# Quantitative analysis of the improvement in omnidirectional maritime surveillance and tracking due to real-time image enhancement

Jason P. de Villiers[a,b], Asheer K. Bachoo[a, b], Fred C. Nicolls[b] and Francois P.J. le Roux[a]

[a]Council for Scientific and Industrial Research, Pretoria, South Africa;
[b]University of Cape Town, Cape Town, South Africa

## ABSTRACT

Tracking targets in a panoramic image is in many senses the inverse problem of tracking targets with a narrow field of view camera on a pan-tilt pedestal. In a narrow field of view camera tracking a moving target, the object is constant and the background is changing. A panoramic camera is able to model the entire scene, or background, and those areas it cannot model well are the potential targets and typically subtended far fewer pixels in the panoramic view compared to the narrow field of view. The outputs of an outward staring array of calibrated machine vision cameras are stitched into a single omnidirectional panorama and used to observe False Bay near Simon's Town, South Africa. A ground truth data-set was created by geo-aligning the camera array and placing a differential global position system receiver on a small target boat thus allowing its position in the array's field of view to be determined. Common tracking techniques including level-sets, Kalman filters and particle filters were implemented to run on the central processing unit of the tracking computer. Image enhancement techniques including multi-scale tone mapping, interpolated local histogram equalisation and several sharpening techniques were implemented on the graphics processing unit. An objective measurement of each tracking algorithm's robustness in the presence of sea-glint, low contrast visibility and sea clutter - such as white caps is performed on the raw recorded video data. These results are then compared to those obtained with the enhanced video data.

**Keywords:** Maritime, surveillance, tracking, real-time, enhancement, omnidirectional

## 1. INTRODUCTION

### 1.1 The need for omnidirectional surveillance

With the increase in piracy along the eastern coastline of Africa, it has become even more important to be able to monitor the immediate area around a ship. Whilst at sea, radio detection and ranging (RADAR) is able to identify threats at distance far beyond visual range. However, when the ship is off shore or anchored in a foreign harbour, there are frequently a large number of craft in the vicinity. It is possible that hostile forces can masquerade as lawful civilians and then launch a guerrilla attack once within range, as happened to the USS Cole when it was anchored at Aden, Yemen in October 2000. Activities such as this has lead to the the drafting of a white paper[1] on an integrated European Maritime Surveillance Strategy.

Detecting small craft at short ranges is a difficult task for RADAR, but this is where optical surveillance works best. Optical sensing has received increasing attention with the advent of improved imaging and processing technologies. Several European and North American companies produce products for maritime and harbour surveillance, with the cameras being mounted on static structures as well as water and aircraft. This work uses data captured with the prototype omnidirectional Wide Area Surveillance System (WASS)[2] developed by the South African Council for Scientific and Industrial Research (CSIR). This paper is one of a pair on maritime tracking, the other[3] focuses on tracking targets with a high zoom camera mounted on a pan-tilt pedestal.

Further author information: (Send correspondence to J.d.V.)
J.d.V.: E-mail: jdvilliers@csir.co.za

Acquisition, Tracking, Pointing, and Laser Systems Technologies XXV,
edited by William E. Thompson, Paul F. McManamon, Proc. of SPIE Vol. 8052, 80520F
 · doi: 10.1117/12.884052

## 1.2 Challenges of operational environment

There are many challenges to implementing an optical system in the visual spectrum outside a controlled lab environment. Some of these challenges, which are present in a harbour environment, are detailed below.

### 1.2.1 Dynamic background

A computer perceives the world only as a two-dimensional array of quantized intensity values. This array includes both the constantly moving ocean and clouds, which are both uninteresting motion and need to be detected as background. In addition, whitecaps are present on the waves and the varying surface of the sea leads to changes in the glint of the Sun reflected off of it.

### 1.2.2 Platform motion

Operationally deployed cameras are unlikely to be statically located. However, it is likely that the motion will be known (to some degree of accuracy) using Inertial Measurement Units (IMU), Global Positioning System (GPS) receivers and external inputs from the higher level systems on the ship. Examples of camera movement include:

Translation: A moving ship will see the same background objects from different angles and at different distances. Additionally, objects at different distances will appear to move at different speeds in the image.

Vibration: The engines, and possibly other systems in the ship, can cause vibration which may lead to image blurring in long exposures.

Rotation: Even when at anchor, ships experience roll and pitch disturbances due to wind and waves. Whilst moving, ships will experience yaw and roll when turning as well as pitch perturbations as waves are crested.

### 1.2.3 Uncontrolled lighting

The system has to operate in an outside environment. This means that the lighting will change:

24-hour operation: The level of ambient illumination (in the visible spectrum) changes drastically throughout the course of a day, varying 8 orders of magnitude from several milli-Lux in overcast starlight with a new moon to well over 100 kilo-Lux at midday in summer in the tropics.

Clouds: Passing clouds cause localised changes in illumination from the shadows they cast.

Shadows: The direction, length and shape of shadows of objects in the Field of View (FOV) change throughout the course of a day.

### 1.2.4 Atmospheric effects

Fog and heat shimmer cause slight variations in both the transmissivity and refraction of the air through which the cameras are viewing the scene. Fog decreases the contrast of the received images and scintillation causes variable localized blurring and spatial warping of the image data.

### 1.2.5 Environmental conditions

An operationally useful system is required to operate in a wide range of conditions, which means it has to withstand a wide range of temperatures and humidity, as well as exposure to various types of precipitation and sunlight and salt spray. Similarly, the power consumption, mass and dimensions of the system have to be appropriate for the target platforms and their mission profiles.

### 1.2.6 Wildlife

Birds and animals could be present in the system's sensory area whilst not posing an actual threat. These represent a wide gamut of possible motions, from the rapid (in pixels per second) movement of a nearby bird, to the slower movement of distant domesticated animals and humans.

#### 1.2.7 Real-time operation

It is required to process images at least as fast as they are produced by the cameras. While it is helpful to know, forensically, where and how a ship was attacked, it is much more useful to be able to take some form of corrective and preventative action. This also means that a system must be capable of running continuously (without overflowing buffers or variables, for example).

### 1.3 Contribution of this paper

The system as described above with all the challenges of the maritime environment, provides video data with which it is extremely difficult to track objects. In order to improve the tracking of objects in the video, it is necessary to either increase the complexity and effectiveness of the tracking algorithms, or to improve the quality of the data on which the algorithms are run. This paper investigates the latter option.

A variety of real-time image enhancements techniques which improve the contrast by either sharpening the image or performing a tone mapping are tested for three common tracking algorithms.

## 2. TRACKING TECHNIQUES

Three tracking techniques are investigated in this paper. Background modelling (§2.1) is used as the basis for the first two techniques discussed in §2.2 and §2.3, whereas particle filtering (§2.4) works on the raw video sequences.

### 2.1 Modelling the background

One of the primary benefits of having a 360° FOV created with a static, staring camera array is that the entire local environment can be modelled. Image features that do not conform to the model can then be designated as foreground. Due to the short focal length of the cameras, and the working range of up to 1km, the dynamic surface of the ocean can be modelled as a stationary statistical process.

It has previously been found by Szpak and Tapamo[4] that in a maritime environment as observed with a gray-scale camera, a single Gaussian model can effectively model the change in intensities of the ocean surface. Pixels whose current intensities fall more than a specified threshold number of standard deviations away from their historic mean are then deemed as foreground objects. A Chan-Vese[5] level set is used to shrink to the contour of the ship being tracked, as it ignores pixels outside of its boundary that do not touch it. Thus, within a few frames, transient erroneously detected wave motion is ignored once the level set has shrunk to be between it and the ship.

The above implicitly assumes that each pixel can be uniquely identified with a three dimensional vector in world-space. While this is not valid a valid assumption for a camera experiencing translation, rotation can be properly corrected. Duvenhage *et al.*[6] provide a Graphics Processor Unit (GPU) based real-time image stabilisation technique based on two Kanade-Lukas-Tomasi trackers[7]. If available, this purely image-based stabilisation can be seeded, replaced or augmented by inertial data.

A slight variation of the algorithm suggested by Szpak and Tapamo was implemented. Instead of a single Gaussian for each pixel, two were used but they were 50% out of phase. Each Gaussian is trained for the first half of its life and then used to perform foreground/background separation for the second half. This both ensured that the sum of the first and second moments of the intensities could not overflow, and also made the system quicker to react to transient temporary lighting changes (such as by clouds). More information on the implementation details and algorithmic enhancements that were performed to make the system real-time can be found on the authors' paper on the development of the omnidirectional system[2].

### 2.2 Least Squares Polynomial Extrapolation

An auxiliary output of the level set algorithm (§2.1) is the centroid, or intensity-weighted center of gravity of the pixels within the boundary. The primary use of this is to predict the future position of the tracked target in the next frame, which allows the level set to be shifted so that it is still centered on the target. This improves real-time performance by ensuring that the minimum number of iterations are required for the level set to converge.

More importantly, it also allows for the continued tracking of targets which are moving sufficiently rapidly such that there is no overlap of the object's position in successive frames.

This simple prediction fits the minimum Root Mean Square (RMS) error polynomial independently to the recent history of tracked/detected horizontal and vertical target positions as a function of frame number. In matrix form this is expressed mathematically as:

$$\vec{X} = (A^T A)^{-1} A^T \vec{B} \tag{1}$$

where:

$\vec{X} =$ column vector of coefficients of the desired least square error polynomial,

$$A = \begin{bmatrix} 1.0 & t_0 & t_0^2 & \dots & t_0^n \\ 1.0 & t_1 & t_1^2 & \dots & t_1^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1.0 & t_{i-1} & t_{i-1}^2 & \dots & t_{i-1}^n \end{bmatrix}, \tag{2}$$

$\vec{B} =$ column vector of recent horizontal/vertical tracked values, and

$n =$ the order of the polynomial being fitted.

Note that if relative instead of absolute timestamps are used, then the matrices $(A^T A)^{-1}$ and $A^T$ become constant and need only be calculated once. In this work the last four tracked positions are used for the extrapolation. Using relative numbering and (assumed) constant frame rate, the time stamps were given the following values: $t_0 = 4$, $t_1 = 3$, $t_2 = 2$, and $t_3 = 1$. Thus prediction of the centroid in the next frame required only evaluation of the fitted polynomial at $t = 5$:

$$P = \begin{bmatrix} 1 & 5 & 5^2 & \dots & 5^n \end{bmatrix} \vec{X} \tag{3}$$

where:

$P =$ the predicted horizontal/vertical position.

Since only four points are stored, only linear and quadratic polynomials were tested, as cubic polynomials would be an exact fit and thus fit any noise in the tracked positions and extrapolate that too, particularly if the most recent measurement was noisy.

### 2.3 Kalman Filter

Kalman filters[8] are the optimal predictors for linear systems exposed to random zero-mean Gaussian noise, which are then observed by a separate process which returns measurements also corrupted by zero-mean Gaussian noise. In this paradigm, one requires a state vector ($\vec{X}$) that encapsulates all the knowledge of the system, a state transition matrix ($\Phi$) which calculates a new state vector given the current one, and an observation model ($H$) which predicts the observed measurement based on the current state vector. Eq. 4 provides the state vector which containing the last three measurements as well as the current Newtonian velocity and acceleration estimates. Only the first three terms are directly observable.

$$\vec{X} = \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ velocity \\ acceleration \end{bmatrix} \tag{4}$$

where:

$P_n =$ the position $n$ frames before the current one.

In order to predict the next state based on the current state, the latest position is updated using the classical displacement equation $s = s_0 + ut + \frac{1}{2}at^2$. The previous first two measurements get shifted to the second and third measurements. The new velocity is simply an update using the acceleration: $v = u + at$. The new

acceleration is the change in average velocity between the first and second measurements and the second and third measurements, and so represents the acceleration when $P_1$ was taken. This is encapsulated in the state transition matrix

$$\Phi = \begin{bmatrix} 1.0 & 0 & 0 & t & 0.5t^2 \\ 1.0 & 0 & 0 & 0 & 0 \\ 0 & 1.0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.0 & t \\ t^{-2} & -2t^{-2} & t^{-2} & 0 & 0 \end{bmatrix} \tag{5}$$

where:

$t =$ the sample period.

The observation model predicts the new displacements to be the average of the current estimates, and those predicted by the displacement equation: $s = s_0 + ut + \frac{1}{2}at^2$. This is expressed mathematically as

$$H = \begin{bmatrix} 0.5 & 0.5 & 0 & 0.5t & 0.25t^2 \\ 0.0 & 0.5 & 0.5 & 0.5t & 0.25t^2 \\ 0 & 0 & 0.5 & 0.5t & 0.25t^2 \end{bmatrix}. \tag{6}$$

The noise and process error covariance matrices were empirically tuned.

### 2.4 Particle Filter

In most tracking problems the posterior distribution is non-Gaussian/non-linear. The particle filter[9,10] is a Monte Carlo method that makes use of recursive Bayesian estimation to estimate a posterior density with non-linear/non-Gaussian form. The posterior density is the probability that a set of measurements has resulted in the current state. It is represented by a set of weighted particles, where each particle is a multidimensional state, and the estimated state is obtained by a weighted sum of the particles (or the particle with the highest weight). The particle filter tracks a patch of image pixels (normalized to 31×31 pixels) undergoing an affine transform. This is similar to Jepson *et al.*[11] except that we use a rectangular patch of pixels rather than an elliptical one. The state vector is

$$\begin{bmatrix} \mathbf{x}_t & \mathbf{v}_t & \mathbf{s}_t & r_t \end{bmatrix}^T$$

where $\mathbf{x}$, $\mathbf{s}$ and $\mathbf{v}$ are the component position, scale and velocity of the target respectively. The symbol $r$ is the rotation of the target on the image plane. The dynamic state model is

$$\mathbf{x}_t = \mathbf{x}_{t-1} + \mathbf{v}_{t-1} + \mathcal{N}(\mathbf{0}, \omega) \tag{7}$$

$$\mathbf{v}_t = \mathbf{x}_t - \mathbf{x}_{t-1} \tag{8}$$

$$\mathbf{s}_t = \mathbf{s}_{t-1} + \mathcal{N}(\mathbf{0}, 0.01) \tag{9}$$

$$r_t = r_{t-1} + \mathcal{N}(0, 0.02). \tag{10}$$

State variables at time $t-1$ are stored in memory during tracking. In our experiments, the above equation works well since the velocity is updated at every frame. The noise process is assumed to be Gaussian with standard deviations (in pixels) as shown above. The vector $\omega$ is adapted so that its elements are proportional to the magnitudes of the component velocities. The minimum value for a component of $\omega$ is 4.0 pixels; an element of $\omega$ is set to the magnitude of the respective component velocity if that value is greater than 4.0 pixels. The state variables $\mathbf{x}$, $\mathbf{s}$, $r$ are used to describe the image patch. Particles are sampled using the transitional prior.[9]

The observation model is a robust error function similar to Zhou *et al.*[12] that compares a particle's patch to an image template. This template is created when a track is initialized by the user. At each time $t$, an image patch is extracted using the state vector of a particle. The state estimate is selected as the particle that most likely matches the template (i.e. it has the highest likelihood (weight)). Our function is defined as follows:

$$\rho_i(\epsilon) = \begin{cases} 1.0 & \text{if} \quad \epsilon < T \\ 0.0 & \text{otherwise} \end{cases} \tag{11}$$

In the above equation, $\epsilon = \frac{|x_i - y_i|}{d}$, where $x$ is the candidate pixel and $y$ the template pixel. The parameter $d$ is a normalizing factor and also controls the effective width of the error function. The pixel number is represented by $i$. $\rho(\epsilon)$ is computed for all pixels in the target template and then summed and normalized by dividing the sum by the total number of pixels processed:

$$\gamma = \frac{1}{N} \sum_{i=1}^{N} \rho_i \tag{12}$$

where $N$ is the number of pixels in the template. Thus, the patch likelihood is determined by $\gamma$. The threshold $T$ is currently determined empirically. For the experimental analysis, the particle filter is initialized with 300 particles and $d = 0.05$.

## 3. REAL-TIME IMAGE ENHANCEMENT

It has been noted that image restoration can improve the success rate of detection by object by humans[13]. In this work two aspects of contrast were looked at to determine if automated target tracking would improve too. Figure 1 shows an example of the enhancements considered.

### 3.1 Sharpening

Sharpening is the process of making an image appear to be more in-focus. This brings out fine detail and has been shown to increase the probability of target acquisition.[14] It also increases the apparent depth of field of the image. Two locally adaptive real-time spatial-domain sharpening algorithms[15] were implemented. Both were designed to increase the edge strength in the image without adding noise in uniform regions. The algorithms were shown to increase the image's information content as measured by the Shannon entropy. Details can be found in the paper[15], only a brief description is presented here.

#### 3.1.1 Standard Deviation Spatial Gain

The standard deviation of pixel intensities in a neighbourhood, is an indication of the presence of an edge. A damped function of the standard deviation of a $5 \times 5$ window centered around each pixel is used as spatially varying gain for standard unsharp mask sharpening. Figure 1(b) shows the results of this enhancement. The pixel specific gain, $g(x, y)$ is calculated from the local intensities, $I(x, y)$, as follows:

$$g(x,y) = ln\left(\sqrt{\frac{1}{(2n+1)^2}\sum_{i=y-n}^{y+n}\sum_{j=x-n}^{x+n}(I(i,j))^2 - \left(\frac{1}{(2n+1)^2}\sum_{i=y-n}^{y+n}\sum_{j=x-n}^{x+n}(I(i,j))\right)^2}\right). \tag{13}$$

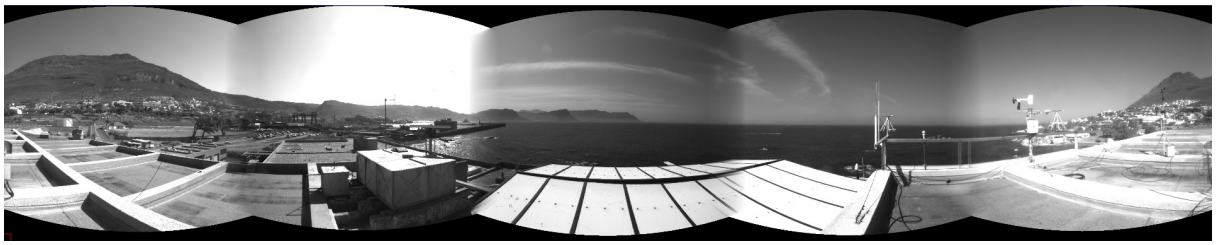
#### 3.1.2 Sobel Gradient Gain

A similar technique to detect and enhance only image regions near edges, is based on the classical Sobel operator[16]. Figure 1(c) shows the results of this enhancement. A damped version of the Sobel edge magnitude is used as follows for the variable gain:

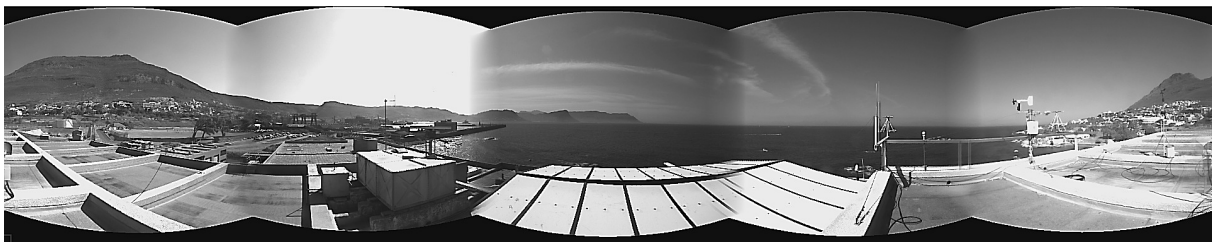
$$g(x,y) = 1 + ln\left(\sqrt{\left(\frac{\delta}{\delta x}I(x,y)\right)^2 + \left(\frac{\delta}{\delta y}I(x,y)\right)^2}\right). \tag{14}$$

### 3.2 Tone Mapping

Tone mapping computes or uses precomputed functions to map a pixel's original grey level value to a new value. The mapping function may be derived and applied at a local or global image level. The local, or adaptive, methods are more robust to changes in illumination and can greatly improve the contrast of small image details. They are considered here despite their increased computational complexity. Two methods are outlined below. More detail can be found in the paper on their real-time implementation[17].

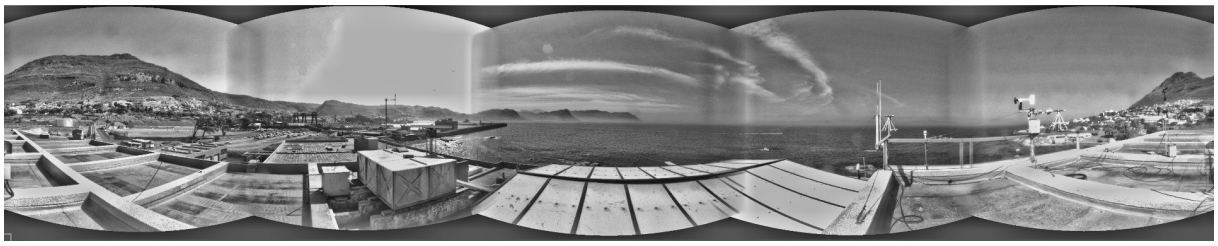


(a) Original Image



(b) Standard Deviation Gain Sharpened Image

(c) Sobel Gain Sharpened Image



(d) Interpolated Local Histogram Equalisation Enhanced Image

(e) Multiscale Gray Enhanced Image

Figure 1. Real-Time Image Enhancements

### 3.2.1 Local Histogram Equalization using Interpolation

Adaptive histogram equalization, computes a local histogram for every image pixel. However, this is a computationally intensive task. It can also be performed efficiently using bilinear interpolation[18]. The input image is divided into square regions and the histogram mapping function for each region is computed. Each pixel is then transformed by using a bilinear interpolation of the mappings in surrounding regions to approximate its mapping. Edge pixels with only two blocks in their proximity are mapped using a linear combination of the two image functions. Pixels at the corner of the image have only a single mapping function i.e. the image function of the closest block. The local histograms are computed on the central processing unit (CPU) and stored in a texture together with block location information. This information and the original image are transferred to the GPU for processing where pixel blending is executed in parallel. Figure 1(d) shows the results of this enhancement.

### 3.2.2 Multiscale Image Enhancement

Tao and Asari[19] describe a multiscale approach for enhancing images. They perform dynamic range compression and then adaptive contrast enhancement. Luminance information $I(x, y)$ is first normalized i.e. $I_n(x, y) \in [0, 1]$. Thereafter, dynamic range compression is performed to produce $I'_n$. Multiple Gaussians $G_i$ are then used to smooth the original image $I(x, y)$ at different scales, producing different blurred images $I_{G_i}(x, y)$. For each blurred image a parameter $E_i$ (which is a function of the original image's standard deviation and the ratio of the blurred and original images) is computed. $I_n{}'$ is then enhanced at multiple scales to produce the final output image:

$$S(x, y) = \sum_i 255 w_i {I_n}'(x, y)^{E_i(x,y)}, \tag{15}$$

where $w_i$ is a weight factor for each output. To achieve a high processing frame rates for this algorithm, the integral and squared integral image[20] are computed on the CPU. These together with image based parameters are transferred to the GPU for the image blurring and pixel tone mapping. Figure 1(e) shows the results of this enhancement.

## 4. EXPERIMENTAL SETUP

### 4.1 Hardware

The physical camera apparatus consisted of an outward staring array of 5 Prosilica 8-bit greyscale Gigabit Ethernet machine vision cameras with resolutions ranging from $1380 \times 1024$ to $1600 \times 1200$. Each camera had a 4.8mm 82° FOV Navitar lens. The cameras' video streams were synchronised and photogrammetrically stitched into a single omnidirectional panorama[21] of $4096 \times 820$ resolution conforming to a Mercator map projection. The geo-position of the camera array and its orientation relative to the local North-East-Down datum were measured.

All processing took place on a standard Personal Computer (PC) running Debian Linux connected to a GPS based time server. The PC had an Intel Quad Core2$^{TM}$ 2.83GHz processor, 4GB of RAM, an NVidia GeForce GTX280 GPU and a dedicated Gigabit Ethernet link to each camera. The algorithms were implemented as shaders in the OpenGL shading language (GLSL) so that use could be made of the GPU. The Framework for Live Image Transformation (FLITr)* and Open Scene Graph† was used to handle the loading of the shaders, and the download of newly received camera images to texture/video memory.

### 4.2 Data Set

Video footage of watercraft in False Bay, South Africa, was captured from approximately 25m above sea level atop a shore-side building in Simon's Town. Some of the craft had a differential GPS receiver placed aboard, whose logged data was reprojected into the time-stamped panorama to create a ground truth set of pixel positions of the boats at regular time intervals. In addition further ground truth data sets were created by manually clicking on the center of all ships in the FOV of the system. This was done for frames 1 second apart in the footage and each target also received a manual target number so as to eliminate confusion should there be multiple ships and

*FLITr provides a means to quickly implement or test image processing algorithms that use GLSL (http://code.google.com/p/flitr/).

†Open Scene Graph is a high performance open source cross platform 3D graphics toolkit (www.openscenegraph.org).

they were clicked in a different order. A differential GPS and three manually tagged ground truth data sets were used for this evaluation. Together these datasets include targets ranging from jetskis, science vessels, and rigid hulled inflatable boats (RHIB) to law enforcement craft. Figure 2 shows example 50×50 pixel regions clipped from the 360° panoramas containing the targets of interest. The targets can have a very low contrast against the ocean surface (e.g. Figure 2(c)), can be much smaller and less distinct than the wake they create and constitute only a few pixels.

(a) Yacht (b) Jetski (c) RHIB

Figure 2. Example data set images

### 4.3 Tracking Metrics

Performance metrics were derived using ideas from Senior *et al.*[22]. The following metrics are used (with symbols shown in brackets):

1. **Average error ($\mu$):** The average error for each video is computed using the estimated target position and the ground truth position. The error for a single frame is

$$error = \sqrt{(x_g - x_e)^2 + (y_g - y_e)^2}, \tag{16}$$

   where $x_g$ is the ground truth $x$-coordinate and $x_e$ is the $x$-coordinate estimated by the tracker.

2. **Standard deviation of the error ($\sigma$):** This is the standard deviation of the error over the entire video sequence.

3. **Longest tracked sequence ($l$):** This is the longest consecutive sequence of frames that is tracked. It is expressed as:

$$l = \frac{\text{length of consecutive track}}{n} \times 100, \tag{17}$$

   where $n$ is the total number of frames in the video sequence. A track ratio of 100% implies uninterrupted tracking. This metric must be assessed in conjunction with the mean errors. An estimated track breaks when the error for a frame is greater than 25.0 pixels.

4. **Number of re-initializations ($\phi$):** This is the average number of times the tracker is re-initialised, per 100 frames, so that it can complete tracking a target in a video sequence.

## 5. RESULTS

Table 1 shows the processing rates achieved for each algorithm on the hardware described in §4.1. The processing rates are given for both the input image dimensions as well as the resolution of the output stitched image.

Tables 2 through 5 provide the outcomes of the tracking improvement experiments. Each table provides the metrics for the raw recorded data, and the same footage enhanced by the standard deviation and Sobel

Table 1. Algorithm processing rates

| Algorithm | Resolution | | |
|---|---|---|---|
| | 1380 × 1024 | 1600 × 1200 | 4096 × 820 |
| | FPS | FPS | FPS |
| Standard Deviation Sharpening | 585 | 457 | 284 |
| Sobel Sharpening | 602 | 475 | 283 |
| Local Histogram Equalisation | 75 | 61 | 45 |
| Multiscale Enhancement | 30 | 22 | 12 |

gain sharpenings as well as the interpolated local histogram equalisation and multiscale grey tone mappings respectively. The metric is provided for each of the four ground truth sets. Table 2 provides the results of the linear least squares centroid extrapolation. Table 3 provides the results of the quadratic least squares centroid extrapolation. Table 4 provides the results of the Kalman filtering and Table 5 provides the results of the particle filtering.

Table 2. Tracking improvements for linear centroid extrapolation

| Algorithm | Metric | Yacht (GPS) | Jetski | Patrol Boat | RHIB |
|---|---|---|---|---|---|
| Raw Data | $\mu$ | 19.4 | 8.5 | 6.1 | 16.1 |
| | $\sigma$ | 15.0 | 5.9 | 2.3 | 5.2 |
| | $l$ | 94 | 49 | 100 | 15 |
| | $\phi$ | 0.4 | 1.3 | 0 | 2.2 |
| Standard Deviation Sharpening | $\mu$ | 17.4 | 11.3 | 8.3 | 17.0 |
| | $\sigma$ | 15.2 | 4.8 | 1.4 | 4.8 |
| | $l$ | 93 | 43 | 100 | 23 |
| | $\phi$ | 0.3 | 2.2 | 0 | 2.3 |
| Sobel Gain Sharpening | $\mu$ | 17.0 | 11.4 | 6.7 | 16.8 |
| | $\sigma$ | 12.3 | 5.1 | 2.5 | 4.8 |
| | $l$ | 100 | 46 | 100 | 20 |
| | $\phi$ | 0.0 | 2.0 | 0 | 2.5 |
| Local Histogram Equalisation | $\mu$ | 4.4 | 10.0 | 5.9 | 15.9 |
| | $\sigma$ | 3.5 | 5.3 | 2.3 | 4.3 |
| | $l$ | 100 | 50 | 100 | 26 |
| | $\phi$ | 0 | 0.9 | 0 | 2.2 |
| Multiscale Enhancement | $\mu$ | 6.3 | 11.9 | 6.1 | 15.7 |
| | $\sigma$ | 2.8 | 5.2 | 2.3 | 4.3 |
| | $l$ | 100 | 38 | 100 | 25 |
| | $\phi$ | 0.0 | 1.9 | 0 | 2.2 |

## 6. DISCUSSION OF RESULTS

From Table 1 it can be seen that the two sharpening algorithms and the local histogram equalisation algorithm can be performed in real time regardless of whether they are performed on the input pre-stitched video streams or on the output stitched panorama. The multiscale enhancement algorithm is approaching real time, and could be made real time via the use of additional hardware, so that the input images could be enhanced in parallel. Figure 1 provides example outcomes of the enhancements applied to a stitched panorama from the RHIB sequence. The effects of the two sharpening sequences are subtle in these images and can most easily be seen in the increased detail apparent on the distant buildings on the left and right of the images. The local histogram equalisation (Figure 1(d)) strongly brings out the texture in the image in both the dark and light regions. The multiscale

Table 3. Tracking improvements for quadratic extrapolation

| Algorithm | Metric | Yacht (GPS) | Jetski | Patrol Boat | RHIB |
|---|---|---|---|---|---|
| Raw Data | $\mu$ | 10.9 | 8.3 | 6.1 | 15.3 |
| | $\sigma$ | 8.2 | 5.9 | 2.3 | 5.3 |
| | $l$ | 69 | 45 | 100 | 12 |
| | $\phi$ | 0.1 | 1.9 | 0 | 2.7 |
| Standard Deviation Sharpening | $\mu$ | 9.2 | 11.7 | 8.2 | 16.3 |
| | $\sigma$ | 7.8 | 5.1 | 1.6 | 5.3 |
| | $l$ | 42 | 35 | 100 | 11 |
| | $\phi$ | 0.5 | 2.6 | 0 | 2.9 |
| Sobel Gain Sharpening | $\mu$ | 5.7 | 11.9 | 6.7 | 15.2 |
| | $\sigma$ | 4.5 | 5.0 | 2.5 | 5.1 |
| | $l$ | 66 | 30 | 100 | 16 |
| | $\phi$ | 0.5 | 2.4 | 0 | 3.2 |
| Local Histogram Equalisation | $\mu$ | 15.0 | 9.1 | 5.1 | 15.9 |
| | $\sigma$ | 11.4 | 5.3 | 2.3 | 4.6 |
| | $l$ | 87 | 27 | 100 | 17 |
| | $\phi$ | 0.3 | 2.0 | 0 | 2.2 |
| Multiscale Enhancement | $\mu$ | 9.6 | 11.5 | 6.1 | 15.4 |
| | $\sigma$ | 9.7 | 5.4 | 2.4 | 4.5 |
| | $l$ | 54 | 18 | 100 | 21 |
| | $\phi$ | 0.4 | 2.6 | 0 | 2.3 |

Table 4. Tracking improvements for Kalman Filtering

| Algorithm | Metric | Yacht (GPS) | Jetski | Patrol Boat | RHIB |
|---|---|---|---|---|---|
| Raw Data | $\mu$ | 17.0 | 6.6 | 6.1 | 14.8 |
| | $\sigma$ | 12.5 | 5.6 | 2.3 | 5.6 |
| | $l$ | 84 | 34 | 100 | 16 |
| | $\phi$ | 0.8 | 3.0 | 0 | 2.9 |
| Standard Deviation Sharpening | $\mu$ | 12.4 | 10.8 | 8.2 | 14.9 |
| | $\sigma$ | 10.6 | 5.2 | 1.5 | 6.4 |
| | $l$ | 78 | 38 | 100 | 11 |
| | $\phi$ | 0.3 | 2.8 | 0 | 3.9 |
| Sobel Gain Sharpening | $\mu$ | 8.8 | 10.9 | 6.7 | 14.0 |
| | $\sigma$ | 4.9 | 4.5 | 2.4 | 5.4 |
| | $l$ | 52 | 24 | 100 | 13 |
| | $\phi$ | 0.3 | 2.6 | 0 | 3.4 |
| Local Histogram Equalisation | $\mu$ | 14.8 | 10.2 | 5.8 | 14.8 |
| | $\sigma$ | 11.0 | 5.4 | 2.2 | 5.4 |
| | $l$ | 100 | 44 | 100 | 22 |
| | $\phi$ | 0 | 1.7 | 0 | 3.1 |
| Multiscale Enhancement | $\mu$ | 5.7 | 12.2 | 6.0 | 15.2 |
| | $\sigma$ | 2.7 | 6.6 | 2.2 | 5.0 |
| | $l$ | 100 | 14 | 100 | 22 |
| | $\phi$ | 0 | 3.9 | 0 | 2.7 |

enhancement algorithm (Figure 1(e)) has the effect of making the image seem better exposed, but induces some haloing around strong edges in the image.

Table 5. Tracking improvements for particle filtering.

| Algorithm | Metric | Yacht (GPS) | Jetski | Patrol boat | RHIB |
|---|---|---|---|---|---|
| Raw Data | $\mu$ | 5.4 | 4.1 | - | - |
| | $\sigma$ | 4.0 | 5.5 | - | - |
| | $l$ | 42 | 84 | - | - |
| | $\phi$ | 0.6 | 0.2 | - | - |
| Standard Deviation Sharpening | $\mu$ | 4.3 | 2.9 | 3.1 | 3.6 |
| | $\sigma$ | 3.1 | 3.2 | 3.8 | 3.1 |
| | $l$ | 52 | 100 | 48 | 100 |
| | $\phi$ | 0.1 | 0.0 | 0.4 | 0.0 |
| Sobel Gain Sharpening | $\mu$ | 3.8 | 5.6 | - | - |
| | $\sigma$ | 2.4 | 5.7 | - | - |
| | $l$ | 67 | 63 | - | - |
| | $\phi$ | 0.1 | 0.3 | - | - |
| Local Histogram Equalisation | $\mu$ | 5.3 | 7.7 | 1.6 | - |
| | $\sigma$ | 4.5 | 6.6 | 2.5 | - |
| | $l$ | 42 | 76 | 62 | - |
| | $\phi$ | 0.5 | 0.3 | 0.2 | - |
| Multiscale Enhancement | $\mu$ | 4.4 | 7.3 | 4.1 | - |
| | $\sigma$ | 3.1 | 7.3 | 5.2 | - |
| | $l$ | 46 | 70 | 28 | - |
| | $\phi$ | 0.2 | 0.5 | 1.2 | - |

Table 2 provides the results of the linear centroid extrapolation tracking. In general this algorithm provided the best results for all the video sequences, except that of the jetski - the tracking of which was not improved by any of the enhancements. The tone mapping algorithms both made the Yacht sequence completely trackable, and improved the pixel error of the tracking significantly. The patrol boat sequence was already completely trackable and none of the enhancements improved the pixel errors. The performance of the RHIB tracking showed only marginal improvements.

Table 3 gives the results of the quadratic centroid extrapolation. This was consistently the worst performing algorithm due to it being more sensitive to noise in the centroid determination. While the tracking results for the raw data are similar to the linear extrapolation, none of the enhancements showed any improvements to the tracking. The RHIB sequence is an exception, it showed slight improvements for the tone mapping algorithm.

Table 4 provides the results of the Kalman filter tracking. The results for the raw footage are again similar to those of the linear extrapolation, however the improvements gained from the application of the enhancement algorithms were more substantial. The tracking error for the yacht sequence was significantly improved by the Sobel and multiscale enhancements, all the algorithms improved the track ratio (if not the longest tracked sequence). The two tone mapping algorithms made the sequence completely trackable. Local histogram equalisation almost halved the number of reinitialisation required for the jetski sequence. No effect was observed on the already trackable patrol boat sequence and minor gains were found with the multiscale enhancements for the RHIB sequence.

For particle filter tracking, Table 5 shows that in all cases the standard deviation based sharpening (§3.1.2) improved the tracking performance. The length of the longest run, the tracking error and the number of reinitialisations required were improved. Indeed, this sharpening made tracking possible for the patrol boat and RHIB sequences, as it successfully enhanced the target whilst suppressing the white caps on the ocean surface. The smaller targets, in terms of number of pixels, performed worse than the larger targets especially if there was significant wake. The low contrast patrol boat sequence was made trackable by the tone mapping algorithms, with local histogram equalisation generally outperforming the multi-scale enhancement. Particle filter tracking provided the best results for the jetski sequence.

## 7. FUTURE WORK

It has been seen that in all cases the tracking results can be improved, however a means in order to automatically determine which algorithm will be the most beneficial needs to be determined. Similarly the parameters of the enhancement techniques such as the gains and window sizes needs to be automatically optimised to suit the current conditions.

It was stated that the purpose of this paper was not to optimise the tracking algorithms, but rather to see what the improvements could be made by improving the input to the tracking algorithms. However, it is possible to improve each of these tracking techniques. Both the polynomial extrapolation techniques would benefit from a longer history of points to which the curves may be fitted, so as to minimise the effects of noisy data. The Kalman filter would benefit from better determination of the initial process and noise covariance matrices, as well as using more than the minimum number of three points to determine the acceleration. The primary problem experienced by the particle filter was that the targets did not subtend enough pixels for the template matching to be robust, thus an increase in the resolution of the omnidirectional system is required for it to perform better.

## 8. CONCLUSION

Of the two sharpening techniques the standard-deviation based adaptive sharpening provided generally better results, especially for the particle filter tracking. Local histogram equalisation, in addition to being the less processing intensive of the two tone mapping algorithms provided the overall improvement in tracking performance.

It has been shown that for difficult video sequences the tracking can be substantially improved by applying real-time image enhancement techniques prior to the application of the tracking algorithms. The tracking of sequences that were initially trackable was not degraded by the application of the enhancement algorithms.

## REFERENCES


[1] European Commision: Directorate-general afor maritime affairs and fisheries, "Non-paper on maritime surveillance," (10 2008).

[2] de Villiers, J. and le Roux, F., "Omnidirectional maritime surveillance," in [*Proceedings of the CSIR 3rd Biennial Conference 2010. Science Real and Relevant.*], (2010).

[3] Bachoo, A., de Villiers, J., Nicolls, F., and le Roux, F., "Quantitative analysis of the improvement in high zoom maritime tracking due to real-time image enhancement," in [*SPIE Defense, Security and Sensing*], (2011).

[4] Szpak, Z. L. and Tapamo, J. R., "Maritime surveillance: Tracking ships inside a dynamic background using a fast level-set," *Expert Systems with Applications* **38**(6), 6669 – 6680 (2011).

[5] Chan, T. and Vese, L., "Active contours without edges," *Image Processing, IEEE Transactions on* **10**, 266 –277 (Feb. 2001).

[6] Duvenhage, B., Delport, J., and de Villiers, J., "Implementation of the Lucas-Kanade image registration algorithm on a GPU for 3D computational platform stabilisation," in [*AFRIGRAPH '10: Proceedings of the 7th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*], 83–90, ACM, New York, NY, USA (2010).

[7] Lucas, B. D. and Kanade, T., "An iterative image registration technique with an application to stereo vision," (1981).

[8] Kalman, R. E., "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME—Journal of Basic Engineering* **82**(Series D), 35–45 (1960).

[9] Arulampalam, S., Maskell, S., Gordon, N., and Clapp, T., "A tutorial on particle filters for on-line non-linear/non-Gaussian Bayesian tracking," *IEEE Transactions on Signal Processing* **50**(2), 174–188 (2001).

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

[11] Jepson, A., Fleet, D., and El Maraghi, T., "Robust online appearance models for visual tracking," **25**, 1296–1311 (October 2003).

[12] Zhou, S., Chellappa, R., and Moghaddam, B., "Visual tracking and recognition using appearance-adaptive models in particle filters," *IEEE Transactions on Image Processing* **13**, 1434–1456 (2004).

[13] Kopeika, N. S., Rotman, S. R., Taig, I., and Vander, A., "Effects of image restoration on target acquisition," *Optical Engineering* **42**(2), 534–540 (2003).

[14] Vollmerhausen, R. H., Jacobs, E., and Driggers, R. G., "New metric for predicting target acquisition performance," *Optical Engineering* **43**(11), 2806–2818 (2004).

[15] de Villiers, J., "A comparison of image sharpness metrics and real-time sharpening methods with GPU implementations," in [*AFRIGRAPH '10: Proceedings of the 7th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*], 53–62, ACM, New York, NY, USA (2010).

[16] Sobel, I., *Camera Models and Machine Perception*, PhD thesis, Stanford University, Palo Alto, California (1970).

[17] Bachoo, A. K., "Using the CPU and GPU for real-time video enhancement on a mobile computer," in [*Signal Processing (ICSP), 2010 IEEE 10th International Conference on*], 405 –408 (2010).

[18] Pizer, S., Amburn, E., Austin, J., Cromartie, R., Geselowitz, A., Greer, T., Romeny, B. T. H., Zimmerman, J., and Zuiderveld, K., "Adaptive histogram equalization and its variations," *Computer Vision, Graphics and Image Processing* **39**, 355–368 (1987).

[19] Tao, L. and Asari, V., "Adaptive and integrated neighbourhood dependent approach for nonlinear enhancement of colour images," *Journal of Electronic Imaging* **14**(4) (2005).

[20] Crow, F., "Summed-area tables for texture mapping," in [*SIGGRAPH '84*], (1984).

[21] de Villiers, J., "Real-time stitching of high resolution video on COTS hardware," in [*Proceedings of the 2009 International Symposium on Optomechatronic Technologies*], *ISOT2009* **9**, 46–51 (2009).

[22] Senior, A., Hampapur, A., Tian, Y., Brown, L., Pankanti, S., and Bolle, R., "Appearance models for occlusion handling," in [*2nd IEEE Workshop on Performance Evaluation of Tracking and Surveillance*], (2001).