage the development of new achievements in this field.