

Acquiring Multi-Scale Images by Pan-Tilt-Zoom Control and Automatic Multi-Camera Calibration.

A.W. Senior*, A. Hampapur, M. Lu
IBM T. J. Watson Research Center,
PO Box 704, Yorktown Heights, NY 10598

<http://www.research.ibm.com/peoplevision>

Abstract

This paper describes a system for automatically acquiring high-resolution images by steering a pan-tilt-zoom camera at targets detected in a fixed camera view. The system uses a novel method to automatically calibrate between multiple cameras, estimating the homography between the cameras in a home position, together with the effects of pan and tilt controls and the expected height of a person in the image. These calibrations are chained together to steer a slave camera. In addition we describe a simple manual calibration scheme.

1. Introduction

In any computer vision application, there is some minimum resolution for which the desired processing is practical or effective. For instance, in face recognition, surveillance and audio-visual speech recognition, sufficiently high resolution images of the face are necessary for recognition (by human or machine) to be practical. The area of coverage of such a system is usually limited by the need for resolution.

Additional coverage can be achieved by adding cameras at additional expense, for cameras and processing power, and increased architectural complexity for the system. Recently, high resolution cameras have offered an alternative, but still at additional cost for cameras and processing. A further option, explored here, is the use of zoom cameras that use a foveation strategy to direct the sensor elements available to a region of interest. The problem is now one of controlling the zoom camera so that the desired behaviour is achieved.

Here, one option is to have a single camera that tracks regions of interest, but is oblivious to events going on beyond its current narrow field of view [2]. Alternatively two-or more cameras can be combined, with one giving an overview of the scene, from which is derived information for the control of the zoom camera. Now the problem becomes one of relating the views of the two cameras. Again

previous solutions have been proposed, including the use of colocated cameras, whose viewpoints are assumed to be identical, or learning a linear mapping from manually established correspondences [10]. Previously, we have used two or more calibrated cameras [5] to triangulate an object's position and determine the steering parameters for a third, pan-tilt-zoom camera that is calibrated to the same coordinate system. Kang *et al.* [6] have described a system that tracks across moving and stationary cameras, but this system also requires calibration between the cameras.

In this paper we propose two methods that allow a steerable camera to be automatically steered to a region of interest determined from the view of a second, fixed camera, doing away with both the calibration and the need for two fixed cameras in our previous scheme. One method, described in Section 2, uses a manual registration stage and the other, described in Section 3 is based around a system for automatically learning the homography between the cameras in some home position, and learning the effect of pan/tilt/zoom on the steerable camera. Combining these with a camera control policy, we have created an automatic camera control system.

This system has been deployed in a surveillance scenario to achieve automatic multi-resolution data. From a wide-angle, fixed camera view, we acquire information about the location and general behaviour of objects, but from the automatically steered zoom camera, details of the object appearance are available at much higher resolution.

2. Control without calibration

One simple way to achieve the desired link between positions in one camera's view to steering parameters for a second camera, is simply to set up a look-up table. We built an initial system using this method and have used it to investigate the design of an automatic camera control system before constructing an automatic calibration module that does away with the need for the manual look-up table.

In this system, a number of cameras are available, any of which may be fixed or steerable. The fixed cameras are termed "masters" and in these we run our standard tracking

*Contact aws@us.ibm.com

algorithms that use background subtraction to detect moving objects and track these objects through occlusions to maintain object tracks. Tracks and object appearances are transmitted to a central database where they are available for subsequent search and play-back. This “Smart Surveillance System” is described elsewhere [4].

The system is manually calibrated by designating a number of target areas or regions of interest (particularly such areas as entrances, high traffic areas etc.) in the master cameras’ views. These target areas are associated with ‘bead’ positions (p, t, z) for any or all of the slave pan-tilt-zoom cameras. Using a graphical user interface, the operator draws the region of interest, and steers the corresponding slave to the bead position to establish the correspondence.



Figure 1: A master camera’s view tiled with regions of interest each tied to a bead location of a slave camera. Each slave may have such a tiling in each master view.

In operation then, the system tracks objects in the master camera, and predicts the location forward in time using a first order model. These locations are then compared to the set of regions of interest for each of the slave cameras and, if a match is found, the camera is steered to the corresponding bead. If continuous tracking across a region is required, individual points can be marked, as described by Zhou *et al.* [10], and pan/tilt/zoom parameters interpolated.

We have built a multi-scale browsing tool (Figure 2) that allows the high-resolution imagery to be browsed along with track information in the database. Tracks can be selected and replayed along with the corresponding automatically-acquired zoomed-in multi-scale imagery, allowing security operators enhanced ability to understand situations and recognize individuals, vehicles and so on.

2.1 Camera control policies

In practice, we may have a complex arrangement with multiple fixed and steerable cameras. In any one camera view, multiple objects may be being tracked. This situation then requires a camera assignment and control policy to determine which cameras should be steered at which objects at

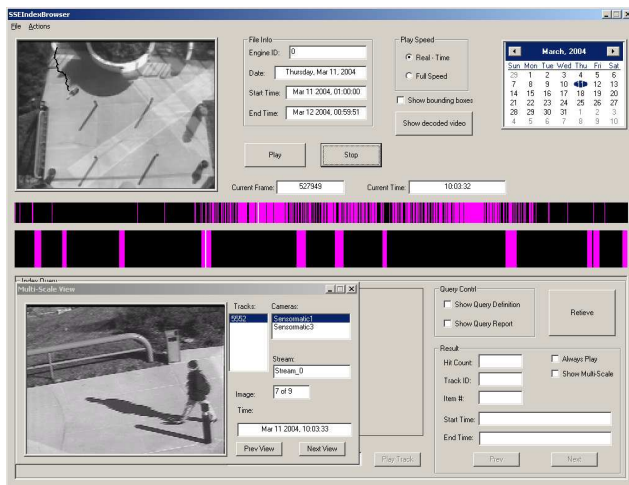


Figure 2: The multi-scale browser. At top left the master view, with superimposed track information. In the centre are two activity bars showing track activity in two different time scales. At lower left is the close-up imagery captured for the current track. The query interface is shown in the lower right corner.

any given time. Such policies are application dependent, but might be designed to optimize the following criteria:

- Observe each target at least once;
- Observe each target from as many views as possible;
- Maintain continuous tracks of objects;
- Steer the camera most likely to give a frontal face image (estimated using the fixed camera view [9])
- Steer the cameras to maximize a combined quality measure of the imagery of the targets (e.g. maximal coverage, highest resolution, most frontal face views, least motion blur), taking into account the objects current and predicted appearance, behaviour, location and the previously acquired images.

One simple policy that we have implemented assigns all slave cameras to the first target, drawing off cameras to additional targets as they are observed, and permuting cameras every two seconds so that different views of each target are obtained. Costello *et al.* [3] have recently investigated the effectiveness of various time-based active camera control policies.

2.2 Other extensions

Another scenario that we have been able to implement with the above scheme is a one-camera system where the camera functions as both master and slave. Here, as with any master, regions of interest are designated for the camera, and as with any slave, the corresponding beads are set up. In operation, the camera starts as a master, with the tracking system running as normal. Objects are tracked, but when a track is predicted to enter the region of interest, the tracking

is suspended and the camera is steered as a slave to the bead corresponding to the activated target. The camera dwells in the bead position for a certain time before returning to the home position. Then the tracker and background model are reinitialized, and the camera continues as a master until the next target is triggered.

We have implemented such a camera as a licence plate recorder. A normal PTZ camera surveys an entrance, recording tracks of passing vehicles and pedestrians, which are entered into the Smart Surveillance Index as normal. However, when a vehicle pulls up to an entrance barrier, it enters a region of interest, triggering the camera to zoom in on the region where a licence plate is to be expected. One or more images are recorded and then the camera returns to the home position to continue wide-angle surveillance.

Similarly, we have implemented a system for environments where tracks are rare, wherein two cameras each act as masters, and as slaves to each other. When a track is observed in one camera, it continues to track the object, but the other camera is temporarily drafted in to acquire multiscale imagery. In this way we require no additional cameras, but acquire multi-scale imagery at the penalty of short periods of inattention to one camera’s view.

3 Automatic calibration for PTZ control

Steering a slave camera to a point seen by another camera can be achieved if the 3D position of the object is inferable (e.g. by triangulation from two cameras) and the relation of the slave camera’s position to the coordinate frame is known, which requires extrinsic calibration, together with knowledge of the pan/tilt behaviour of the camera. Since we assume only one master and no calibration, this 3D approach is not available. Calibration information is generally obtained through a supervised calibration procedure, although Sinha and Pollefeys [7] have shown automatic intrinsic and extrinsic calibration of PTZ cameras with similar viewpoints. Instead, we steer the slave camera to a pan/tilt coordinate calculated using the series of transformations shown in Figure 3, each of which is learned automatically from unlabelled training data.

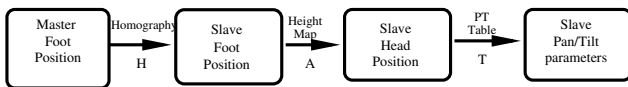


Figure 3: The sequence of transformations to allow a slave camera to track an object tracked in the master camera’s view.

The transformations are based on the assumption that objects are moving such that a known part of the objects moves on a plane. In this case we assume that the lowest point of

an object is the point of contact of the foot with the ground and that the ground is approximately planar.

3.1 Camera-to-camera calibration

The calibration of one camera relative to another is carried out after the single manual step of having a user point the slave camera at the same area that is viewed by the master, *i.e.* so that the slave has an overview of the region in which it will be used to track objects.

We assume that the ground in the area viewed is approximately planar, and thus that a linear transform (a homography) is sufficient to map ground plane points in the view of the master into points in the view of the second camera. The system learns this homography automatically using a method based on that of Stauffer and Tieu [8]:

The homography, H , is chosen as the homography that best matches a set of points, (x, y) , on the ground plane in one image to the positions, (x', y') , of those same points when seen from the other camera.

$$x' = \frac{\hat{x}}{\hat{z}} \quad (1)$$

$$y' = \frac{\hat{y}}{\hat{z}} \quad (2)$$

$$\begin{pmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \end{pmatrix} = H \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \quad (3)$$

Such pairs of corresponding points are obtained by using our tracking system. In a short training phase, The Smart Surveillance Engine is applied to the views of both cameras simultaneously, so that object tracks are obtained from objects seen in both views. The SSE is run for a short time (an hour or 100 tracks has proved sufficient) and the tracks stored, as usual in a database.

Tracks from the database are processed so that only tracks of sufficient duration (at least 100 frames) are retained and parts of tracks that take place when no track is visible in the other view are discarded as they can have no corresponding points.

The homography is determined using a RANSAC procedure whereby sets of three tracks are randomly selected from one view. For each track a simultaneous track in the other view is selected and points with the same timestamp are assumed to correspond. Using these corresponding points, the least-squares fit homography is estimated and this is applied to all the points in the database with correspondences. The number whose image under the homography fall close to the corresponding simultaneous point are counted. We then iterate (500 iterations) retaining the homography with the highest number of matches as the best fit. As Stauffer and Tieu point out, if there are multiple

tracks at the same time, a correspondence is harder to determine, so we bias the sampling distribution towards tracks that have only one possible corresponding track, as well as biasing towards long tracks.

Figure 4 shows an example of the automatically determined homographies between two views based on track correspondences.

3.2 Pan-tilt calibration

To be able to steer the pan-tilt-zoom camera automatically to a determined point, the system needs to know the effect of the pan, tilt and zoom controls. These can be derived from manufacturer’s specifications, in terms of degrees, which also requires the focal length of the camera to be known. However, to be able to deal with unknown, uncalibrated cameras, or cameras whose specifications may change with time, we have created a system that automatically learns the mapping from pan/tilt commands to image motion. A similar method learns the effect of zoom controls but is not used by this system.

The system begins by steering each camera to the known home position where the camera correspondence homography was trained. Here, a point tracker is initialized by finding corners. Then the camera is issued a set of pan & tilt commands, (p, t) , to move it in a known pattern around the home position, while patches around the points are tracked with a Lucas Kanade tracker [1]. RANSAC is again used to find the affine transform that best fits the points’ motions, making the fit robust to mismatches and points that are obscured by objects in the scene during this calibration procedure. For each pan/tilt pair the motion, $(x' - x'_0, y' - y'_0)$, of the optic centre, (x'_0, y'_0) , is recorded. After completing a pattern (we use a spiral and twelve-armed star configurations repeated until the motion has moved the optic centre beyond the original view’s boundary), a least-squares linear transform T is fit to all the points:

$$\begin{pmatrix} p \\ t \end{pmatrix} = T \begin{pmatrix} x' - x'_0 \\ y' - y'_0 \\ 1 \end{pmatrix}. \quad (4)$$

Figure 5 shows the measured x displacement plotted against pan and tilt coordinates for the spiral part of the training pattern.

3.3 Head tracking

The above calibration allows us to map image points on the ground plane of one view into steering parameters for the other camera to foveate on the same point. Unfortunately, it is rare for the region of interest to lie in the ground plane. If the regions of interest were of a uniform height the homography could be learned on the plane where the regions of

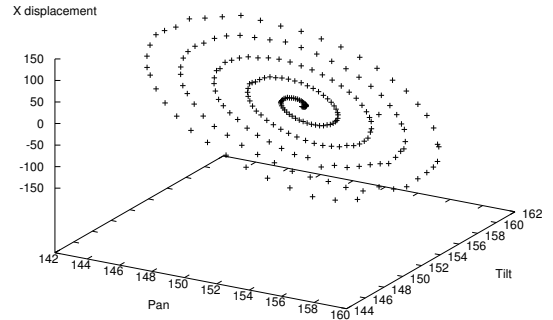


Figure 5: Estimated displacement x' plotted against pan and tilt position for the spiral part of the training pattern.

interest lay (*e.g.* we could learn the homography between the planes in which all heads lay). Here we wish the system to zoom-in on pedestrians’ heads, regardless of height, and approximate the head position with the highest pixel in a tracked object. To determine this location in the slave view, we use a system to find the mapping between object height h in the master image and height h' in the slave image. We assume that for a given foot location, the height of an object in the slave view is proportional to the height in the master view, and that the head and foot pixels will each closely correspond in two different views (*i.e.* the objects are tall and thin). The factor of proportionality depends on the location of the object, so we estimate the transform $A = (a_0, a_1, a_2)$ such that

$$h' = h A \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}, \quad (5)$$

where (x, y) is the object’s position in the master image. The transformation A is estimated by a least-squares fit on the set of training tracks used to determine the homography, using only the points for which the homography determines the correct correspondence. For the moment, cameras are assumed to be aligned (and have small distortion) so that heads are assumed to lie vertically above the feet. A more complex mapping can be learned if this is not true.

Figure 4 shows for each view the predicted height and position of an object that is a twelfth of the image height, when the object is placed at points on a regular grid throughout the other view.

4 Experiments

To validate the performance of the tracker, we operated it for a period with the slave camera remaining at a wide angle (the “home position” zoom whereby the slave’s field

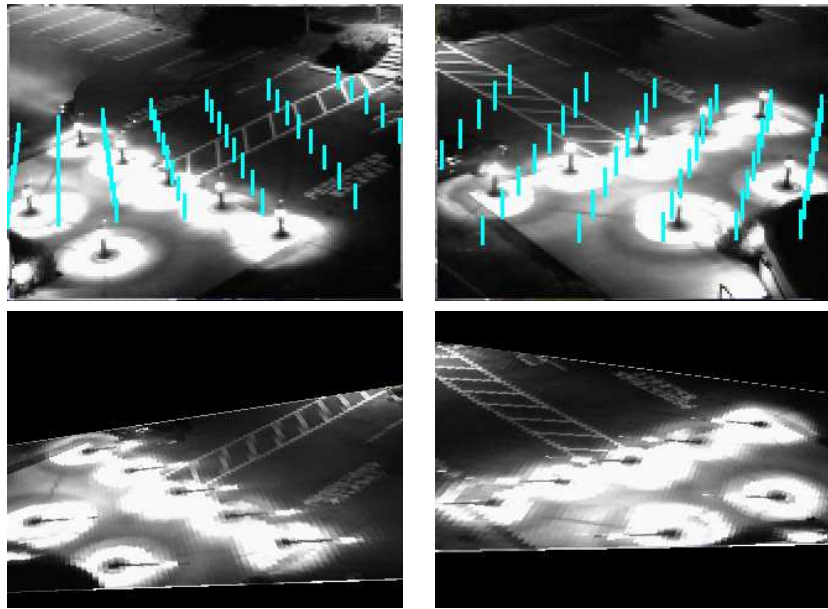


Figure 4: The learned camera homographies. Top row: the views of the two cameras. Bottom row: View of the other camera mapped to match the view above with the learned homography. In the top images, are shown the estimated positions and heights of objects placed on a regular grid in the other view, and with a height of an twelfth of the image height.

of view was approximately the same as the master’s). Every half second while the slave is tracking an object, a still frame from the slave’s video is stored. 455 of these frames were examined (from 30 single-person tracks) and the centre of the tracked head was marked. In an ideal system the head would be in the centre of the frame (though we actually only try to predict the location of the *top* of the head). From these images, for any given zoom value, we can count the number for which the head would still be visible in the frame, and thus estimate the probability for a tracked target to be kept in field for a chosen zoom policy. Figure 6 shows the relative position of the head in the frame for each of the frames. Table 1 shows the probability of observing a head for a given zoom ratio, together with the approximate head size at that zoom ratio.

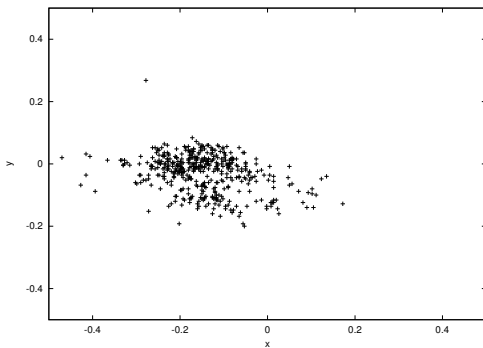


Figure 6: Distribution of head locations within the recorded frames ± 0.5 being the image boundaries. The head was not visible in five (of 455) frames.

Zoom Factor	Probability %	Head area
1.0	98.9	8
1.5	97.1	20
2.0	89.0	32
2.5	72.1	50
3.0	55.6	72
4.0	30.3	128
5.0	17.6	200
6.0	11.2	288
8.0	7.0	512
10.0	4.8	800

Table 1: Probability of observing a head for a given zoom ratio, together with the approximate head area in pixels.

A fixed camera would also achieve less than 100% coverage simply because there are areas visible in the master view that are beyond the edge of the slave view. We see that 89% coverage can be achieved with a $2\times$ zoom. Overall accuracy is influenced by errors in segmentation (both foot position and object height), errors at each stage in the coordinate transformations, failures in the ground plane assumption and due to lag in the camera control system. (For these experiments Sensormatic VII dome surveillance cameras were used, controlled by serial commands). Since these rates are per frame, and many images are acquired per track, the probability of observing a head close-up of a given track is much higher.

The component of error due to homography miscalculation was estimated by calculating a homography on a set of hand-marked points. Over two different pairs of views, the

average pixel error between point positions calculated using the automatic and hand-estimated homographies was 5.0 pixels and the maximum error was 10.6 pixels (estimated on a uniform sampling of pixels in one view whose image would be visible in the second view).

To estimate the combined error due to homography and pan/tilt map estimation error, we conducted a simple test of steering the slave to a small number of ground plane points designated in the master image, and measuring the displacement between the corresponding point and the centre of the slave image for each, with no segmentation error, tracking noise, camera control delays or height prediction errors (but some human error in the localization). The error distribution is shown in Figure 7. This gives an RMS error of 24.0, vs 58.0 pixels (on 320×240 images) for the head position error of the full tracking system.

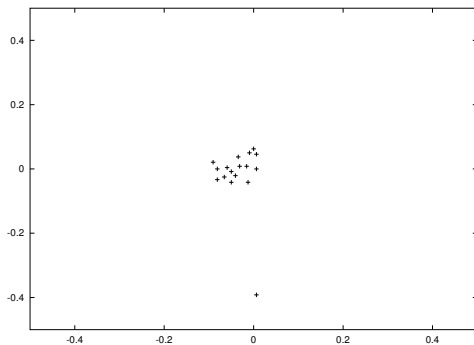


Figure 7: Distribution of intended target position relative to frame centre (± 0.5 being the image boundaries.)

5 Conclusions and further work

We have described two methods to steer pan-tilt-zoom cameras to automatically follow targets tracked in a master camera view, neither of which requires extrinsic or intrinsic calibration. One system requires the manual setting up of a set of “beads”, the other uses a fully automatic method to calibrate systems of multiple cameras. The system is general, supporting arbitrary combinations of master & slave cameras, even allowing cameras to perform both roles according to the current situation. We have conducted performance experiments and experimented with a few camera control policies out of a large space of possibilities. There is much scope for investigation of camera control policies for different applications.

The automatic system performs well, affording a significant magnification over a wide area. Because of tracking noise, the pan/tilt controls are noisy, and we are currently introducing a Kalman filter to smooth the tracking and prediction. With smoother camera control, it is hoped that image registration can be applied in the slave’s video stream

to allow closed-loop feedback control to track objects in the close-up view. Currently master cameras operate independently, but we are investigating multi-camera tracking algorithms that would correctly handle tracks for the same object in different views.

A further planned refinement of the system is to model the scene with greater accuracy, for instance allowing piecewise homographies between the two views for greater accuracy on non-planar scenes.

References

- [1] Jean-Yves Bouguet. *Pyramidal Implementation of the Lucas Kanade Feature Tracker*. Intel OpenCV Documentation, 1999.
- [2] D. Comaniciu and V. Ramesh. Robust detection and tracking of human faces with an active camera, 2000.
- [3] C. J. Costello, C. P. Diehl, A. Banerjee, and H. Fisher. Scheduling an active camera to observe people. In *Visual Surveillance and Sensor Networks*, page 46. ACM, October 2004.
- [4] A. Hampapur, L. Brown, J. Connell, S. Pankanti, A.W. Senior, and Y.-L. Tian. Smart surveillance: Applications, technologies and implications. In *IEEE Pacific-Rim Conference On Multimedia*, Singapore, Dec 2003.
- [5] A. Hampapur, S. Pankanti, A.W. Senior, Y.-L. Tian, L. Brown, and R. Bolle. Face cataloger: Multi-scale imaging for relating identity to location. In *IEEE conference on Advanced Video and Signal Based Surveillance*, pages 13–20, Miami, July 2003. IEEE Computer Society.
- [6] Jinman Kang, Isaac Cohen, and Gerard Medioni. Continuous tracking within and across camera streams. In *Proceedings of Computer Vision and Pattern Recognition*, 2003.
- [7] S.N. Sinha and M. Pollefeys. Towards calibrating a pan-tilt-zoom cameras network. In P. Sturm, T. Svoboda, and S. Teller, editors, *OMNIVIS*, May 2004.
- [8] Chris Stauffer and Kinh Tieu. Automated multi-camera planar tracking correspondence modeling. In *Proceedings of Computer Vision and Pattern Recognition*, volume I, pages 259–266, July 2003.
- [9] Y.-L. Tian, L. Brown, J. Connell, S. Pankanti, A. Hampapur, A. Senior, and R. Bolle. Absolute head pose estimation from overhead wide-angle cameras. In *Intl. Workshop on Analysis and Modelling of Face and Gesture*, October 2003.
- [10] Xuhui Zhou, Robert T. Collins, Takeo Kanade, and Peter Metes. A master-slave system to acquire biometric imagery of humans at distance. In *First ACM SIGMM International Workshop on Video Surveillance*, 2003.