Tracking Performance of a Foveated Vision System

Donald G Bailey
School of Engineering and Advanced Technology
Massey University
Palmerston North, New Zealand
D.G.Bailey@massey.ac.nz

Christos-Savvas Bouganis
Department of Electrical and Electronic Engineering
Imperial College London
London, United Kingdom
christos-savvas.bouganis@imperial.ac.uk.

Abstract—**Foveal images have variable spatial resolution, enabling a significant reduction in image size and data volume. Recently, a new class of active foveated vision system was proposed [1]. This paper examines the performance of this system for tracking a single target. It demonstrates that in the case of static targets, the fovea is able to be positioned within 1 pixel of the true target location within two frames. In the case of dynamic targets, it is demonstrated that there is a significant improvement over using a uniform low resolution image with the same number of pixels, where using a foveated image gives only a slight increase in tracking error compared with tracking the target using a high resolution image.**

Keywords—active vision, foveal vision, tracking, Kalman filter

I. INTRODUCTION

In recent years image sensors have reached very high capacity. This allows the acquisition of high resolution images, which usually has a positive impact on the overall performance of many computer vision algorithms. However, this improvement comes with an increased computational burden in processing systems when the whole information from the sensor has to be processed. For embedded vision systems, where real-time constraints have to be met, the increased data volume and computational cost make high resolution sensors less practical. Thus, there is a trade-off between high resolution and processing power.

Multi-resolution techniques are usually employed to address the above problem. In many applications, such as tracking and pattern recognition, it is not so important to maintain the same resolution across the image sensor as to have a wide field of view and have high resolution only on specific regions of the sensor. Recently, space-variant or foveating image sensors have been introduced that address the problem in its origin. These sensor architectures have variable spatial resolution across the surface of the sensor targeting data reduction without a severe impact to the final performance of the application. Active vision techniques are then used to ensure that the high resolution part of the sensor corresponds to the region of the scene where it can be most effective. This is usually in the centre of the region of interest.

A range of foveation techniques have been considered for tracking applications. In [2] Xue and Morrell demonstrated in 1D that a foveated sensor was able to track an object with greater accuracy than a fixed resolution sensor. They dynamically adjusted the resolution within the fovea based on the expected position of the target.

An alternative approach was taken by Cui et al [3] in that they used two sensors. A panoramic camera was used to give the entire region at a low resolution. A separate camera was then panned and tilted to obtain a high resolution image of the target. The disadvantage of this approach is that it requires multiple cameras, and the tracking speed is limited by the mechanics of the pan-tilt head.

A single camera solution, inspired by the log-polar map of the human visual system, was proposed by Martinez and Altamirano [4]. It used a conventional Cartesian camera, and applied a mapping in hardware to obtain a significantly smaller image with variable acuity. By moving the centre of the mapping within the original image, the target being tracked was maintained within the fovea. They demonstrated that a data reduction by a factor of 22 can be achieved without significant degradation in the performance of their tracking algorithm.

More recently, we have proposed a foveated vision system based on an off-the-shelf high-resolution (3M or 5M pixel) CMOS sensor combined with an FPGA that performs the mapping to a foveated image [1]. From within the high resolution frame, only data from within a 512x512 window was read out. Repositioning the window within the frame under programme control from one frame to the next provided the equivalent of a very fast pan and tilt. An FPGA then reduced the data within the window to a 64x64 image (a data reduction factor of 64) using a foveal mapping. The architecture for this system is shown in Fig. 1.

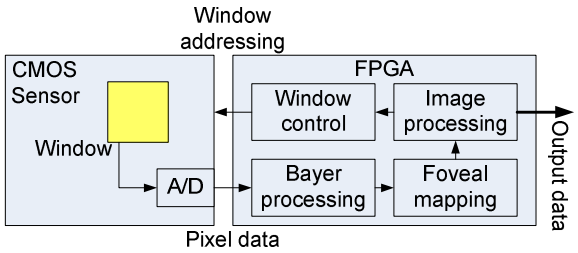

Figure 1. Proposed system architecture.

A range of foveal mappings was presented in [1]. Let the coordinates relative to the centre of the input window or foveated output image respectively be defined as (x_u, y_u) and (x_f, y_f). Let u be the distance from the centre of the input window and f be the distance from the centre of the foveated

image. The foveal mapping can then be defined in terms of the reverse mapping

$$u = \mathrm{map}_r(f). \tag{1}$$

Three types of mapping were defined, based on different definitions of distance. A radial Euclidean mapping, using the L_2 distance metric is defined as

$$u = \sqrt{x_u^2 + y_u^2} = \mathrm{map}_r\left(\sqrt{x_f^2 + y_f^2}\right). \tag{2}$$

With a radial mapping, the angle of the point relative to the centre of the image is unchanged. This means that the x and y coordinates are both scaled by the magnification for the given radius:

$$\begin{aligned} x_u &= x_f \frac{\mathrm{map}_r(f)}{f} \\ y_u &= y_f \frac{\mathrm{map}_r(f)}{f} \end{aligned} \tag{3}$$

A second type of mapping uses the L_∞ or chessboard distance metric, again with radial scaling using (3):

$$f = \max\left(|x_f|, |y_f|\right). \tag{4}$$

A third type of mapping transforms x and y independently giving a separable mapping:

$$\begin{aligned} x_u &= \mathrm{map}_r(x_f) \\ y_u &= \mathrm{map}_r(y_f) \end{aligned} \tag{5}$$

Of these three types of mapping, it was suggested that the L_∞ mapping provided the best compromise between accuracy and computational complexity [1].

This paper revisits the three proposed mappings and evaluates their performance under tracking of a moving object. The paper firstly considers the problem of the accurate estimation of the position of the object, and secondly investigates the accuracy in tracking the moving object using a Kalman filter.

II. Object Tracking

In order to evaluate the effects of different foveal mappings on tracking performance, a very simple idealized scenario was defined and simulated using MatLAB. The target consisted of a white circle of constant radius against a black background. A circular target was chosen to avoid problems associated with the edges of a square target being aligned with the pixels in the foveal image. This will be the case with both the separable mapping and the L_∞ mapping, and would introduce a bias making those methods appear worse than they really are. The location of a circular object can be estimated with sub-pixel accuracy more accurately than a rectangular object aligned with the grid.

For object tracking, it is desired to maintain the fovea centred on the target as it moves through the scene. Therefore tracking is performed using the following steps:

1. The fovea is centred on the predicted position of the target, and a foveated image is obtained from the sensor. In the architecture of Fig. 1, this is equivalent to centring the image readout window on the predicted position of the target, reading out the pixels within the window and transforming them to give the foveated image.
2. The target is then detected within the foveated image. In this simplified scenario, it consists simply of thresholding at mid-grey. This corresponds to using a thresholding based detector, where each pixel is classified as belonging to the object or background. More sophisticated object detection methods are beyond the scope of this paper. The result is a binary low resolution foveated image.
3. The next step is to estimate the true location of the target, based on the detected pixels in the foveated image. Three different approaches for accomplishing this are evaluated in section III.
4. The detected location of the target is fed into the tracking system in order to predict the next location of the target. For this study, a Kalman filter was used for the tracker. Dynamic tracking performance is evaluated in section IV.

The analysis in this paper used the mapping function proposed in [1]:

$$\mathrm{map}_r(f) = f + \tfrac{7}{32} f^2 \tag{6}$$

This has a relatively small fovea, with the acuity (the size in the original image of each foveated pixel) dropping rapidly [1]. However, the central 128x128 of the 512x512 window has better acuity than uniformly downsampling the image.

III. Static Performance

Before investigating the tracking performance, it is necessary to quantify the static performance, that is how accurately the system is are able to estimate the location of an object from a single image.

A. Object Location

The main issue with detecting the location of the target in the foveated image is that this is distorted by the transform. This distortion is very evident in Fig. 2. The naïve approach would be to simply measure the centre of gravity of the detected target in the foveated image and transform this location back to the original image space (COG method). This approach is attractive because of its computational simplicity. Let P_i be the location of a detected point in the foveated image,

and T be the estimated target location. The target location is given by:

$$T_{COG} = \text{map}_r\left(\frac{\sum_i P_i}{N}\right) \tag{7}$$

However, as a result of the foveal mapping, each pixel in the detected target corresponds to a different size region in the original image. Therefore, a better approach would be to weight each pixel in the centre of gravity calculation with the area represented by each pixel in the original space, A_i, (wCOG method).

$$T_{wCOG} = \text{map}_r\left(\frac{\sum_i A_i P_i}{\sum_i A_i}\right) \tag{8}$$

The final approach is to transform the foveated image back to the high resolution space before calculating the centre of gravity (tCOG method). In practice, this calculation may be performed directly in the foveated image by transforming each point first before calculating the centre of gravity:

$$T_{tCOG} = \frac{\sum_i A_i \,\text{map}_r\left(P_i\right)}{\sum_i A_i} \tag{9}$$

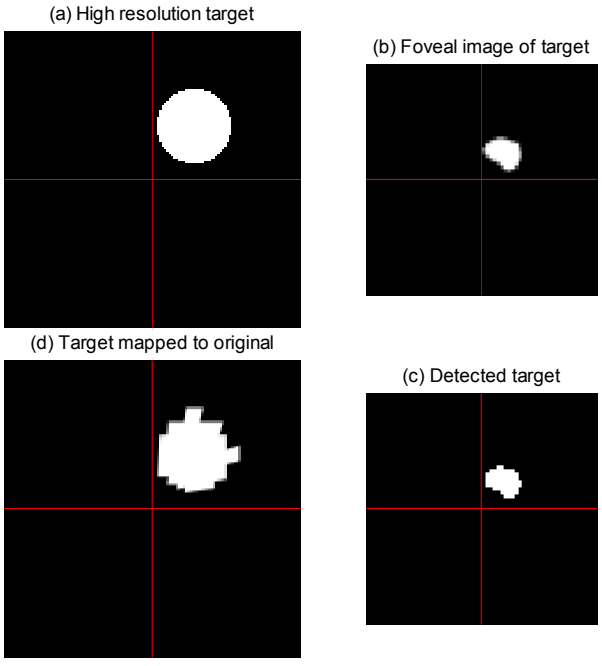

Figure 2. Distortion introduced by foveal mapping. A circular target, radius 16 pixels, at (18,23) relative to the centre of the window, is transformed using the L_∞ map. For clarity, only the central 128x128 of the high resolution windows are shown. The red cross represents the centre of the fovea.

This approach takes into consideration not only the size of each pixel in the foveated image, but also accounts for the distortion introduced by the mapping. However, it is also the most complex to calculate.

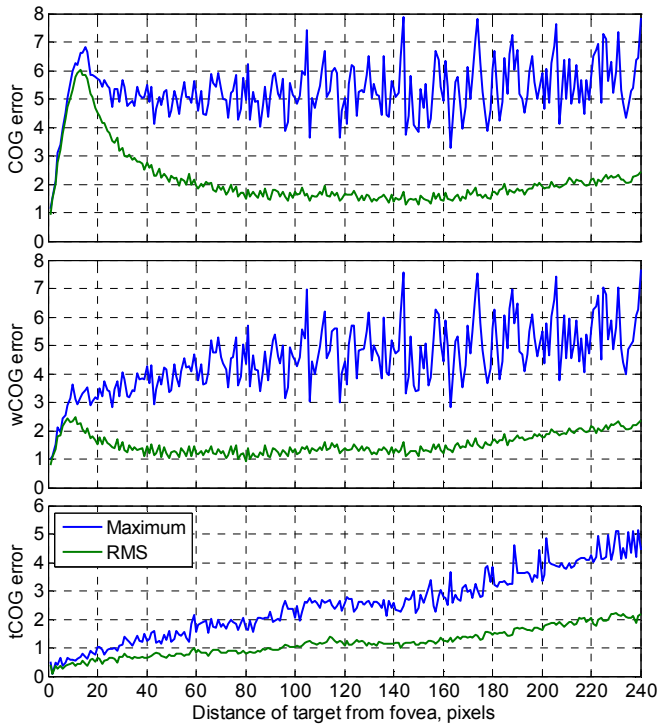

Figure 3. The effect of location calculation method on estimating target location. A circular target of radius 16 pixels was mapped using the L_∞ map.

A 16 pixel radius target was moved to every possible position in the high resolution image. The location of the target was estimated using each of the three calculation methods using the L_∞ mapping. The error between the measured locations and true locations were calculated. The errors are compared in Fig. 3 as the function of the target distance from the centre of the fovea. It is clearly seen that the COG method of estimating the target position, and to a lesser extent the wCOG method, introduce a significant bias in the result. This is most noticeable near the centre of the fovea where the target spans the greatest variation in pixel size. Giving equal weight to the pixels is effectively undervaluing pixels further from the fovea, resulting in a significant under-estimate of the position of the target. In the periphery, where the pixels are both larger (the target spans fewer pixels) and are more uniformly sized, this effect is less significant.

Weighting each foveal pixel by its effective area (wCOG) reduces this bias, but still does not eliminate it. This is because the foveal pixels are not equally spaced as is assumed by (8). Again, this effect is most noticeable in the centre of the fovea where the effective size of the pixels changes most rapidly. This is corrected by weighting the transformed location of each pixel in (9).

Since the goal of tracking is to maintain the target in the centre of the fovea, the strong bias of the COG and wCOG methods near the centre of the fovea is not good because with

successful tracking this is where the target is most likely to be located. Therefore, the more complex tCOG method is the most suited for target location.

For a target anywhere in the input window, the tCOG method is able to determine its location with a maximum error of 5 pixels (see Fig. 3). Therefore, for a static target, in the next frame the fovea can be positioned to within 5 pixels of the target, from where it will be moved to within 1 pixel in the next frame. Therefore the fovea can always be positioned over a static target within two frames.

B. *Effect of mapping type*

The effects of the different mapping types are compared with that of a uniform low resolution image in Fig. 4. There is little consistent difference that can be seen between the different mapping types. All of the foveal mappings perform significantly better than uniform downsampling within the fovea and worse in the periphery. This was expected because the central part of the foveated image (up to about 64 pixels radius) has better acuity than uniform downsampling, and outside this region, the acuity is poorer.

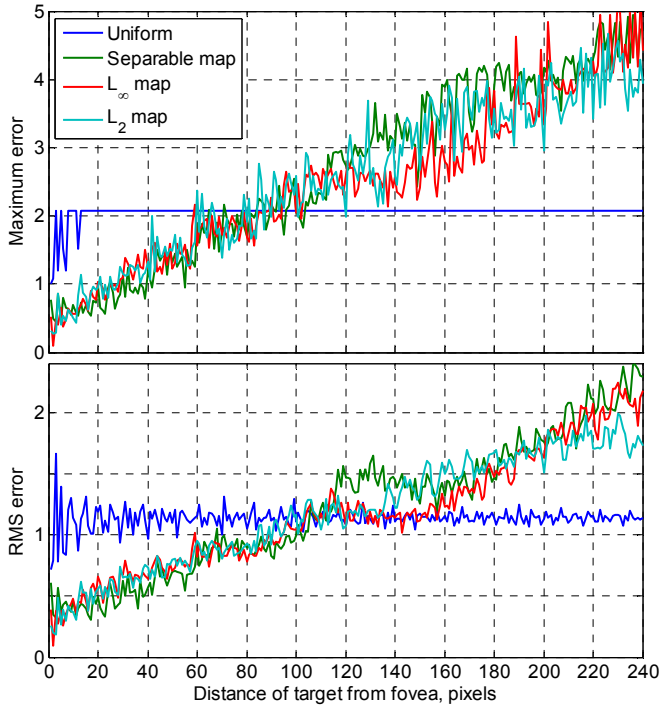

Figure 4. The effect of different maps on estimating target location. A circular target of radius 16 pixels was used. Uniform represents uniform downsampling (by a factor of 8) for comparison.

IV. Dynamic Performance

In this section, the performance of the different mapping types for tracking of a moving target is investigated. The proposed system estimates the position of the moving target in the next frame using a Kalman filter, and configures the sensor to position the fovea to the predicted target position for the next frame.

A. *Kalman filter*

The Kalman filter [5] provides an estimate of the state $\mathbf{x} \in \Re^n$ of a discrete-time process expressed under the stochastic difference equation (assuming that there is no external control):

$$\mathbf{x}_k = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{w} \tag{10}$$

using a measurement $\mathbf{z} \in \Re^m$ that is related to the state of the system through the observation matrix, $\mathbf{H}$:

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v} \tag{11}$$

The random variables $\mathbf{w}$ and v represent the process and measurement noise and it is assumed that they follow normal distributions with covariance matrices $\mathbf{Q}$ and $\mathbf{R}$:

$$\begin{aligned} \mathbf{w} &\sim N(0, \mathbf{Q}) \\ \mathbf{v} &\sim N(0, \mathbf{R}) \end{aligned} \tag{12}$$

The Kalman filter estimates the state of the process using a set of equations. These are categorized to the *time update* equations which are responsible for predicting the next state of the process, and the *measurement update* equations, which are responsible to provide a feedback to the filter using a measurement. The time update equations are:

$$\begin{aligned} \mathbf{x}_k^- &= \mathbf{F}\mathbf{x}_{k-1} \\ \mathbf{P}_k^- &= \mathbf{F}\mathbf{P}_{k-1}\mathbf{F}^T + \mathbf{Q} \end{aligned} \tag{13}$$

where $\mathbf{P}_{k-1}$, the covariance matrix, denotes the uncertainty on the state prediction. The minus superscript denotes the *a priori* estimate: that is the estimate without incorporating the new measurement. The measurement update equations are:

$$\begin{aligned} \mathbf{G}_k &= \mathbf{P}_k^-\mathbf{H}^T\left(\mathbf{H}\mathbf{P}_k^-\mathbf{H}^T + \mathbf{R}\right)^{-1} \\ \mathbf{x}_k &= \mathbf{x}_k^- + \mathbf{G}_k\left(\mathbf{z}_k - \mathbf{H}\mathbf{x}_k^-\right) \\ \mathbf{P}_k &= \left(\mathbf{I} - \mathbf{G}_k\mathbf{H}\right)\mathbf{P}_k^- \end{aligned} \tag{14}$$

The measurement update equations provide the *a posteriori* estimate by taking into account the measurement of the state. $\mathbf{G}_k$ is the Kalman gain.

In computer vision, the main application of the Kalman filter is to track an object. That is, to estimate the position and speed of the object in each frame. Under this problem, the state vector $\mathbf{x} \in \Re^4$ is given by $\mathbf{x} = [x, y, \Delta x, \Delta y]^T$, where the evolution and observation matrices are defined as follows:

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{15}$$

and $$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \quad (16)$$

B. *Tracking performance*

The performance of the different maps under Kalman filter tracking was evaluated and compared against an original high resolution image map. One thousand random paths were generated in MatLAB using the linear difference equation (10) and setting **Q** to the identity matrix. Initial experiments demonstrated that the best performance of the algorithms is obtained when **R** matrix is set to a diagonal matrix with values equal to 0.2, except for the case of the uniform image sensor with low resolution, where the best performance was obtained by setting the values to 0.35. Fig. 5 illustrates one of the generated paths, along with the predicted path from the Kalman filter using the L_2 map.

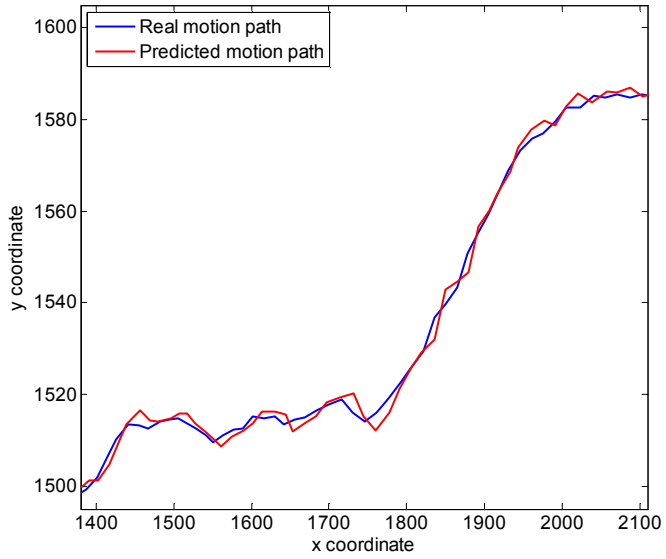

Figure 5. A generated path along with the predicted motion path using the Kalman filter under the L_2 map. The coordinates are in pixels.

For each path the maximum and RMS errors were recorded. The results from the one thousand runs were averaged, and are summarized in tables 1 and 2 for the different foveal mappings. The last row presents the results achieved when the original high resolution image is used, which acts as a baseline for reference.

The results demonstrate that the foveated sensor performs almost as well as the high-resolution uniform grid, requiring only a fraction of the pixels in order to capture the same field of view. The uniform mapping using a low-resolution image sensor (with the same number of pixels as the foveated image) performs considerably worse than the proposed foveal mappings. The above performances are explained by the fact that the target under tracking is always near to the center of the foveated sensor where the acuity is high.

The results in table 2 demonstrate that the maximum tracking error using the foveal mappings is very close to the maximum error under a uniform high-resolution image sensor. The low-resolution image sensor, which has the same number of pixels as the foveated images, performs considerably worse.

TABLE I. RMS TRACKING ERROR OF THE DIFFERENT MAPPINGS OVER A GENERATED PATH. THE CONFIDENCE IS CALCULATED FROM THE RESULTS ALONG 1000 PATHS.

Mapping	RMS error	Confidence (95%)
Uniform (low-resolution)	2.9530	±0.0070
Separable map	2.4860	±0.0056
L_∞ map	2.4310	±0.0055
L_2 map	2.4192	±0.0055
Uniform (high-resolution)	2.3899	±0.0054

TABLE II. AVERAGE MAXIMUM ERROR IN THE TRACKING OF A MOVING OBJECT USING THE DIFFERENT FOVEATED MAPS FOR 1000 RANDOMLY GENERATED PATHS.

Mapping	Maximum error	Confidence (95%)
Uniform (low-resolution)	7.089	±0.044
Separable map	5.977	±0.040
L_∞ map	5.858	±0.040
L_2 map	5.829	±0.038
Uniform (high-resolution)	5.752	±0.038

Summarizing, the obtained results demonstrate the efficiency of using the proposed foveated mappings for tracking of a moving target. In all cases, all the proposed mappings produce better results than the uniform low-resolution mapping, which uses the same number of pixels, and very close to the tracking performance achieved by the uniform high-resolution mappings. Out of all the proposed mappings, the L_2 mapping achieved the best results, although the performance for the L_∞ and separable mappings were only slightly worse.

V. FPGA IMPLEMENTATION ISSUES

The small size of the foveal image allows considerable flexibility in the detection of the target. Once the target has been detected, the tCOG calculation of (9) requires both the undistorted position and area of each foveal pixel. These may be calculated directly from the mapping function if a relatively simple mapping such as that in (6) is used. For more complex mappings, this may be too expensive in terms of FPGA resources. In this case the functionality may be implemented using a lookup table. By exploiting symmetry, for a 64x64 foveal image only two 32x32 tables are required for an L_2 mapping: one containing the effective area of the pixel, and one containing the undistorted x coordinate of the centre of the pixel. The undistorted y coordinate may be obtained from the x coordinate table by transposing the x and y addresses.

For the L_∞ mapping the tables can be even smaller, because from (4) the scaling depends only on the maximum of x and y. Therefore only three 32 element tables are required: one for the area, one for the minimum coordinate, and one for the maximum coordinate. For the separable mapping, it is even simpler, with only two 32 element tables required: one for the width (or height) and one for the coordinate. However, the effective pixel area must be calculated by multiplying the height and the width.

The problem of mapping an arbitrary Kalman filter into an FPGA has attracted the attention of the hardware design community due to the large number of applications combined with imposed real-time constraints. In [6], the authors propose a system based on the Faddeeva algorithm for the execution of matrix operations, whereas in [7] the most computationally expensive step, the Kalman gain calculation of the filter, is based on an approximation of the inverse function of the covariance matrix using Taylor expansion. Moreover, several FPGA implementations [8,9] are based on the conventional formulation of the Kalman filter and are restricted to small matrix sizes.

In this work, the Kalman filter is intended for tracking a moving object in the scene. Thus, the dimensionality of the evolution and the observation matrices are 4x4 and 2x4 respectively. Due to the small sizes of these matrices, the FPGA implementation of the Kalman filter suggested here is based on the conventional formulation. The most expensive step of the algorithm is the calculation of the Kalman gain (14), which involves the calculation of the inverse of a matrix that is updated in each iteration. However, since this matrix is only 2x2, a direct analytic calculation of its inverse in each iteration is feasible. The remaining computations include only multiplications and additions between small size matrices and vectors. Here, the fact that both **F** and **H** are sparse with the remaining elements equal to 1 also simplifies the calculations considerably.

VI. Discussion and Conclusions

The active foveated vision system is capable of tracking an object almost as well as performing the tracking in the original high resolution image. The 64 fold reduction in data volume has very little effect on the tracking ability because the target is maintained in the foveal region that exhibits high acuity.

The results from this experiment corroborate the conclusion drawn in the original paper [1] that the L_∞ foveal mapping provides a good compromise between computational complexity and performance.

The experiments described here represent an idealized case, with high contrast and no noise. Consequently they represent best case performance of the system. In practice, the target object will be more difficult to detect, particularly in the periphery of the foveal image where the acuity is lower. One of the strengths of the proposed system is that if the object drifts from the fovea, the mapping can be dynamically changed to improve the resolution in the periphery, enabling the target to be re-acquired.

Further work is required to look in more detail at the object detection of targets within real images. It is also necessary to implement the complete tracking algorithm on an FPGA to verify the expectation that only modest resources are required.

VII. References

[1] D. Bailey and C.S. Bouganis, "Reconfigurable foveated active vision system", in *International Conference on Sensing Technology*, Tainan, Taiwan (2008).

[2] Y. Xue and D. Morrell, "Adaptive foveal sensor for target tracking", in *Thirty-Sixth Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, vol. 1, pp. 848-852 (2002).

[3] Y. Cui, S. Samarasekera, Q. Huang, and M. Greiffenhagen, "Indoor monitoring via the collaboration between a peripheral sensor and a foveal sensor", in *1998 IEEE Workshop on Visual Surveillance*, Bombay, India, pp. 2-9 (1998).

[4] J. Martinez and L. Altamirano, "FPGA-based pipeline architecture to transform cartesian images into foveal images by using a new foveation approach", in *IEEE International Conference on Reconfigurable Computing and FPGA's*, San Luis Potosi, Mexico, pp .1-10 (2006).

[5] R.E. Kalman, "A new approach to linear filtering and prediction problems", *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35-45 (1960).

[6] H.G. Yeh, "Systolic implementation on Kalman filters", *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 36, no. 9, pp 1514-1517 (1988).

[7] Y. Liu, C.S. Bouganis, and P.Y.K. Cheung, "Efficient mapping of a Kalman filter into an FPGA using Taylor expansion", in *International Conference on Field Programmable Logic and Applications (FPL 2007)*, Amsterdan, The Netherlands, pp. 345-350 (2007).

[8] C.R. Lee and Z. Salcic, "High-performance FPGA-based implementation of Kalman filter", *Microprocessors and Microsystems*, vol. 21, no. 4, pp. 257-265 (1997).

[9] R.D. Turney, A.M. Reza, and J.G.R. Delva, "FPGA implementation of adaptive temporal Kalman filter for realtime video filtering", in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Phoenix, AZ, vol. 4, pp. 2231-2234 (1999).