

# Region of Interest Extraction and Virtual Camera Control Based on Panoramic Video Capturing

Xinding Sun, Student Member, IEEE, Jonathan Foote, Don Kimber,  
B. S. Manjunath, Senior Member, IEEE

**Abstract**—We present a system for automatically extracting the region of interest and controlling virtual cameras control based on panoramic video. It targets applications such as classroom lectures and video conferencing. For capturing panoramic video, we use the FlyCam system that produces high resolution, wide-angle video by stitching video images from multiple stationary cameras. To generate conventional video, a region of interest (ROI) can be cropped from the panoramic video. We propose methods for ROI detection, tracking, and virtual camera control that work in both the uncompressed and compressed domains. The ROI is located from motion and color information in the uncompressed domain and macroblock information in the compressed domain, and tracked using a Kalman filter. This results in virtual camera control that simulates human controlled video recording. The system has no physical camera motion and the virtual camera parameters are readily available for video indexing.

**Index Terms**—Panoramic Video, Region of Interest, Virtual Camera Control, MPEG.

## I. INTRODUCTION

Distance learning and teleconferencing are becoming increasingly popular. A typical scenario is a speaker giving a lecture in a seminar or teleconference. Good speakers typically move around and gesture to enhance the lecture. Ideally, an automated video capture system will keep the speaker “in the frame.” Though a wide-angle camera may do this, it will necessarily result in a low resolution image of the distant speaker. Conversely, a mechanical pan-tilt-zoom camera can capture a good image, but the speaker may wander out of the camera view even with sophisticated automatic tracking. This paper presents a solution where a good video image is automatically produced by cropping a region of interest (ROI) containing the speaker from a high-resolution panoramic image.

The first problem considered here is the design of the video capture system. It is natural to use a panoramic camera to capture a wide-angle view of the entire scene, with the advantage that the speaker is always in the scene. The ROI image of the speaker is obtained by cropping the panoramic

video. The system is robust because the subject is always in the panoramic view. In this paper, the FlyCam [10] system generates wide-angle panoramic video. Fig. 1(a) shows an example panoramic video frame, and a corresponding ROI is shown in Fig. 1(b).

Cropping a ROI from the panoramic video results in a “virtual camera” that can be panned and zoomed. The second problem considered here is how to control the virtual camera to produce a smooth ROI sequence for viewing. In the real-time case, the panoramic video is typically in raw (uncompressed) format, and needs processing in the uncompressed domain. For applications where recorded video is viewed later, compressed domain processing is more efficient. Our system supports MPEG-1 and MPEG-2 video compression. The primary objective is to provide fast and robust virtual camera control in both the compressed and uncompressed domains.

We integrate ROI detection and tracking for virtual camera control. In the uncompressed domain [25], the ROI is detected from motion and color information. In compressed domain [24], the ROI is first located from P frame macroblock information. Then, detection results are up-sampled to obtain the ROI for the whole video stream. The ROI location is then processed using a Kalman filter to steer a virtual camera for display or recording. The Kalman filter output smoothes the ROI motion to mimic the response of a human camera operator (as discussed later). Since the panoramic camera is statically mounted, no physical camera control is needed.

The paper is organized as follows: Section II discusses the related work. Section III introduces the FlyCam system and the general system architecture. Section IV discusses the ROI detection in uncompressed and compressed domain. Section V discusses tracking using Kalman filtering. Section VI details the virtual camera control strategy. Experiments and discussions are given in section VII and section VIII, respectively.

## II. RELATED WORK

Research on automatically capturing lectures or conferences can be categorized into two areas. The first involves active camera tracking and the second involves virtual camera control based on panoramic video capturing.

Active camera control has been investigated by Zheng et al. [34] for robot control. Zobel et al. [35] design camera control method for the purpose of visual tracking. Sony’s EVI-D30 camera [22] can be used to track moving objects and has the basic functions needed for the application presented in this

Manuscript received May 20, 2003; revised October 27 2003; accepted January 25, 2004.

Xinding Sun and B. S. Manjunath are with ECE department, University of California, Santa Barbara, CA, 93106, USA. E-mail: {xdsun, manj}@ece.ucsb.edu.

Jonathan Foote and Don Kimber are with FX Palo Alto Lab, 3400 Hillview Ave, Palo Alto, CA 94304, USA. E-mail: {foote,kimber}@fxpal.com.

paper. However, in our experiment, we find this kind of steerable camera suffers from the drawback that the objects are difficult to track once they drift out of the camera's field of view. Mukhopadhyay and Smith [18] use IR beacons for tracking, which also suffer the same problem.

Chen and Williams [6] and many others have developed systems that compose existing still images into a panorama that can be dynamically viewed. Teodosio and Bender [29] have developed a system that composites successive video frames into a still panorama. Nayar [19] has created an omnidirectional digital camera using curved mirrors. A conventional camera captures the image from a parabolic mirror, resulting in a hemispherical field of view. Majumder et al. [16] use 12 video cameras and a mirror apparatus to form a spherical panoramic image using texture-mapping hardware. Swaminathan and Nayar [27] have taken a similar approach, using an array of board cameras. Instead of piecewise image warping, a table lookup system directly warps each image pixel into the composite panorama. In a more recent work by Nicolescu and Medioni [20], a camera array is used for panoramic image composition. There is also commercially available low-resolution panoramic camera used for meeting recording, for example, by Lee et al. [15]. Other commercially available systems include BeHere [3], IPIX [14], etc. In our work, we use the FlyCam [10] system to capture panoramic video. FlyCam stitches video from multiple cameras to create a high-resolution output. Other recent systems that stitch multiple camera input include RingCam [8] developed by Cutler et al. for meeting recording. While in this work we use FlyCam for panoramic video capturing, in general any kind of panoramic video can be used with our system.

Speaker tracking is needed to generate the best ROI video from the captured panoramic video. Previous person-tracking efforts date back to the early 1980s. An example is O'Rourke and Badler's [21] work on 2-D kinematic modeling. Other vision-based techniques include skin-color-based tracking [23] by Stiefelhagen et al., motion-based tracking [7] by Cutler and Turk, and shape-based tracking [2] by Baumberg and Hogg. Darrell et al. [9] integrate stereo, color, and face detection with person tracking. Wang and Brandstein [30] combine image and audio data (from a microphone array) for the purpose of face tracking. Wang and Chang [31] have developed a system that can detect a face in an MPEG video based on DCT coefficients, avoiding the expense of decompression. Their face detection rates reportedly approach 90%. Depending on the application, the tracking system can be complex, for example, Tao et al. [28] use a layered representation for multiple object tracking. As we will see in section IV, complex speaker tracking models are not needed for the specific application presented in this paper.

Since the main objective of the above systems is tracking or detection, the output of these systems is usually an object outline. Using this kind of raw tracking results to steer ROI selection usually produces objectionable jitter in the video output. Therefore, the ROI output must be processed for optimal control of the virtual camera. Examples include the 3D virtual

cinematographer by He et al. [13], 3D animation by Gleicher and Witkin [11], and fovea area view by Wei and Li [32]. Since the work here involves moving a small ROI rectangle inside a large panoramic video, our virtual camera control is equivalent to controlling a moving camera in a 2D image plane. Our main concern is to design a new method that simulates the response of a human operator, as discussed in section VI.

### III. SYSTEM ARCHITECTURE

#### A. *The FlyCam System*

Fig. 2(a) shows the FlyCam system developed at FX Palo Alto Laboratory. This system generates panoramic video from multiple adjacent cameras in real-time. Lens distortions are corrected and the images are stitched seamlessly by digital warping. Fig. 2(b) shows a modified version of the cylindrical camera that covers a 180° frontal view. A FlyCam is compact and inexpensive, using component cameras that cost around \$100 each. The system presented here uses a 180° view FlyCam as the video input device.

#### B. *General System Architecture*

Fig. 3 shows the general system architecture of the camera control system based on a 180° FlyCam. Each component camera in the FlyCam system produces NTSC video that is digitized using a frame grabber. The FlyCam software unwarps and stitches the input video streams to produce panoramic video. The panoramic video can be kept in raw format or compressed into MPEG stream for recording or viewing. The video is further processed through ROI detection, Kalman filtering, and virtual camera control. After the above processing, the output digital ROI video can be recorded or distributed, for example over the web. The core part of the system is ROI detection, Kalman tracking and virtual camera control, as discussed in the following sections.

### IV. ROI DETECTION

Many methods have been proposed for object tracking. Since our primary objective is to capture a single speaker in a panorama, complex models such as those of section II are not needed. The speaker is modeled as a point object corresponding to the centroid of the speaker's motion. The ROI output is a rectangular region of predetermined size, for example 200x200, surrounding the centroid point. Thus, ROI detection reduces to detecting the centroid of the moving part of the body.

While in some applications the panoramic video must be processed in uncompressed domain, other applications require compressed domain processing. For example, only panoramic video may be available from a server, and thus the virtual camera control must be done on the client side. Streaming video is almost always delivered in a compressed format. A straightforward solution is to uncompress and process the raw video, but it is much more efficient if the processing can be done in the compressed domain. The only difference between uncompressed and compressed video processing is our approach to ROI detection. We present two separate methods

for ROI detection depending on whether the source video is uncompressed or compressed.

#### A. Uncompressed Domain ROI Detection

1) *Feature Extraction*: Two principal features are considered for ROI detection: optical normal flow and color. The method is based on the overall confidence of motion and color change at each pixel. The confidence value is computed as a weighted sum of the color and motion information.

Given two consecutive frames, a standard measure of optical flow from pixel intensity is given by

$$I_x U + I_y V + I_t = 0 \quad (1)$$

Where  $I_x$  and  $I_y$  are the spatial derivative of image intensity in the  $x$  and  $y$  directions,  $U$  and  $V$  are the corresponding velocities, and  $I_t$  is the temporal derivative. The normal flow is typically defined as [1]:

$$V_n = \frac{I_t}{\sqrt{I_x^2 + I_y^2}} \quad (2)$$

This value is normalized to [0,1], and is used as the confidence of motion at each pixel. The normalized values are denoted by  $C_m(x, y)$ .

Besides motion, color provides important information about the scene. While any of the traditional color spaces (such as RGB, HSV, Luv, etc.) can be used, it has been observed that the HSV space is better suited for computing the color changes. Separating hue from the saturation and brightness adds robustness under most lighting variations. For example, Bradski [5] uses the distribution of the hue (H) value for tracking. The pixel-wise hue difference between two consecutive frames is computed and normalized to between 0 and 1. The normalized color confidence is denoted by  $C_c(x, y)$ .

The overall confidence value of motion and color change at each pixel is then computed as a weighted sum of  $C_c(x, y)$  and  $C_m(x, y)$ :

$$C(x, y) = w_1 C_m(x, y) + w_2 C_c(x, y) \quad (3)$$

Though weights in (3) can be fixed, a better way to combine the motion and color information is to use a spatially varying weight according to the homogeneity of the image. This can be obtained directly from the spatial derivatives of the image as shown in Fig. 4. We choose  $w_1$  to be  $Max(I_x, I_y)$ , where  $I_x$  and  $I_y$  are the normalized (0-1) spatial derivatives at  $x$  and  $y$  directions, and set  $w_2 = 1 - w_1$ . If the spatial derivatives at a given pixel are very small, (2) tends to create large errors for normal flow estimation. In this case  $w_1$  can be set to zero. The above choice of  $w_1$  is observed to work well in our experiments.  $I_x$  and  $I_y$  represent the homogeneity at a given pixel. The computation of them just involves the subtraction of pixel intensities at two neighbor pixels. Therefore, they can be reliably computed and the complexity of computation is very

low. This is ideal for real time applications. Optimal selection of the weights requires extensive data training that is not discussed here. After the confidence value at each pixel is computed, a confidence map for a given video frame is obtained. This confidence map is then used for feature tracking. Fig. 5(b) shows the confidence map of a frame from a panoramic video shown in Fig. 5(a).

2) *Centroid Detection*: Thresholding the confidence map separates the moving part of the body from non-moving region (background plus non-moving part of the body) of the scene. Fig. 5(c) shows such moving part obtained from Fig. 5(b). The white pixel is the non-moving region, and the black pixel is the moving part of the body. The centroid of the moving part of the body (and thus the ROI) can be located from the first order spatial moment of the moving part, which yields a reasonable estimate of the centroid. The moment computation is realized using the Intel Image Processing Library. Fig. 5(d) shows the manually segmented moving part. It is observed that when the speaker's clothes do not have much texture as shown in Fig. 5(a), the detected moving part tends to be located at the edges of the body. Since the body is symmetric, the detected centroid will not drift much (have much error) in the  $x$  direction but it will drift more in the  $y$  direction. Nevertheless, in general the drift is very small (shown in Table I) compared to the ROI output that is as large as 200x200. The drift in  $y$  direction will not significantly change the viewing result and this is also observed in the experiment.

#### B. Compressed Domain ROI Detection

Compressed domain video processing can be done rapidly and efficiently. While Zhang et al. [33] and many others use compressed domain features for video data segmentation and indexing, very few efforts have been made to use them for detection purposes. An example of compressed domain face detection is given in [31]. The method proposed here is based on our previous work on motion activity detection [26]. The ROI is detected using the P frames in the MPEG stream. This includes extracting P frames from the video and detecting the ROI in a P frame. The detected ROI position is then propagated to neighboring frames by temporal up-sampling. Note that even though we discuss the algorithms based on recorded video, they can be applied to real-time encoded MPEG stream the same way.

1) *MPEG Motion Compensation*: At this point, it is helpful to briefly review the MPEG motion compensation scheme [12]. Motion compensation helps reduce temporal redundancy between the frames in a video sequence. An MPEG-encoded video contains three types of frames: *I frames*, which are intra-coded, *P frames*, which are coded using forward prediction, and *B frames*, which are coded using both forward and backward prediction. Fig. 6 shows a typical MPEG sequence, which is also a group of pictures (GOP). Note that the display and transmission have different orders. For simplicity, only progressive video is discussed here, however, the proposed method also works on interlaced video if only one field of a

frame is taken for processing.

To exploit temporal redundancy, MPEG adopts macroblock motion estimation. To reduce the bit rate, some macroblocks in the P or B frames are coded using their differences from reference macroblocks. Since only the P frames are used for camera control, we ignore the others. During motion compensation, the encoder first searches for the best match of a macroblock in the neighborhood in the reference frame. If the prediction macroblock and the reference macroblock are not in the same position, motion compensation is applied before coding. When a macroblock has no motion compensation, it is referred to as a No\_MC macroblock. Generally, there are two kinds of No\_MCs: one is the No\_MC intra-coded and the other is the No\_MC inter-coded. A typical MPEG encoder contains a classifier that determines whether a macroblock is intra- or inter-coded by comparing the prediction error with the input picture elements (pels). If the mean squared error of the prediction exceeds the mean squared pel value then the macroblock is intra-coded, otherwise it is inter-coded. The No\_MC intra-coded and inter-coded scheme can be obtained correspondingly. In general, the condition for a No\_MC inter-coding can be written as:

$$\sum_{\mathbf{x}} (I_c(\mathbf{x}) - I_r(\mathbf{x}))^2 < \sigma \quad (4)$$

Where  $\mathbf{x} = (x, y)^t$  is the position of a point in the macroblock,  $\sigma$  is the threshold,  $I_c$  is the current (P type) frame, and  $I_r$  is the reference frame which is either an I frame or a P frame.

Only P frames have No\_MC inter-coded macroblocks. In the special case when the macroblock perfectly matches its reference, it is skipped and not coded at all. For illustration, the skipped macroblock is categorized as a No\_MC inter-coded macroblock as shown in Fig. 6. Note that unless skipped, No\_MC inter-coded macroblocks still take significant bandwidth for encoding and decoding.

#### 2) Detecting the ROI Centroid in P Frames:

The MPEG motion compensation scheme borrows its strategy from traditional region-based optical flow estimation, even though the motion vectors it provides are not the same as optical flow. In the case of a video where there is only one moving object, motion compensation information becomes especially important in locating the moving part of the body.

If the center of a macroblock is to represent the whole block, then a sub-sampled image of the original frame can be obtained. Since the macroblock size is 16x16, the height and width of the sub-sampled image are 1/16th of the original frame height and width respectively. If an estimation of the centroid of the ROI in the sub-sampled image is obtained, it can then be up-sampled, i.e. the estimated centroid position  $(x, y)$  can be scaled by 16 (which is fixed for MPEG video) to estimate its location in the original frame. Sub-sampling tends to create aliasing effects when there are high frequency signals in the original image. That is why traditional motion estimation methods usually filter the images with a Gaussian filter before sub-sampling.

Given a point and its neighborhood, where a point refers to the centroid of a macroblock and the neighborhood refers to the whole macroblock, motion can be estimated by minimizing the following summed square difference of intensity constancy [4]:

$$\sum_{\mathbf{x}} (I_c(\mathbf{x}) - I_r(\mathbf{x} - \mathbf{V}(\mathbf{x})))^2 \quad (5)$$

If the motion is zero, i.e.  $\mathbf{V}(\mathbf{x}) = \mathbf{0}$ , then we can find that

$\sum_{\mathbf{x}} (I_c(\mathbf{x}) - I_r(\mathbf{x}))^2$  is to be minimized. A simple comparison shows that it achieves the same objective as (4). Therefore No\_MC inter-coded macroblocks can be used to represent the non-moving region of the scene, which has no motion.

In a panoramic video scene, the ROI should contain the region with motion. In the compressed domain, the non-moving region is first detected using the macroblock motion information. The ROI is then detected by taking the complement of the non-moving region. Fig. 5(a) shows an example of a frame from a panoramic video, and it is also a P frame. Fig. 5(e) shows the ROI detection results based on No\_MC coding information, where the white macroblock is the non-moving region, and the black macroblock is the moving part of the body. Since the body region is assumed to be connected, a median filter improves the detection by eliminating unconnected macroblocks. After detecting the moving part, the object's centroid can be easily located in the sub-sampled image domain. The location is then scaled by 16 times to estimate the ROI centroid in the original video frame. In theory, when the speaker's clothes do not contain much texture, the compressed-domain centroid estimate will be close to that of the uncompressed domain. However, as shown in Fig. 5 after spatial up-sampling to the original video size, we find the detected motion region is larger than that from the uncompressed domain. Motion detection is generally more robust for macroblocks than for pixels, because the larger macroblocks tend to average out noise.

The usual 29.97 frames per second video rate is higher than is necessary for ROI processing, so temporal sub-sampling can reduce the necessary computation. Fig. 8(a-d) show four consecutive frames from a seminar room video. Note that the frame-to-frame motion of the speaker is quite small. The ROI centroid moves only several pixels per frame. Therefore, there is no noticeable difference if the centroid of Fig. 8.a is applied to the following frames (note that the original ROI is shifted 40 pixels upward in the picture to center it on the upper body).

For efficiency, it is reasonable to use the sub-sampled video consisting of only P frames. After the ROI is detected in each P frame, it is then up-sampled to obtain ROI positions of neighboring I and B frames in the original video sequence. ROIs in the I and B frames can be determined by interpolating between the ROI locations in the two neighboring P frames. In this case, the ROI of the last P frame must be determined before the interpolation. Note that the sub-sampling is not regular, since the P frame distances are generally not uniform.

## V. TRACKING USING A KALMAN FILTER

Detecting ROI centroid coordinates is generally a noisy process. Errors and noise may come from many sources, such as sub-sampling, lighting change, etc. If the noise is assumed to be Gaussian, then it can be handled with an extended Kalman filter. The centroid has a trajectory in 2D space. The trace in the  $x$  direction can be modeled by the second-order Taylor series expansion of the form:

$$x(k+1) = x(k) + v_x(k)T + a_x(k)T^2/2 + h.o.t. \quad (6)$$

$$v_x(k+1) = v_x(k) + a_x(k)T + h.o.t. \quad (7)$$

Where  $x(k)$  is the centroid coordinate in the  $x$  direction,  $v_x(k)$  is the velocity, and  $a_x(k)$  is the acceleration,  $T$  is the time interval, and  $h.o.t.$  are higher order terms. Similarly, the same model applies to the  $y$  component of the trajectory. Combining these gives a centroid system model:

$$\mathbf{F}(k+1) = \Phi\mathbf{F}(k) + \Gamma\mathbf{w}(k) \quad (8)$$

Where  $\mathbf{F}(k) = [x(k), y(k), v_x(k), v_y(k)]^t$ ,  $y(k)$  is the centroid  $y$  coordinate,  $v_y(k)$  is the velocity, while  $\mathbf{w}(k)$  is the system Gaussian noise, representing the centroid acceleration in the  $x$  and  $y$  directions, and

$$\Phi = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \Gamma = \begin{bmatrix} \frac{T^2}{2} & 0 \\ 0 & \frac{T^2}{2} \\ T & 0 \\ 0 & T \end{bmatrix}$$

Higher order Taylor series expansions can be applied to the centroid system model, which would lead to higher model orders. However, in our experiments we found it did not appreciably improve results. Additionally, the system variables provide enough information for virtual camera control as discussed in the following section.

Since the speaker is modeled as a point, a location measurement can be modeled as:

$$\mathbf{Z}(k) = H\mathbf{F}(k) + \mathbf{n}(k) \quad (9)$$

Where  $\mathbf{Z}(k)$  is the measurement,  $\mathbf{n}(k)$  is the Gaussian measurement noise, and  $H$  is the measurement transfer function (in this case a scaling factor).

The covariance form of Kalman filtering recursively updates the prediction based on the innovation information at each step. The prediction at each update is output for further virtual camera control purposes. The predicted or estimated variable used to control the recording process is  $\hat{\mathbf{F}}(k) = [\hat{x}(k), \hat{y}(k), \hat{v}_x(k), \hat{v}_y(k)]^t$ .

## VI. VIRTUAL CAMERA CONTROL

Kalman filtering reduces most of the noise inherent in the tracking estimate, and suffices for most purposes. However, if the tracking result is used to move the ROI window directly, the

quality of the output video is often jittery. The resulting motion is less smooth than that of a physical camera, which has inertia due to its mass. Therefore, an additional filtering step is taken to produce a smoother and more pleasing video output.

The approach to virtual camera control is based on the following observation. When a speaker is motionless or moving only within a small region, an experienced camera operator usually does not move the camera (*stabilization control*). When the speaker changes his position by a large distance, the operator must move the camera to catch up with the speaker (*transition control*). After the speaker has been centered, the operator then follows further movement (*following control*). Accordingly, the virtual camera control operates in three similar regimes.

“*Stabilization control*” is based on the Kalman filter estimates of position and velocity. The initial centroid position is registered first, denoted as  $\mathbf{Y}_R(k) = [x_R(k), y_R(k)]^t$ , where  $x_R(k), y_R(k)$  correspond to its  $x$  and  $y$  coordinates. In each frame, the estimated speed and position are obtained from  $\hat{\mathbf{F}}(k)$  in the Kalman filter update calculation. If the following two conditions are satisfied, the virtual camera is fixed and the registered position is used as a position output. First, the new position must be within a specified distance of the registered position. Second, the estimated speed must be below a specified threshold in a given direction. Otherwise, the virtual camera control is changed to the “*transition*” regime. The “*stabilization control*” conditions can be formalized as:

$$\mathbf{Y}(k) = \mathbf{Y}_R(k) \quad (10)$$

$$\text{if } |\hat{x}(k) - x_R(k)| < \sigma_1, \quad |\hat{y}(k) - y_R(k)| < \sigma_2, \text{ and} \\ |\hat{v}_x(k)| < \sigma_3, \quad |\hat{v}_y(k)| < \sigma_4$$

Where  $\sigma_1, \sigma_2, \sigma_3$  and  $\sigma_4$  are thresholds, and  $\mathbf{Y}(k)$  is the ROI output.

In the “*transition*” regime, a low pass filter is used to smoothly update the virtual camera location. For this purpose, a first order IIR filter is used:

$$\mathbf{Y}(k+1) = \alpha_1\mathbf{Y}(k) + \alpha_2\hat{\mathbf{X}}(k) \quad (11)$$

Where  $\alpha_1 + \alpha_2 = 1$ ,  $\alpha_1, \alpha_2 > 0$ , and  $\hat{\mathbf{X}}(k) = [\hat{x}(k), \hat{y}(k)]^t$  is the centroid estimated from the Kalman filter, and serves as the input to the IIR filter. The virtual camera now follows  $\mathbf{Y}(k)$ , which is smoother than the Kalman filter output. It also helps to reduce the noise in the case of abrupt changes that the Kalman filter does not handle well. Experiments show that values of  $\alpha_1 = 0.8, \alpha_2 = 0.2$  give a reasonable simulation of human camera operation.

Since the IIR filter (11) tends to create delays in the output, the number of steps in the “*transition*” stage is limited. After a certain time in the “*transition*” regime, for example 0.5 seconds, the camera control is switched to the “*following*” regime. Updating the ROI position directly from the Kalman filter output realizes this objective:

$$\mathbf{Y}(k) = \hat{\mathbf{X}}(k) \quad (12)$$

Note that this is equivalent to setting  $\alpha_1 = 0, \alpha_2 = 1$ , in the IIR filter (11). Fig. 9 and Fig. 10 show the results for three kinds of camera control in uncompressed and compressed domain for a video recorded at the same frame rate. The view angle is chosen to emphasize the control process in the  $x$  direction.

The Kalman filter assumes environmental noise is Gaussian, and handles lighting change and occlusion very well. But many noises are not Gaussian. For example, the projection display and the audience both can produce constant noise in fixed regions, as seen in Fig. 1. This knowledge can be incorporated into the tracking system to improve performance, especially because the panoramic video cameras are fixed with respect to the background. Configuration parameters allow some part of these regions such as projection display to be ignored. By offering this kind of flexibility, the tracking technology can be easily adapted to different environments.

## VII. EXPERIMENTAL RESULTS

Experimental evaluation of our tracking system was performed on panoramic video taken in a seminar room during seminars and presentations. The speaker moves around at the front of the seminar room during a lecture. Panoramic video can be produced in real-time at around 15 fps. To ensure frame rate, the panoramic video was stitched off-line and stored. Five video sequences up to 30 minutes were recorded and compressed to MPEG-2 format, using the following settings: 4MBit/s, 29.97fps, 800x352 pixels/frame.

A 200-frame video sequence showing a speaker moving from right to left in the seminar room was used for testing. The moving part of the body was manually segmented for each frame, and its centroid served as ground truth for ROI detection and tracking experiments. Though precise segmentation is difficult, the precision is not absolutely critical for virtual camera control. As long as the speaker is contained in the ROI output, the manual segmentation result is good enough as a reference ground truth.

First, ROI detection is processed for P frames from the test MPEG video and the corresponding frames from the uncompressed test video. The drifts (errors) between computed centroids and those of ground truth are calculated. As shown in Table I, the drifts for uncompressed domain and compressed domain are quite close. Since ROI detection in uncompressed domain has a higher resolution than that (spatially sub-sampled using macroblocks) in compressed domain, the uncompressed domain detection performs better in the  $x$  direction. On the other hand, as discussed in section IV, ROI detection in compressed domain is more stable even though it has lower resolution. Therefore no significant difference can be seen between the performance of ROI detection for both compressed and uncompressed domain in  $y$  direction.

The drifts between computed centroids after Kalman filtering and ground truth are also calculated. Kalman filtering generally improves performance for both uncompressed domain and

compressed domain, except for the  $y$  direction in uncompressed domain. This can be explained the same way as discussed in centroid detection in uncompressed domain. Note that temporal up-sampling of P frame ROI detection result in general does not affect overall result after tracking. Since a much larger ROI window (200x200) is used as output, the average drifts and their standard deviations indicated in Table I in general ensure that the speaker is covered in the ROI output.

We note that the speaker need not be exactly at the center of the ROI video for most purposes. It is also observed that there is no single standard for when and by how much to move the virtual camera as far as the smoothness is concerned. Therefore, to benchmark the system we only determine whether the speaker is covered in the ROI video output and whether the video is smooth. The results turn out to meet the requirement in general. Two demonstration sequences showing virtual camera control results have been put on the website at: <http://vision.ece.ucsb.edu/~xdsun/data/ROI>.

Software was developed to view ROI video in both compressed and uncompressed domains. The uncompressed-domain processing software is developed based on Intel Image Processing Library. A video player was also developed based on the MPEG player distributed by the MPEG Software Simulation Group [17] to view compressed-domain output video. Fig. 10 shows a user interface to the player. ROIs of size 200x200 are displayed to judge the tracking results. Since the upper body is more important in viewing, the ROI is shifted upward by a fixed number of pixels when playing. Experiments show that after initialization, the software controls the virtual camera to follow the speaker. We tested the programs by enabling and disabling the ROI processing. In terms of computation complexity, we observe that the delay created by virtual camera control is not noticeable. Since the size of the P frame macroblock information is insignificant compared to the entire video stream, the compressed domain ROI processing can be done extremely rapidly.

## VIII. DISCUSSION

In this paper a new method is presented for determining the region of interest in a classroom scene captured by a panoramic video system. The method integrates detection, tracking and recording processes to mimic human camera control. Processing can be done efficiently in both the compressed and uncompressed domains. The entire process is fully automated and experiments show that it is sufficiently robust and rapid for real-time applications.

Provided there is only one speaker in the scene, this method can be applied to a panoramic view of up to 360° using the system shown in Fig. 2(a). Note that, even though the FlyCam system is used here for panoramic view capturing, our approach applies to panoramic video produced by other systems as well. Even though the MPEG format is discussed here, the method can be applied to other formats such as H.263 which have similar forward motion compensation schemes.

For typical lectures where the speaker remains at roughly the

same distance from the camera, zooming is not necessary. However, digital zooming could be achieved by scaling the ROI for applications, as discussed in [23] and [30]. Physically zooming panoramic cameras is not generally practical, thus the highest ROI resolution depends on the resolution of the original panorama. Zooming in at an even higher resolution is an interesting research problem.

The cropped ROI video can be streamed over the Internet as part of on-line learning software interface. It can also be sent to devices such as smart phones for wireless access. In these cases, the resolution of ROI video can be scaled as appropriate. Adapting ROI video to fit different resolutions like those mentioned above and industry standards is part of our future work.

Since the panoramic camera is stationary, the tracking information also provides indexing features for the video content. Spatial data about the lecture environment can be combined with the tracking information to provide descriptive indexes about lecturer activity. Since the region of interest is isolated from other objects in the scene, the recorded result may also be useful for object based coding, such as in MPEG-4. Other research possibilities include virtual camera control for multiple objects, synchronizing the ROI output with PowerPoint slides, analyzing speaker activity, or using the ROI image as a basis for gesture tracking or face recognition.

#### ACKNOWLEDGMENT

Sun and Manjunath are supported in part by a grant from ONR #N00014-01-1-0391.

#### REFERENCES

- [1] J. L. Barron, D. J. Fleet and S. S. Beauchemin, "Systems and experiment performance of optical flow techniques," *Intern. J. Comput. Vis.*, 12:1, pp. 43-77, 1994.
- [2] A. Baumberg and D. Hogg, "An efficient method for contour tracking using active shape models," IEEE Workshop on Motion of Non-Rigid and Articulated Objects, 1994.
- [3] www.behere.com.
- [4] J. R. Bergen, P. Anandan, K. J. Hanna, and Hingorani, R., "Hierarchical model-based motion estimation," *ECCV'92*, pp.237-252, 1992.
- [5] G. R. Bradski, "Real time face and object tracking as a component of a perceptual user interface," *Proc. WACV'98*, pp. 214-19,1998.
- [6] S. Chen, and L. Williams, "View interpolation for image synthesis," *SIGGRAPH'93*, pp. 279-288, 1993.
- [7] R. Cutler and M. Turk, "View-based interpretation of real-time optical flow for gesture recognition," *IEEE Automatic Face and Gesture Recognition*, April 1998.
- [8] R. Cutler, et al. "Distributed meetings: a meeting capture and broadcasting system," in *Proc. ACM Multimedia*, 2002.
- [9] T. Darrell, G. Gordon, M. Harville, and J. Woodfill, "Integrated person tracking using stereo, color, and pattern detection," in *Proc. CVPR'98*, pp. 601-608, 1998.
- [10] J. Foote, and D. Kimber, "FlyCam: practical panoramic video and automatic camera control," *Proc. ICME'2000*, pp. 1419-1422, 2000.
- [11] M. Gleicher and A. Witkin, "Through-the-lens camera control," *Proc. SIGGRAPH*, pp.331-340, 1992.
- [12] G. Haskell, A. Puri, and A. N. Netravali, "Digital video: An introduction to MPEG 2," *Chapman and Hall*, 1997.
- [13] L. He, M. Cohen, and D. Salesin, "The virtual cinematographer: a paradigm for automatic real-time camera control and directing," *Proc. SIGGRAPH*, pp.217-224, 1996.
- [14] www.ipix.com.

- [15] D. Lee, B. Erol, J. Graham, J. J. Hull, and N. Murata, "Portable meeting recorder," in *Proc. ACM Multimedia*, 2002.
- [16] A. Majumder, W. B. Seales, M. Gopi, and H. Fuchs, " Immersive teleconferencing: a new algorithm to generate seamless panoramic video imagery," in *Proc ACM Multimedia'99*, pp.169-178, 1999.
- [17] MPEG: <http://www.mpeg.org>.
- [18] S. Mukhopadhyay, and B. Smith, "Passive capture and structuring of lectures," *Proc. ACM Multimedia*, 1999.
- [19] S. Nayar, "Catadioptric omnidirectional camera," in *Proc CVPR'99*, pp. 482-488,1999.
- [20] M. Nicolescu, and G. Medioni, "Electronic pan-tilt-zoom: a solution for intelligent room systems," in *Proc. ICME'2000*, pp. 1581-1584, 2000.
- [21] J. O'Rourke, and N.I. Badler., "Model-based image analysis of human motion using constraint propagation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2(6), pp. 522-536, 1980.
- [22] Sony EVI-D30: [www.sony.com](http://www.sony.com).
- [23] R. Stiefelhagen, J. Yang, and A. Waibel, "Modeling focus of attention for meeting indexing," in *Proc ACM multimedia'99* pp. 3-10, 1999.
- [24] X. Sun, J. Foote, D. Kimber, and B. S. Manjunath, "Panoramic video capturing and compressed domain virtual camera control", *Proc. ACM Multimedia*, pp. 329-338, 2001.
- [25] X. Sun, J. Foote, D. Kimber, and B. S. Manjunath, "Recording the region of interest from FlyCam panoramic video," *Proc. ICIP*, 2001.
- [26] X. Sun, and B. S. Manjunath, "Motion quantized alpha histogram as a video unit descriptor," *ISO/IEC/JTC1/SC29/WG11/P75,MPEG7 group*, 1999.
- [27] R. Swaminathan, and S. Nayar, "Non-metric calibration of wide-angle lenses and polycameras," *Proc CVRP'99*, pp. 413-419,1999.
- [28] H. Tao, H. S. Sawhney, R. Kumar, "Dynamic layer representation with applications to tracking," *Proc. CVPR*, 2000.
- [29] L. Teodosio, and W. Bender, "Salient video stills: content and context preserved," in *Proc. ACM Multimedia 93*, pp. 39-46, 1993.
- [30] C. Wang, and M. S. Brandstein, "A hybrid real-time face tracking system," in *Proc. ICASSP'98*, pp. 3737-3740, 1998.
- [31] H. Wang, and S-F. Chang, "A highly efficient system for face region detection in MPEG video," *IEEE Trans. Circuits and Sys. for Video Tech.*, 7(4), pp. 615-628, 1997.
- [32] J. Wei and Z. N. Li, "On active camera control and camera motion recovery with foveate wavelet transform," *IEEE PAMI* 23(8), pp. 896-903, 2001.
- [33] H. J. Zhang, C. Y. Low, and S. W. Smoliar, "Video parsing and browsing using compressed data," *Multimedia Tools and Applications*, 1(1): pp.89-111, 1995.
- [34] J. Y. Zheng, F. Kishino, Q. Chen, and S. Tsuji, "Active camera controlling for manipulation," *Proc. CVPR*, pp. 413-418,1991.
- [35] M. Zobel, J. Denzler, and H. Niemann, "Entropy based camera control for visual object tracking," *Proc. ICIP*, v3, pp.901-904, 2002.

**Xinding Sun** received the BE degree in automation from the Nantong Institute of Technology 1991, and the MS degree in automation from the Tsinghua university in 1994. He is now a PhD student of electrical and computer engineering at the University of California, Santa Barbara. He visited Microsoft Research Asia in the winter of 2003. During the summer of 2000 and 2001, he worked at FX Palo Alto Lab. He worked in Tsinghua University from 1994 to1996, and Kent Ridge Digital Labs from 1997 to 1998. His research interests include video and image indexing, computer vision, pattern recognition, and multimedia systems.

**Jonathan Foote** Jonathan Foote is a Senior Research Scientist at FX Palo Alto Laboratory, where he has worked since 1997. Before that, he was a Fulbright Fellow at the Institute of Systems Science in Singapore, after postdoctoral work at Cambridge University. He received a BS and a ME (Electrical) degree from Cornell University, and after working as a development engineer for Teradyne, in 1993 he received a Ph.D. in Electrical Engineering from Brown University. Dr. Foote's research interests include audio analysis and retrieval, multimedia signal processing, and panoramic video. Dr. Foote has published more than 60 papers in these areas and has received Best Paper Awards at the ACM Multimedia and ACM SIGIR international conferences.

**Don Kimber** Don Kimber has been involved with computer technology for over 20 years, and in research for about 15. He obtained his B.E. in E.E. from

Stevens Institute of Technology in 1980. He then worked as a software engineer on database query languages, and network applications until returning to University of California, Santa Cruz, to obtain a Masters in Computer and Information Science in 1988. He then worked at Xerox PARC on speech recognition related research while obtaining his Ph.D. at Stanford University in E.E. Don then joined PARC as a full time researcher, working on multimedia indexing and architecture. He left PARC in 1998 to become an independent contractor, and since then has done research for FX/PAL on collaborative systems, panoramic video, and virtual reality.

**B. S. Manjunath** B. S. Manjunath received the B. E. in Electronics (with distinction) from the Bangalore University in 1985, and M. E. (with distinction) in Systems Science and Automation from the Indian Institute of Science in 1987, and the Ph. D. degree in Electrical Engineering from the

University of Southern California in 1991. He is now a Professor of Electrical Computer Engineering and director of the Center for Bio-Image Informatics at the University of California, Santa Barbara. Dr. Manjunath was a recipient of the national merit scholarship (1978-85) and was awarded the university gold medal for the best graduating student in electronics engineering in 1985 from the Bangalore University. His current research interests include data mining, computer vision, learning algorithms, image/video databases and bio-image informatics. He is a co-editor of the book on Introduction to MPEG-7, published by Wiley (2002). He was an Associate Editor of the Transactions on Image Processing.



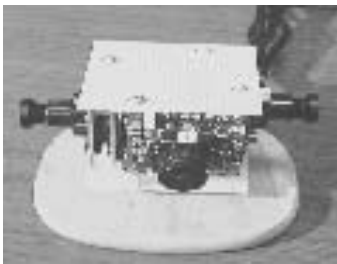


(a) Panoramic scene view.

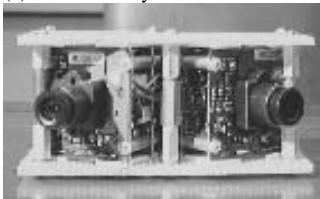


(b) Region of interest.

Fig. 1. An example of a panoramic scene and its region of interest.



(a) 360° view FlyCam.



(b) 180° view FlyCam.

Fig. 2. FlyCam examples.

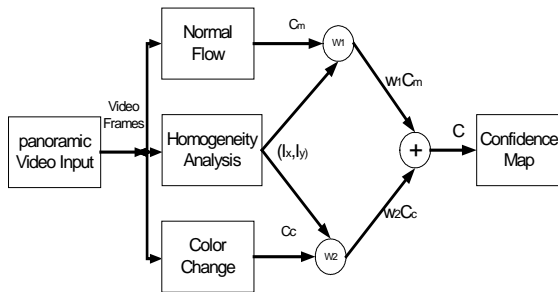


Fig. 4. Building the confidence Map.

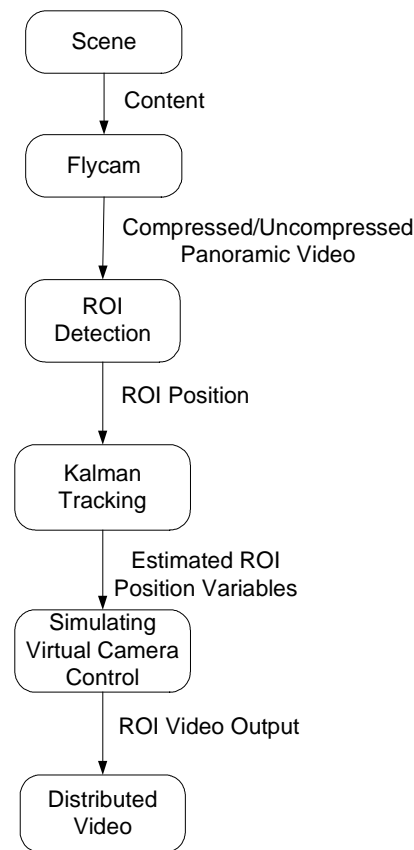


Fig. 3. General system architecture.



(a) A frame from a panoramic video.



(b) Confidence map.



(c) Motion region detection based on confidence map.



(d) Manually segmented motion region.



(e) Compressed-domain motion region after median filtering.

Fig. 5. Detection of moving part for a frame in a panoramic video.

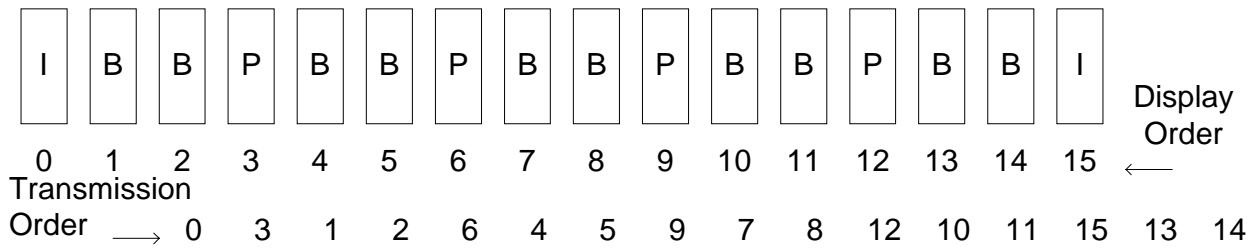


Fig. 6. An example of MPEG group of pictures (GOP).

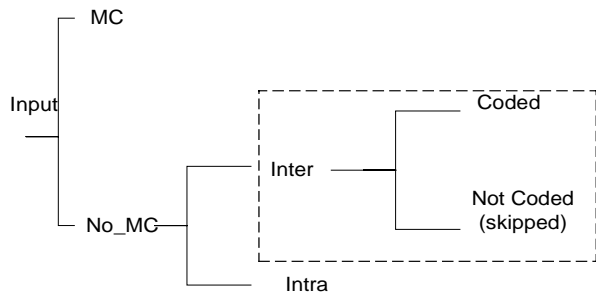


Fig. 7. Coded macroblock types in MPEG video P frame.




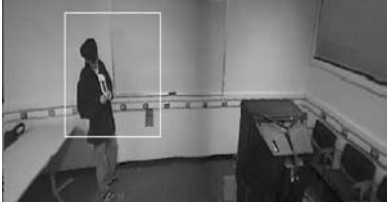
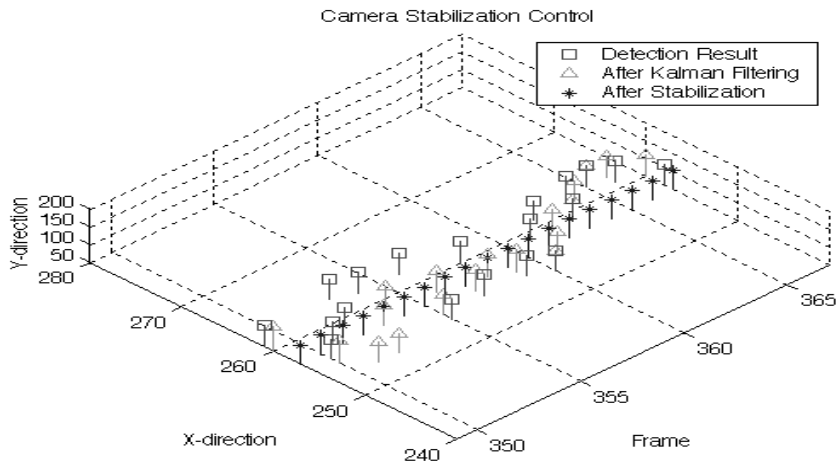
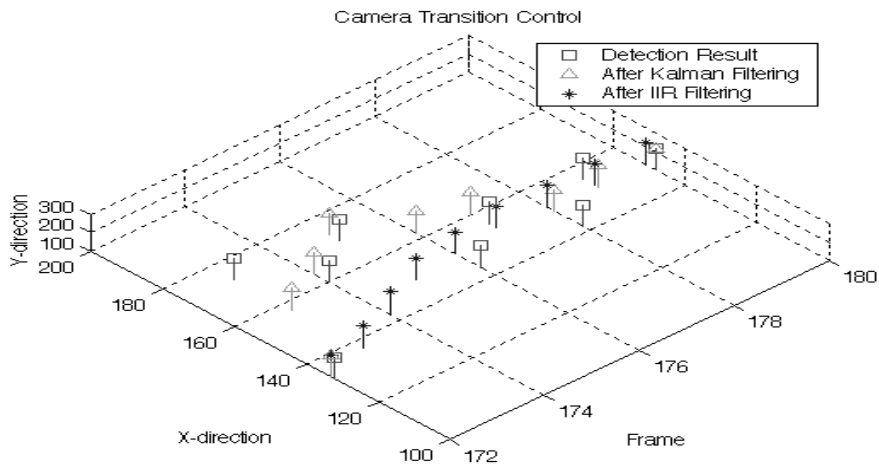
Frame	Frame Information
	(a) P frame Frame Size: 800x352 ROI Size: 200x200 ROI Centroid (x,y) : (231, 125)
	(b) B frame The ROI centroid of (a) is applied here.
	(c) B frame The ROI centroid of (a) is applied here.
	(d) I frame The ROI centroid of (a) is applied here.

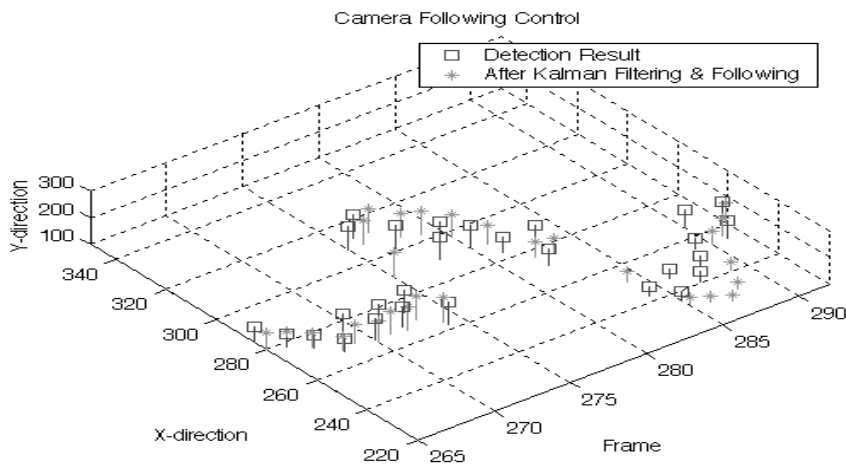
Fig. 8. Four consecutive frames in different frame types in an MPEG video



(a) Stabilization control

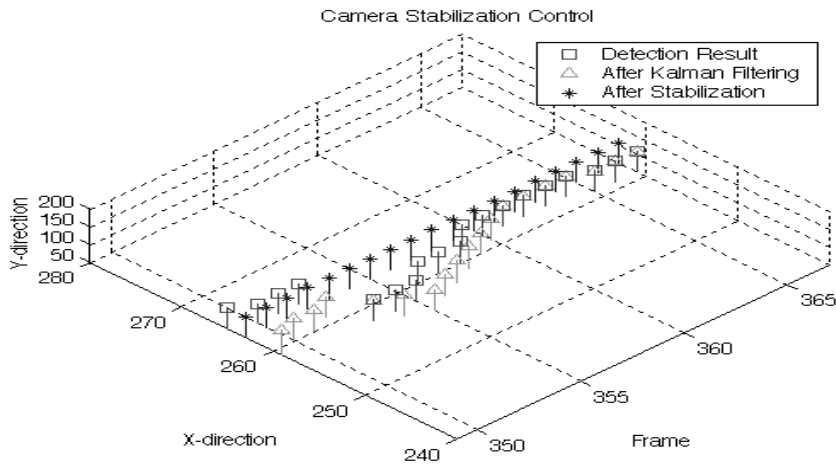


(b) Transition control.

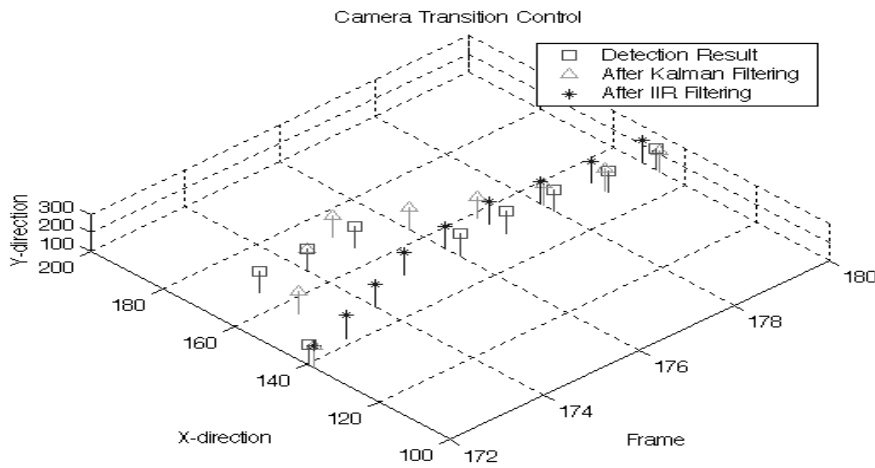


(c) Following control.

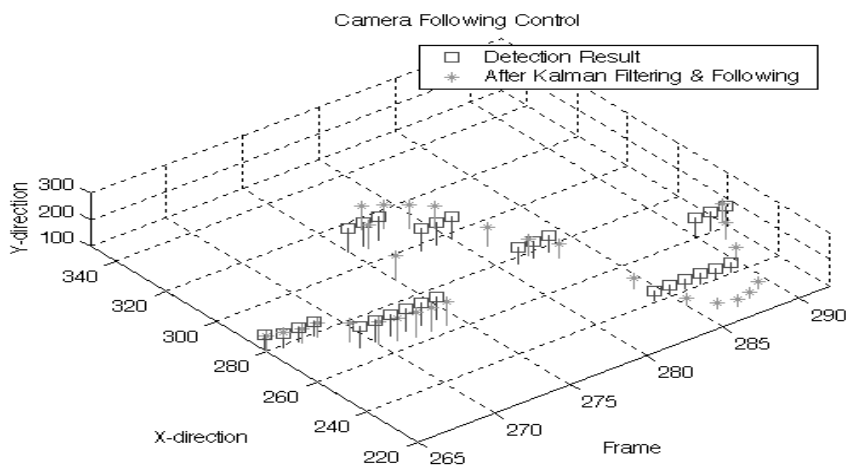
Fig. 9. Simulation of three types of camera control in uncompressed domain.



(a) Stabilization control



(b) Transition control



(c) Following control

Fig. 10. Simulation of three types of camera control in compressed domain.

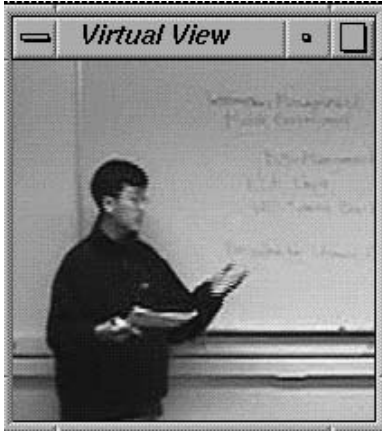


Fig. 11. The interface of the MPEG player.

		ROI detection result for frames at P frame only		ROI tracking result for all the frames	
		Centroid Drift (pixel)	Drift Standard Deviation (pixel)	Centroid Drift (pixel)	Drift Standard Deviation (pixel)
Uncompressed Domain	x direction	9	5	8	5
	y direction	22	9	23	8
Compressed Domain	x direction	23	16	21	15
	y direction	18	9	16	8

Table I. Statistics of ROI detection and tracking result.