Visual Tracking at Sea*

Charles Bibby and Ian Reid
Dept of Engineering Science
University of Oxford
Oxford, OX13PJ, UK
charles.bibby@worcester.ox.ac.uk, ian.reid@eng.ox.ac.uk

***Abstract*— We describe progress in the design, build and test of a mechatronic system capable of visually tracking objects at sea from a moving platform, such as a boat. The mechanism comprises three controllable degrees of freedom that can carry a small camera as payload. The camera position is stabilised using inertial sensing, and the stabilised images are processed using a published colour-based tracking algorithm to achieve visual pursuit of the target [1], [3]. We describe novel improvements to the tracker that enhance its performance in the case of tracking a target at sea. The system is demonstrated on real footage and its performance assessed.**

***Index Terms*— Visual tracking, Active vision, Visual servoing**

I. Introduction

Keeping a fix on a target at sea can be demanding. It is common practice in the marine community for a person to keep pointing at a target so as not to lose it. The UK's Royal National Lifeboat Institution (RNLI) have used pan tilt cameras mounted up the mast of a lifeboat to aid their rescue missions but they still need to dedicate a full time operator to keep the camera pointing at the target.

The problem on many occasions is the amount of motion a boat will experience in even moderate seas. For example +/- 45° at $40-60^\circ$/s would not be uncommon. If this motion is present without visual reference points such as other vessels or landmarks, it can often become disorientating, making it very tough for a person to successfully follow a target.

The core contributions of the paper are two-fold: (i) we describe novel improvements to published colour-based tracking that have measurably better performance in our application; (ii) we describe the integration of the algorithm into a mechatronic system. The paper is organised as follows: we begin with a description of the visual tracking algorithm together with our improvements in section II, then describe the full system in section III and present results and evaluation from field-tests in section IV. We conclude with a discussion of a number of areas for further research.

II. Visual Tracking Algorithm

The Bhattacharyya coefficient is a measure of similarity between two probability distributions, defined for discrete distributions as

$$\rho(p,q) = \sum_u \sqrt{p_u q_u} \tag{1}$$

*This work is partially supported by Servowatch UK Ltd

It can be used for visual tracking by modelling a target by its colour histogram, q, and comparing it to the distribution $p(y)$ in an image window centered at pixel location y. This representation has the virtue of conferring a degree of invariance to deformations in shape. This property makes it an attractive measure for problems, such as ours, where the principal goal is simply repeated localisation to provide an angular demand to a servo system.

Equation (1) is iteratively maximised from a starting image location y_0 to find the new image location. Comaniciu *et al.* [3] proposed that this maximisation could be effectively and efficiently achieved using the *mean shift* algorithm. This algorithm is based on the observation that the mean of a set of samples from a probability density function, in a window of finite support, will be biased towards a local maximum of the density function.

Their derivation of this begins by linearising ρ about the current histogram

$$\rho(p(y),q) \approx \frac{1}{2}\sum_u \sqrt{p_u(y_0)q_u} + \frac{1}{2}\sum_x w_x k(y,x) \tag{2}$$

where

$$w_x = \sum_u \delta[I(x)-n]\sqrt{q_u/p_u(y_0)} \tag{3}$$

and the summation over x is implicitly over all pixels in a window, the summation of u is over all bins in the histogram, and k is a kernel that weights pixels close to the centre of the window more than at the edges (often taken to be the Epanechnikov kernel which decreases linearly to zero at the edge from 1 in the centre). The current estimate of location is refined by replacing it with

$$y_1 = \frac{\sum_x x w_x k(x,y)}{\sum_x w_x k(x,y)} \tag{4}$$

until successive moves fail to improve the objective by more than a threshold.

In various circumstances of interest, however, a standard implementation of mean shift tracking using colour histograms fails to track adequately. In particular, in the following sections we introduce three simple changes that improve the reliability in our application.

A. Using gradient information

Footage taken at sea often lacks good colour contrast, but Comaniciu's algorithm is not restricted to the use of colour information. To augment the colour information, we also compute histograms of *image gradient magnitude*,

where the image gradient is computed using standard finite-difference approximations in x and y from the monochrome image.

The sequences in figure 1 illustrate a typical scenario where there is little colour contrast, and demonstrate that the use of gradient information is beneficial.

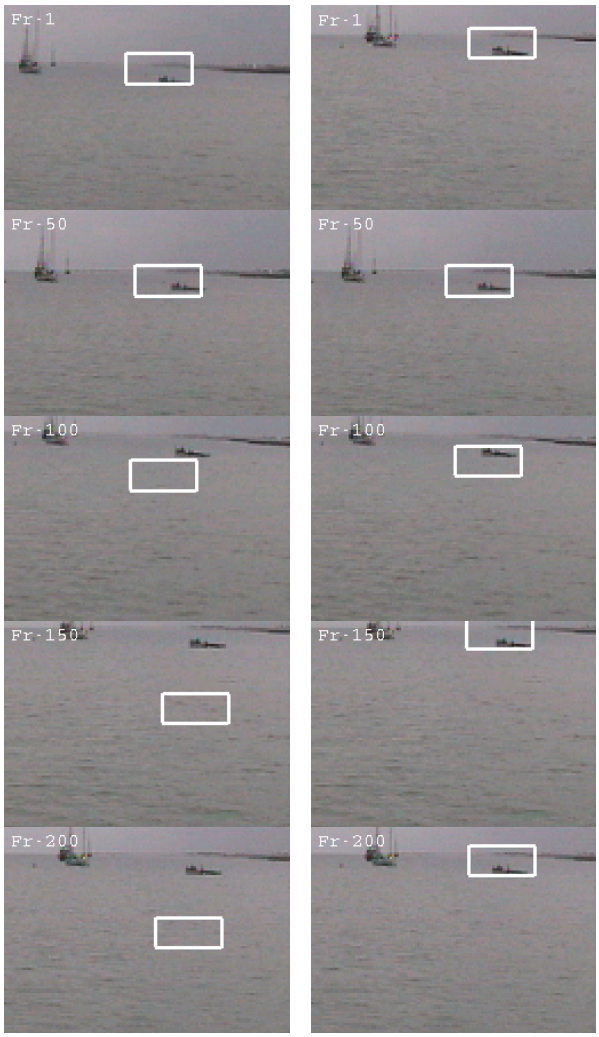

Fig. 1. Tracking in low colour-contrast images: (left) using colour histograms only; (right) using a combination of colour and gradient histograms

B. Background suppression

Other authors (e.g. [4]) have also noted the problem that a target that shares some colours with the background can be difficult to track reliably. To address this problem we have developed a method of background suppression, which also provides a metric of target suitability with respect to mean shift tracking.

Figure 2 shows how the target and background regions are constructed. The inner ellipse of radius h represents region where the values of the Epanechnikov kernel are greater than zero. The background region is the immediate neighbourhood of this target region out to a window radius of h', and we construct a colour/gradient histogram b for

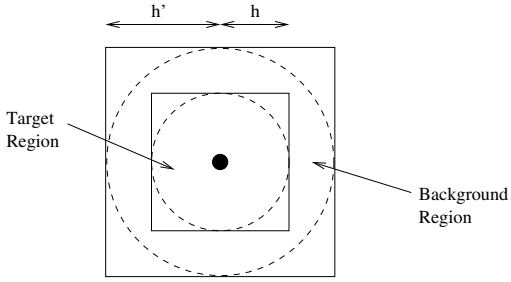

Fig. 2. Geometry of target and background models

all pixels in this neighbourhood. So that information over time is not lost we also construct a cumulative histogram, B defined such that

$$B(t) \propto b(t) + \lambda B(t-1) \tag{5}$$

where $0 < \lambda < 1$ controls the amount of memory the cumulative histogram has.

The algorithm proceeds as before but operating on image and template histograms p^* and q^* obtained by suppressing the influence of background colours:

$$q_u^* = \max(0, q_u - \mu B_u), \qquad p_u^*(y) = \max(0, p_u(y) - \mu B_u) \tag{6}$$

where $0 < \mu < 1$ is a parameter that controls the degree of suppression.

Figure 3 shows a summary of the revised algorithm. It is essentially the same as the standard mean-shift algorithm, except that (i) it operates on histograms in which the background has been suppressed; and (ii) at the end of each iteration the background is updated according to equation 5 and the current level of suppression is computed as a measure of target reliability.

Given the distribution of the target model, q, the previous cumulative background model B, and an estimate of the location in the previous frame:

1) Compute q^* the normalised template histogram according to (6)
2) Compute $p(y_0)$ and $p^*(y_0)$ the normalised histogram (and background suppressed version) at the current location estimate y_0 (6)
3) Compute the Bhattacharyya coefficient $\rho(p^*(y_0), q^*)$
4) Compute a mean shift update for the new location y_1 (4) using weights
$$w(y,u) = \sqrt{q_u^*, p_u^*(y)}$$
5) Compute $p(y_1)$ and $p^*(y_1)$
6) If $||y_1 - y_0|| < \epsilon$ stop, else set $y_0 \leftarrow y_1$ and goto step 2
7) Update $B \leftarrow b + B$ and normalise.

Fig. 3. Modified mean shift algorithm

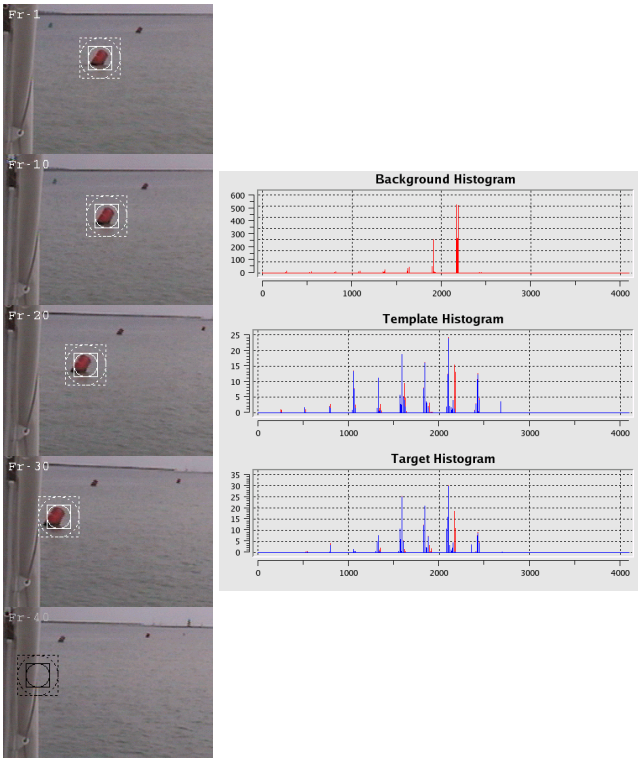

Fig. 4. (left) Tracking with background suppression active; (right) a snapshot of the histograms from a frame in the sequence

C. Coping with occlusion

If the starting point for the optimisation is outside the basin of attraction, then the target will be lost since the algorithm will converge to an incorrect local maximum. This also occurs regularly because of occlusion in the scene. Both can be ameliorated by filtering image location over time and providing predictions via a motion model (e.g. Kalman Filtering), but ultimately some form of recovery mechanism is required. Particle filtering techniques address the problem of incorrect convergence since they have the ability to represent a multi-model distribution over target location, but pay a hefty computational price.

In order to obtain the ability to recover from failure, but without the burden of a full particle filter, we sample the Bhattacharyya surface at discrete locations on a regular grid whose size is commensurate with the current scale (see figure 5). This provides a coarse approximation to the Bhattacharyya surface. Treating this as a likelihood, we combine it with a prior to obtain a coarse posterior. Since our prior is obtained from the prediction step in a Kalman Filter, the posterior is simply the likelihood weighted by a Gaussian centred on the prediction and with covariance given by the filter's prediction uncertainty.

The maximum in this coarse posterior is then chosen as the starting point for iteration of the mean shift algorithm. Figure 6 shows an example of this process in operation. Note that the full Bhattacharyya surface is shown in the figure for illustrative purposes only; during actual operation we compute only the values at the discrete grid locations to obtain a coarse sampling.

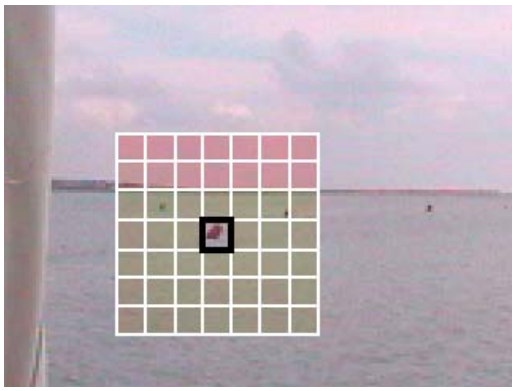

Fig. 5. The Bhattacharyya coefficient is evaluated at a set of discrete locations determined by tiling the image with windows of the current scale outwards from the current location

D. Scaling

Comaniciu's original solution to the question of a target changing scale over time was addressed via an ad-hoc test that grew and shrunk the scale of the target region by $\pm 10\%$ and chose the scale that gave the best Bhattacharyya coefficient. This method suffers from two significant problems: (i) the target is prone to collapse on itself if its distribution is relatively uniform; and (ii) the target window can explode to encompass significant parts of the background unless the background is significantly different.

Though Collins [2] provides an elegant solution via scale-space, we have adopted a simpler method. It is still based on Comaniciu's original idea, but using our coarse sampling of the Bhattacharyya surface to determine if scale changes should be allowed. If the surface has low curvature at the boundaries of the current target window, this is suggestive that either the target has started to collapse (and so parts of the full target are now considered background and outside the target window) or that the target is poorly distinguished from the background. We use a simple measure defined as the difference between the Bhattacharyya coefficient at the centre and the mean value of the surrounding samples. If this difference falls below a threshold scale changes are disabled. Figure 7 shows this technique in action.

III. System Integration

The visual tracking algorithm described in the previous section has been fully integrated into a robotic system incorporating 3 degrees of freedom (pan/tilt/roll), image stabilisation, and visual tracking. The core components of the system are the mechanism and camera, an embedded PC running Linux for visual processing, a PIC microcontroller running low-level control algorithms, and an InertiaCube (TM) inertial sensor[1]. This latter device provides relatively drift-free orientation data via fusion of twice-integrated acceleration sensing, a compass and solid-state gyros.

[1]InterSense InertiaCube^2, Intersense Inc.

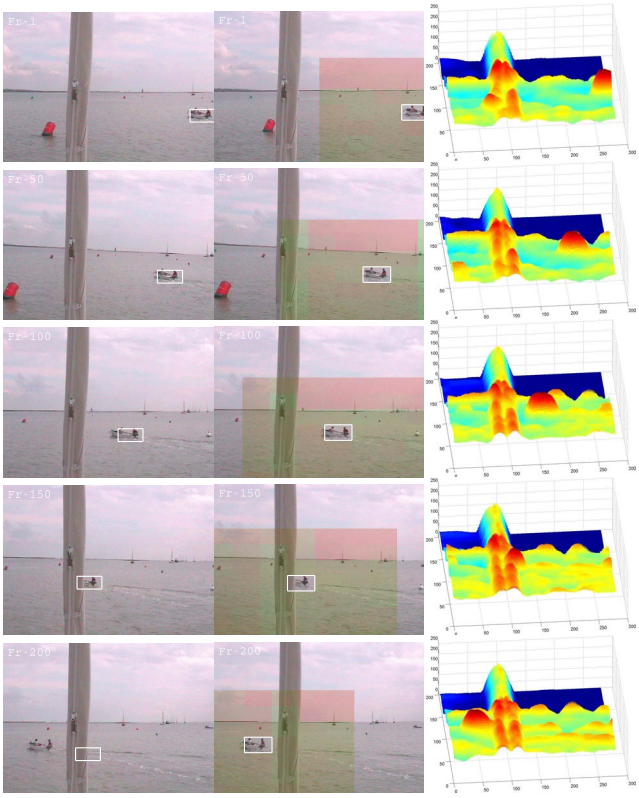

Fig. 6. Recovery from tracking failure: (left) shows typical behavior of the unmodified algorithm in the presence of occlusions; (middle) shows the recovery after complete occlusion using the coarsely sampled Bhattacharyya surface; (right) the full Bhattacharyya surface for each frame

Visual processing is performed on the embedded PC, and visual demands are sent to the PIC microcontroller which is responsible for acquiring visual and inertial sensor data, and generating PWM signals for the servos.

The controller on the PIC comprises two main control loops running at different frequencies. The first loop is to stabilise the platform and runs at 50Hz. It polls the InertiaCube for its current orientation and calculates the inverse kinematics to stabilise the orientation of the camera's natural coordinate frame.

The second loop is fired from an interrupt every time a new image frame arrives (30Hz or 25Hz depending on the camera being used). This loop calls the current tracker with the new frame where a prediction is made of the x,y coordinates that correspond to the best guess of the current target location. These coordinates are then subtracted from the centre of the frame to give two error signals which drive a non-linear controller. The outputs from this controller then make relative changes to the signals produced from the inverse kinematics.

The final demand signals are transmitted at 50Hz, from the PC to the PIC microcontroller via a serial link. The PIC microcontroller is continually reading the incoming serial data and generates the corresponding PWM signals for the actuators, thus moving the camera and closing the feedback loop for the visual controller. The reason for choosing this

Fig. 7. Changing scale: (left) scaling up; (right) scaling down

design is so that the stabilisation and the vision controllers can be kept independent from one another. This means that it is then possible to operate the system with stabilisation alone, vision alone, or both.

A. *Visual Feedback*

Design of the feedback control loop for vision proceeded empirically. A straight proportional controller does not support a high enough gain for a fast response without becoming unstable, while a proportional controller with derivative action improved matters but still provided insufficient gain for tracking fast moving targets. Finally a three stage controller with two proportional stages and a constant saturated stage was implemented (see Figure 10).

B. *Inertial Stabilisation*

The InertiaCube provides three angular measurements that describe its orientation in a global coordinate frame. It is mounted on the mechanism platform, and therefore provides the world to base-frame transformation. The base-frame to tool-frame (i.e. platform to camera) transformation

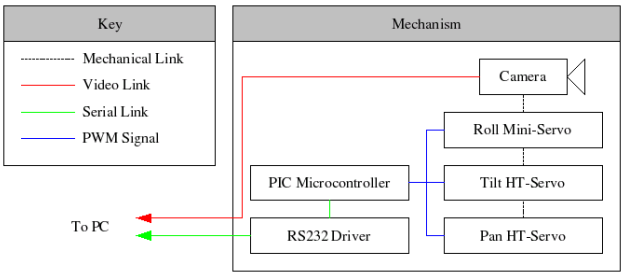

Fig. 8. PTR Mechanism

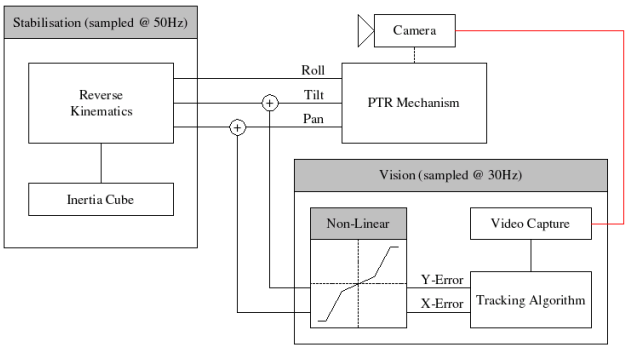

Fig. 9. System overview

R_{PC} is under control and we effectively want to keep the world-frame to tool-frame transformation R_{WC} constant. Thus the set of angles required for stabilisation comes from

$$R_{PC} = R_{WC} R_{WP}^{\top}$$

From the mechanism forward kinematics $R_{PC} = R_z(\psi) R_y(\phi) R_x(\theta)$, so after some simple algebraic manipulation we obtain

$$\begin{aligned} \phi &= \arcsin(R_{PC}(3,1)) \\ \psi &= \arctan(-R_{PC}(3,3)/R_{PC}(3,3)) \\ \theta &= \arctan(-R_{PC}(2,1)/R_{PC}(1,1)) \end{aligned} \quad (7)$$

See figure 11 for the final control algorithm.

IV. Results

Testing was conducted both on a custom built pan/tilt/roll device using small but high torque servos capable of slew rates of up to 300 $^{\circ}/s$, and a commercial pan/tilt mechanism specifically designed to operate in a marine environment (see figure 12). Figure 1 shows the use of gradient information, Figure 4 background suppression, Figure 7 scaling, and Figures 13, 14 the complete system in action.

V. Conclusions

The problem of visual tracking at sea poses some interesting questions which we have begun to investigate through the use of an active camera platform and a simple but effective colour based tracker. In order to overcome some of the specific problems encountered in visual imagery at sea we have proposed several simple but effective modifications that improve the robustness of tracking, and give a measure of confidence in our results. We have demonstrated successful tracking in several scenes including in the presence of unmodelled ego-motion which is compensated by closed loop inertial feedback control.

We have not addressed the issue of automatic target acquisition which would be crucial for search and rescue missions, and which raises a number of interesting questions, such as how to detect a salient target in the presence of constantly moving, textured background like the sea. Equally target re-acquisition after a period of occlusion (in rough water a target is likely to disappear

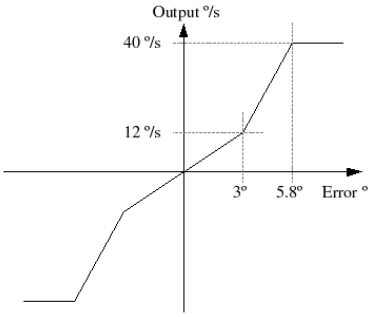

Fig. 10. The piece-wise linear controller developed for visual feedback control

At each control cycle:

1) Calculate δt, time elapsed since last update
2) Obtain R_{WP} from InertiaCube
3) Evaluate $R_{PC} = R_{WC} R_{WP}^{\top}$
4) Decompose R_{PC} into pan (θ), tilt (ϕ) and roll (ψ) using (7).
5) Find target in image and let $x_{\text{err}}, y_{\text{err}}$ be the demand in pixel coordinates
6) Compute pan and tilt demands:

$$\begin{aligned} \theta_{\text{err}} &= \arctan(x_{\text{err}} \cos\psi - y_{\text{err}} \sin\psi) \\ \phi_{\text{err}} &= \arctan(x_{\text{err}} \sin\psi + y_{\text{err}} \cos\psi) \end{aligned}$$

7) Calculate θ_d, ϕ_d using the piecewise linear controller shown in figure 10

Fig. 11. Control algorithm

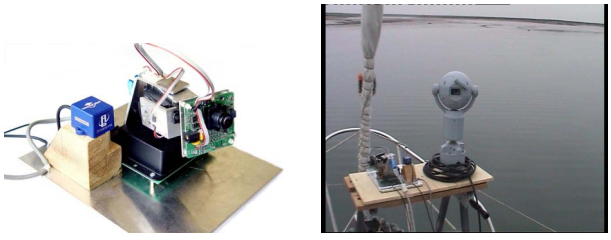

Fig. 12. Tests were performed using both (left) a simple custom built pan/tilt/roll device and (right) a commercial pan/tilt mechanism (Mic1-300 from Forward Vision CCTV Ltd (www.fvcctv.co.uk))

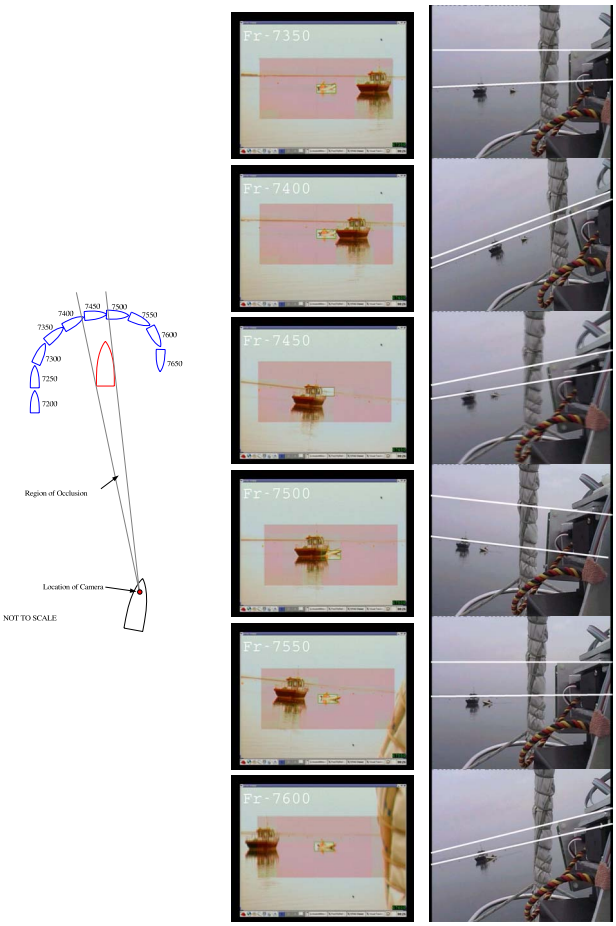

Fig. 13. Stabilisation over a short video sequence: (left) stabilised image; (right) view from a hand-held video camera, with superimposed lines to illustrate motion

into troughs for periods) needs to be addressed. Though our coarsely sampled Bhattacharyya surface provides an effective solution in some cases, further research will look at sensible motion models for typical search/rescue targets. Moreover prolonged tracking would give the opportunity to derive a much stronger appearance model of a target, enhancing the prospects of correct re-acquisition.

REFERENCES

[1] G. Bradski. Computer vision face tracking for use in a perceptual user interface. In *IEEE Workshop on Applications of Computer Vision*, pages 214–219, 1998.

[2] R. T. Collins. Mean-shift blob tracking through scale space. In *Proc. 21st IEEE Conf. on Computer Vision and Pattern Recognition, Madison, Wisconsin*, volume 2, pages 234–240, 2003.

[3] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *Proc. 19th IEEE Conf. on Computer Vision and Pattern Recognition, Hilton Head Island*, volume 2, pages 142–149, 2000.

[4] F. M. Porikli and O. Tuzel. Fast object tracking by adaptive background models and mean-shift analysis. In *International Conference On Computer Vision Systems (ICVS)*, 2003.

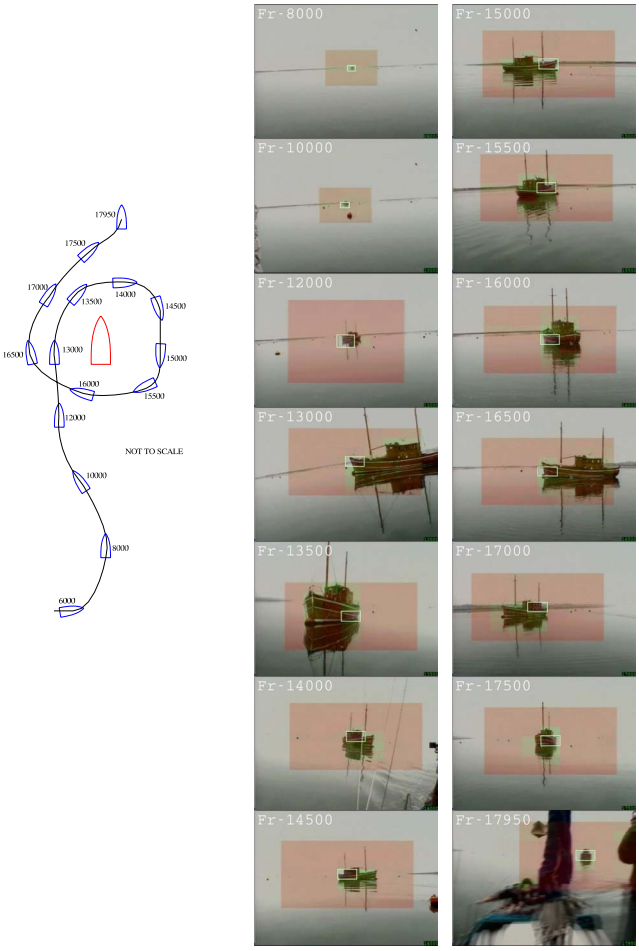

Fig. 14. Stabilisation and tracking over a 7 minute sequence