Realtime Visual Tracking of Aircrafts

Ajmal S. Mian
School of Computer Science and Software Engineering
The University of Western Australia
35 Stirling Highway, Crawley, WA 6009
ajmal@csse.uwa.edu.au

Abstract

Aircraft tracking has applications in guided landing, automatic scoring in aerobatics and target tracking for military. Radar based tracking is expensive, gives away the position of the radar and is prone to jamming. However, vision sensors are low cost, passive and robust to jamming hence motivating their use for aircraft tracking. This paper proposes a modified KLT tracking algorithm and tests its performance by tracking aircrafts. Vision based aircraft tracking is challenging because of the higher speed, agility and distance of the aircraft from the camera. The proposed tracker uses a feature clustering criterion to track an object based on its multiple local features. The local features are continuously updated to make the tracker robust to changing appearance of the object. The proposed tracker is generic and can be used to track multiple objects of any kind by defining multiple feature clusters. Experiments were performed on real data collected during the Air Race 2006 which will be made publicly available. The algorithm achieved accurate tracking in the presence of clouds and background clutter as long as the object size was comparable to the feature window. However, tracking failed at very small scales of the aircraft and in the presence of background clutter while the aircraft was flying low, a scenario where radars also fail.

1. Introduction

Radars are generally used for detecting and tracking aircrafts. However, radars are expensive and use active illumination of aircrafts with electromagnetic waves. This gives away the position of the radar which is undesirable from military point of view and is also a major source of interference with nearby electronic equipment. On the other hand, vision sensors are low cost, passive and robust to jamming. This motivates the use of vision based aircraft tracking. Even though visual sensors have a limited range compared to radars, they can be used for a number of applications. Visual tracking can be used to guide aircrafts during their approach for landing and for automatic scoring of aerobatics performance. In the military, visual aircraft tracking can be used to aim and guide weapons.

Motion estimation and tracking in videos is an established area of computer vision. Computer vision has been successfully used for tracking pedestrians and other objects of interest. Tracking algorithms can be roughly divided into optical flow and local feature based techniques. Optical flow based techniques exploit the fact that there is minimal change in images that are taken at small intervals of time, such as in video, even in the presence of relative motion between the camera and the objects in the scene. Optical flow based techniques give a dense field of flow between frames. However, areas of the image which are not rich in texture (such as plain color) are a major source of error. A comparison of optical flow based tracking techniques is given by Barron et al. [1].

Feature based tracking algorithms extract local features (regions) from the first frame and search for the corresponding features in the subsequent frames. Identifying features, that are good for tracking, is an important first step in these techniques. In some cases, the choice of features can be restricted to a pre-selected object which is required to be tracked such as aircrafts in the context of this paper. In such cases, the object of interest must be detected by an object detection algorithm such as a Haar-based detector [14]. Object detection algorithms search for the appearance of an object of interest in an image, usually at multiple scales. Compared to tracking, object detection takes more computational resources which is why tracking between frames is preferred as opposed to frame by frame detection.

This paper proposes two modifications to the Lucas and Kanade [6] tracking algorithm which is based on optical flow estimation however, it has been extended by Shi and Tomasi [11] to include a feature identification step and cater for affine transformations of the features. These changes makes the algorithm somewhat hybrid between the optical

978-0-7695-3456-5/08 $25.00 © 2008 IEEE
DOI 10.1109/DICTA.2008.33

flow and feature based tracking categories discussed above. The proposed changes and contributions of this paper include (1) a tracking by local feature clustering criterion to track objects as opposed to image points, (2) continuous updating of the object features to cater for changes in the object's appearance due changing pose, scale and illumination, and (3) introduction of a new database for aircraft tracking which is the first of its kind. This database contains various aircrafts performing aerobatics and will be made freely available to the research community.

The rest of the paper is organized as follows. Section 2 gives a brief survey of tracking algorithms. Section 3 gives a summary of the tracking algorithm and Section 4 gives an overview of the database. Results are presented in Section 5 and Section 7 concludes the paper.

2. Literature Review

Trucco and Plakas [13] give a detailed survey of video tracking techniques and divide tracking into *motion* and *matching* problems. The former predicts the location of the tracked object in the next frame whereas the latter confirms the location of the tracked object in the next frame. Thus tracking proceeds by iterating between two stages namely predict (define a search region) and update (confirm the object location). In the simplest model, prediction may be a fixed window around the previous location. The size of the window is a trade-off between accuracy and speed. Small search regions may miss the tracked object in the next frame whereas a large search region requires more computational resources. This problem has been addressed by using multiresolution image pyramids and performing tracking in a coarse-to-fine manner [3].

The simple search window model (including the multiresolution image pyramid based) does not take advantage of the temporal motion information of the object. Using this information, the search region can be more accurately predicted. A well known model that keeps an estimate of the dynamic state of a system to predict the search region for the next frame, is the Kalman Filter [9][7]. Extended versions of the Kalman Filter [9][2] have also been used to deal with non-linear dynamic systems.

Kalman Filter has also been used for estimating depth from image sequences [8]. However, Kalman Filter is based on Gaussian distribution and therefore supports a single peak [13]. In other words, only one target can be tracked. Particle filtering allows multimodal distributions for simultaneous tracking of multiple targets [5]. Even if one target is of interest and needs to be tracked but in the presence of many others, it is sometimes better to track all of them in order to avoid losing the target of interest in the event of interference from others.

An interesting fact is that the Lucas and Kanade [6] algorithm was developed for solving the stereo correspondence problem which is analogous to video tracking in the sense that there is relative movement of the object (in the images) with respect to the camera. However, in stereo, this movement occurs in the frames because the two images are acquired from different viewing points. On the other hand, in video tracking either the object to be tracked or the camera or even both could be moving. Unlike stereo, video tracking can exploit the motion information which is not available in a static pair of stereo images. But again, if both the object and the video camera are moving (as is the case in this paper), it can result in quite complex motion models. The complexity of motion model further increases when there is significant variation in the depth (distance from camera) of the object and when the camera also performs optical zoom operations. These two conditions are also true about the database used in this paper. For such complex motions, the assumption of constant instantaneous acceleration of most dynamic state tracking models does not hold. This is the main reason why a static model is used for tracking aircrafts in this paper.

3. Tracking Algorithm

The tracking algorithm used in this paper is commonly known as the KLT (Kanade-Lucas-Tomasi) algorithm because it was initially proposed by Lucas and Kanade [6] and later developed fully by Tomasi and Kanade [12]. Shi and Tomasi [11] further extended the algorithm by adding a good feature selection step. Detailed descriptions of the feature tracking and selection algorithms are given in [6, 12, 11]. However, brief descriptions are reported here for completeness. Since the feature selection criteria is derived from the tracking algorithm itself, the tracking algorithm is discussed in Section 3.1, prior to the feature selection algorithm in Section 3.2. In Section 3.3 we describe our extensions to the algorithm.

3.1. Feature Tracking

Given that there is minimal motion between consecutive frames of a video sequence, the following equation is satisfied,

$$I(x, y, t + \tau) = I(x - \delta, y - \eta, t), \tag{1}$$

where $I(x, y, t)$ is an intensity image at instant t and τ, δ and η are small increments in time, x dimension and y dimension respectively. Eqn. 1 will not be true if there is a change in illumination and for boundary points which disappear or reappear in the image. However, Eqn. 1 holds true for good features (described in Section 3.2) which are away from boundaries and are somewhat distinct. A feature

is essentially extracted from a small image window, rather than just a single point, which is tracked between frames by estimating the displacement vector $\mathbf{d} = (\delta, \eta)$ i.e. a simple translation. With this definition of $\mathbf{d}$, let

$$J(x) = I(x, y, t+\tau) , \quad \text{and} \tag{2}$$
$$I(x-\mathbf{d}) = I(x-\delta, y-\eta, t) , \tag{3}$$

where J and I are images at time $t+\tau$ and t respectively. We can rewrite Eqn. 1 as

$$J(x) = I(x-\mathbf{d}) + n(x) , \tag{4}$$

where $n(x)$ is noise. Our aim is to find the displacement vector $\mathbf{d}$ that satisfies Eqn. 4 which can be calculated by minimizing the residue error ε given by

$$\varepsilon = \int\int_W [I(x-\mathbf{d}) - J(x)]^2 w(x) dx , \tag{5}$$

where $w(x)$ is a weighting function and W is the search window. The weighting function $w(x)$ is usually set to unity but it could also be Gaussian or derived from the intensity pattern of the window to emphasize certain pixels. Eqn. 5 can be solved in many ways but for small displacements (i.e. $\mathbf{d} << W$), the method presented in [6] is the most efficient. For small displacement $\mathbf{d}$, Eqn. 3 can be approximated by the linear terms of its Taylor series

$$I(x-\mathbf{d}) \approx I(x) - \delta\frac{\partial I}{\partial x}(x) - \eta\frac{\partial I}{\partial y}(x) , \tag{6}$$
$$\mathbf{g} = [\frac{\partial}{\partial x}(I) \quad \frac{\partial}{\partial y}(I)]^T , \tag{7}$$
$$I(x-\mathbf{d}) \approx I(x) - \mathbf{g}.\mathbf{d} , \tag{8}$$

where $\mathbf{g}$ is the gradient of image $I(x)$. Substituting $I(x-\mathbf{d})$ from Eqn. 8 in Eqn. 5 gives us

$$\varepsilon = \int\int_W [I(x) - \mathbf{g}.\mathbf{d} - J(x)]^2 w(x) dx , \tag{9}$$
$$\varepsilon = \int\int_W [h - \mathbf{g}.\mathbf{d}]^2 w(x) dx , \tag{10}$$

where $h = I(x) - J(x)$. To find the value of $\mathbf{d}$ for which the residue ε is minimum, the right hand side of Eqn. 10 is differentiated with respect to $\mathbf{d}$ and set equal to zero

$$\int\int_W [h - \mathbf{g}.\mathbf{d}]\mathbf{g} w(x) dA = 0 . \tag{11}$$

Assuming $\mathbf{d}$ to be constant within the window W and rearranging terms we get

$$\int\int_W \mathbf{g}\mathbf{g}^T w(x) dA \; \mathbf{d} = \int\int_W h\mathbf{g} w(x) dA , \tag{12}$$
$$\mathbf{G}\mathbf{d} = \mathbf{e} . \tag{13}$$

3.2. Feature Selection

Shi and Tomasi [11] looked at the feature selection problem in a reverse manner. They defined good features as those that can be reliably tracked. Previous feature selection methods were based on the presence of texture or corners detected by high standard deviation of intensity or the zero crossings of the Laplacian of the intensity. Shi and Tomasi [11] argue that texture rich regions are not always good for tracking. Good features must be associated to fixed points in the real world rather than, for example, with a depth discontinuity or specular reflection.

Shi and Tomasi's definition of good features is optimal by construction. Mathematically speaking, a feature can be tracked reliably if Eqn. 13 represents good measurements and can be solved reliably. These conditions require that the eigenvalues of the 2×2 coefficient matrix $\mathbf{G}$ in Eqn. 13 are large (for good signal to noise ratio) and do not differ by several orders of magnitude. If λ_1 and λ_2 are the first and second eigenvalues of $\mathbf{G}$, then the corresponding feature window is accepted if $\min(\lambda_1, \lambda_2) > \lambda$ (where the threshold is derived from regions of the image with uniform brightness). The upper bound for λ is derived by repeating the same procedure for highly textured or corner regions.

To cater for situations where a feature becomes occluded or goes through an affine transformation, Shi and Tomasi proposed monitoring of feature quality during tracking. When the dissimilarity between features grows beyond a certain range, it is discarded. Moreover, they also determined affine transformations by Newton-Raphson style minimization in a similar way Lucas and Kanade [6] did for a translation only model. Affine changes are important to measure the dissimilarity between tracked features in distant frames and hence decide when to stop tracking a feature.

3.3. Proposed Modifications

Recall that the KLT algorithm was developed to track point features for image registration or stereo correspondence. A single feature may not correspond to a single object in the real world or there may be multiple features defined over a single object depending on its scale. We introduce two modifications to the algorithm in order to track one or more objects rather than arbitrary points in the video. As discussed earlier, the objects to be tracked must be identified by some other means like a Haar-based detector [14]. Once an object window is identified in a frame, features that satisfy the conditions of Section 3.2 are extracted inside this window only. A distance threshold t_d is initialized with half the window size and during tracking if the distance of any feature from the mean of all feature points increases beyond the threshold, it is dropped. Other features are also dropped

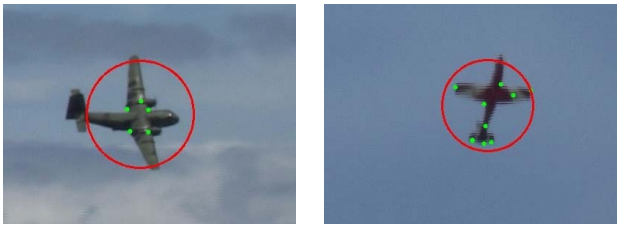
because they do not satisfy the similarity criteria. During this process, the threshold t_d is continuously updated thus giving an estimate of the scale of the tracked object at any time. The mean of the remaining features give the location of the tracked object. We call this tracking by feature clustering which is our first modification to the KLT algorithm.

Our second modification ensures that the features on the object are kept up to date. The appearance of an object can significantly change with its pose, scale and location (with respect to light sources) and with changes in illumination. Therefore, after the number of features on the tracked object fall below a threshold or after every N number of frames, we re-initialize the features on the object using the criteria of Section 3.2. The object window is derived from the last value of t_d and the center of the window is the mean of the last set of feature points. In our implementation, we used a window size of $m \pm \kappa t_d$ (where m is the mean of the feature points and $\kappa > 1$ is a scaling factor). A scaling factor of greater than one allows more features to be detected if the scale of the tracked object increases (due to camera zoom or decreasing distance from the camera). This way, the scale of the object is also updated at regular intervals. Note that decreasing scale of the object is automatically catered due to the tracking by feature clustering criterion discussed above. Both processes (scaling up and clustering) together ensure that the object scale and distance threshold t_d are always updated.

4. Video Database for Aircraft Tracking

The video database was collected during the Air Race 2006 in Perth, Australia. Most of the data was collected in AVI format at 25 frames per second and a frame size of 720×576. There are 27 video sequences of varying length. The longest video sequence lasts for over one minute. In addition to these 27 sequences, there are four video sequences acquired with a low resolution camera at 15 frames per second and a frame size of 640×480. There are a total of over 23000 frames in the database. The database contains six different types of aircrafts including propeller planes, jet planes, aerobatics planes, fighter, transport aircraft and a helicopter. Some of the challenges, from tracking point of view, in the database include the following:

- Camera movement and optical zoom.
- High speed and agility of aircrafts.
- Aircraft pose variation in pitch, yaw and roll.
- Occlusions due to clouds and trees.
- Specular reflections from the aircraft canopy.
- Smoke from aerobatics aircrafts.
- Background clutter during low level flying.



(a) (b)

Figure 1. Sample frames showing (a) a transport aircraft and (b) an aerobatics aircraft. The green points represent detected features and the red circles show the location and scale of the tracked objects. Frames are cropped from 720×576 to 400×300 for brevity.

This is the first database of its kind and will be made available to the research community free of cost [10].

5. Results

The proposed algorithm is generic and can be used for tracking any object of interest. Moreover, it can be used to track multiple objects by defining multiple clusters of features, one for each object. In this Section, we present results for tracking a single aircraft using the database described in the previous section. Since object detection is outside the scope of this paper, the aircraft was manually identified to initialize the tracker. Once, initialized, the tracking algorithm automatically tracked the aircraft in the remaining frames.

The algorithm was implemented in C++ using the OpenCV (Open Computer Vision) libraries [4] in order to achieve real-time tracking performance. Demonstration videos can be downloaded from [10]. For visualization, a red circle is drawn around the tracked object. Local features over the object are indicated by green points. Fig. 1 shows sample frames from different tracking sequences. Note that all frames shown in the figures of this paper have been cropped to a fixed size of 400×300 (from 720×576) so that the aircrafts are more visible. The scale of the aircraft in the cropped frames is still quite small which gives an idea about the difficulty of the tracking scenario.

Fig. 2 shows a tracking sequence of an aircraft tracked over the Perth city sky line. Note that the algorithm detects additional feature points over the buildings when the aircraft comes close to them. However, these points are dropped due the tracking by feature clustering criterion as the aircraft moves away from the building. Fig. 3 shows selected frames from the same sequence in order to elaborate this effect. Notice that the red circle lags behind in the second

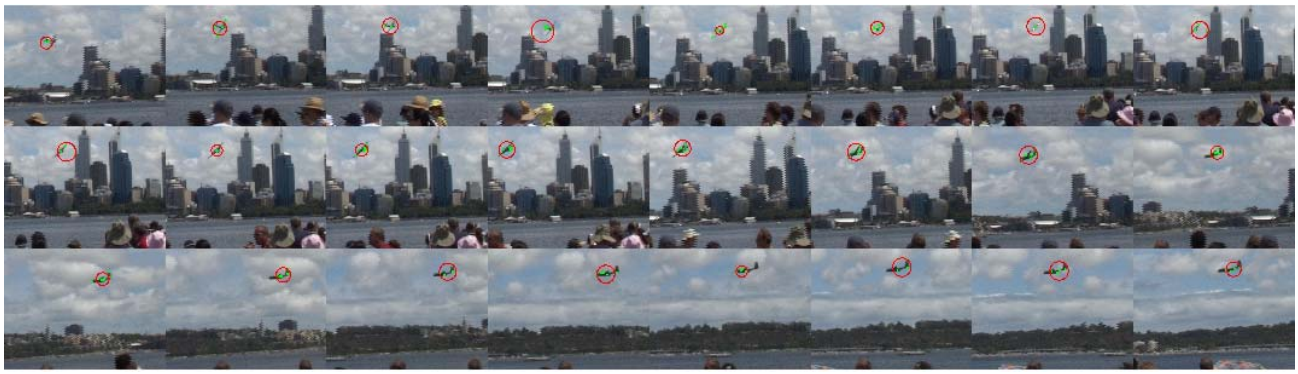


Figure 2. Tracking sequence of an aircraft over the Perth city skyline. All frames are cropped from 720×576 to 400×300 for brevity.

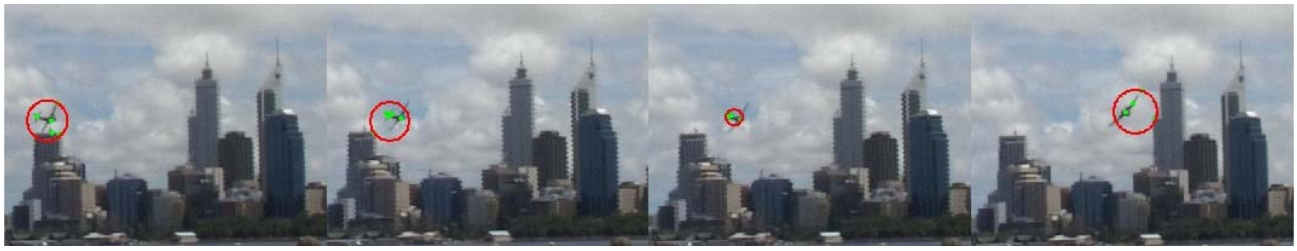
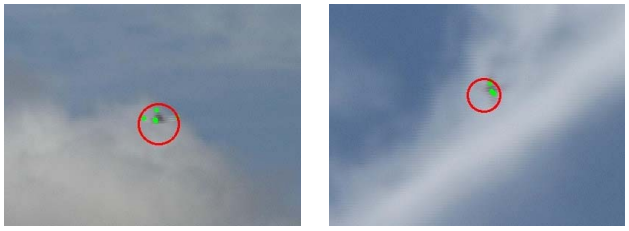


Figure 3. Selected frames from the tracking sequence of Fig. 2 show the effect of nearby objects on tracking. All frames are cropped from 720×576 to 400×300 for brevity.

image (but still contains the aircraft) due to a feature point detected on the building but in the next image the circle gets perfectly centered on the object.

Fig. 4 shows two sample frames from a challenging tracking sequence where an aircraft performing aerobatics is tracked. In both frames, the aircraft is partly occluded by clouds but the tracking algorithm still accurately tracks the aircraft. Notice that the aircraft scale is also very small in both frames. Our results show that in the absence of background clutter, a single feature point is sufficient to track the aircraft.



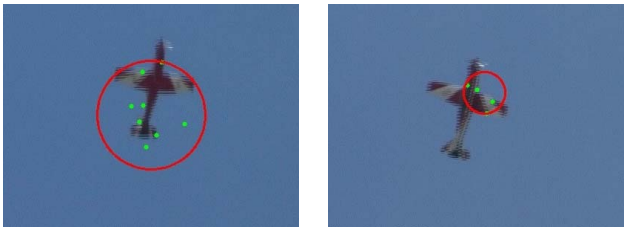
(a) (b)

Figure 4. Two challenging cases where an aircraft, performing aerobatics, has been correctly tracked even though it was partly occluded by clouds. Frames are cropped from 720×576 to 400×300 for brevity.

6. Limitations and Future Work

Recall that the foundation of tracking algorithm lies in the assumption that there is little motion between consecutive frames. In other words, the displacement between frames should be very small compared to the window size ($\mathbf{d} << W$). If this condition is violated, which is possible for high speed objects like aircrafts, the algorithm will fail resulting in the loss of all local features (and hence the tracked object). This situation is further exacerbated when the camera (being hand held as in our case) moves opposite to the direction of the object. Other sources of error are feature points detected outside the object (see Fig. 5-a) and feature points not covering the entire object (see Fig. 5-b). These cases can introduce error in the exact location and scale of the tracked object. In our future work, we intend to overcome these errors by identifying the boundary of complex objects (such as aircrafts) and forcing the local feature identification to within the object boundary.

Another implicit assumption in the proposed algorithm is that the number of identified feature points on the tracked



(a) (b)

Figure 5. Two cases of error. (a) Some feature points have been extracted that do not belong to the object resulting in a shift in the location of the circle. (b) Feature points have been identified on part of the object resulting in an incorrect scale of the object. Frames are cropped from 720×576 to 400×300 for brevity.

object will always be greater than outliers. This assumption holds good when the object scale is comparable to or larger than the feature window W. Even when the object is a few pixels in size and there is no background clutter, tracking is accurate. However, when the object scale is very small and when there is background clutter, the tracker (red circle) shifts to a background object. This situation and the case of fast moving objects can be addressed by introducing a dynamic state model into the tracking algorithm to better predict the location of the object in the next frame and to avoid shifting of the tracker to a background object by not allowing sudden changes in the state model.

Our results show that it is usually a combination of two or more challenges discussed in Section 4 that cause the tracker to fail e.g. background clutter plus very small scale of the object, high speed plus specular reflection plus very small scale of the object. Any single tracking algorithm may not be able to deal with all the challenges at the same time however, using a combination of tracking algorithms, complementing one another, it is possible to achieve more robustness to these factors.

7. Conclusion

We presented a modified KLT algorithm for tracking one or more objects. The tracking algorithm is based on local features and continuously updates the features while tracking. This makes it robust to global and local changes in the object's appearance due to pose, scale and illumination variations. A feature clustering criterion is used to remove outlier features while tracking. The algorithm was tested on a real and challenging video database of various aircrafts in flight and performing aerobatics. This data is the first of its kind and, as an additional contribution, it will be made freely available to the research community.

Acknowledgments

The author would like to acknowledge Professor R. Owens and Professor M. Bennamoun for their support. This research was sponsored by ARC discovery grant DP0881813 and partly by UWA research grant 2007.

References

[1] J. Barron, D. Fleet, and S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, 1994.

[2] A. Baumberg and D. Hogg. Learning flexible models from image sequences. In *European conference on Computer vision*, pages 299–308, 1994.

[3] L. Bretzner and T. Lindeberg. Feature tracking with automatic selection of spatial scales. *Computer Vision and Image Understanding*, 71(3), 1998.

[4] Intel. Open computer vision library. *http://www.intel.com/technology/computing/opencv/, last accessed in July, 2008.*

[5] M. Isard and A. Blake. Condensation - conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.

[6] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. *International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.

[7] G. Manku, P. Jain, A. Aggarwal, and L. Kumar. Object tracking using affine structure for point correspondence. *IEEE CVPR*, pages 704–709, 1997.

[8] L. Matthies, T. Kanade, and R. Szelisky. Kalman filter based algorithms for estimating depth from image sequences. *International Journal of Computer Vision*, 3:209–236, 1989.

[9] P. Maybeck. *Stochastic Models, Estimation and Control*. Academic, London, UK, 1979.

[10] A. Mian. Home page. *http://www.csse.uwa.edu.au/ ajmal/.*

[11] J. Shi and C. Tomasi. Good features to track. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[12] C. Tomasi and T. Kanade. Detection and tracking of point features. *Carnegie Mellon University Technical Report CMU-CS-91-132*, 1991.

[13] E. Turcco and K. Plakas. Video tracking: A concise survey. *IEEE Journal of Oceanic Engineering*, 31(2):520–529, 2006.

[14] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *IEEE CVPR*, 2001.