

SIMULATION OF A RECONFIGURABLE FOVEA DESIGNED FOR OBJECT TRACKING

Achint Oommen Thomas

Dept. of Computer Science and Engineering, University at Buffalo

ABSTRACT

This project deals with the simulation of a reconfigurable fovea designed for single object tracking. Foveal parameters like gate size, number of perifoveal levels and initial fovea location were varied and the system performance in terms of object tracking accuracy versus noise was studied. The case of multiple object tracking was also looked into. Interesting observations were made but detailed analysis could not be carried out.

Keywords: Foveal vision, motion model, object tracking

I. INTRODUCTION

Biological vision systems like the human eye have an image sensor that is designed to provide high resolution in a small region of interest (ROI) within the field of view (FOV). Areas in the FOV outside the ROI are resolved at a lower resolution level. Artificial vision chips, that imitate this biological functionality known as foveal vision, have the advantage of reducing the amount of unwanted information that must be transferred from the image sensor to external image-data-processing circuits. Such vision systems are especially attractive for applications that involve recognition of objects and/or tracking of moving targets.

II. LITERATURE REVIEW

The construction of a foveated silicon retina is described in [6]. This low power, compact vision chip, when interfaced with a pair of motors, can perform target acquisition saccades and smooth pursuit tracking movements. The chip and associated motion mounts, mimic the functionality of the primate visual tracking mechanism by using the perifoveal region to perform target acquisition saccades and using the fovea for smooth pursuit tracking. A drawback of such a system is that the sensor configuration is static. In

order to foveate at a target within the current FOV but not within the foveal region, saccadic target acquisition needs to be performed. This involves the use of external motors, circuitry and logic, to change the orientation of the sensor.

A real time programmable, reconfigurable vision sensor is described in [1]. Using this vision chip, images can be sub-divided in real time into different spatial resolutions. This reconfigurable-vision imager is more powerful than a natural eye or a conventional foveal vision system in the following respects. The multi-resolution lattices (as in the chip in [6] above), are hard-wired, meaning that ROI location, size, or resolution parameters, cannot be changed. The high-resolution windows are typically located near the centers of the image frames, effectively restricting their FOVs and requiring mechanical pointing for target acquisition and tracking. Size, power demands, and slow responses of mechanical aiming mechanisms, make it difficult or impossible to realize low-power, compact, real-time, active vision systems. ^[1]

A number of motion models have been described in the literature. A motion model uses the state values of a system over the last t time instants, to give an estimate for the system state at time instant $t+1$. The constant velocity motion model uses information from the t^{th} and $t-1^{\text{th}}$ time

instants to predict the state of the system at time instant $t+1$. A discrete time Kalman Filter assumes that the relationship between the previous state and the current state is linear with zero-mean additive Gaussian noise. The Kalman update equations give a method of updating the state vector with the new measurements resulting in a Gaussian distribution describing the mean and variance of the state vector. The Extended Kalman Filter addresses non-linearities in the system dynamics. A Hidden Markov Model (HMM) represents the state space as a set of n discrete states $\{x^{(1)}, x^{(2)} \dots x^{(n)}\}$. These states can only be observed through the measurement vector z , which is related to the states through the state-conditional probabilistic sensor model $p(z_t|x^{(i)})$. An advantage that the HMM has over the Kalman Filter is that $p(z_t|x^{(i)})$ can be any distribution, not just a Gaussian. The state dynamics of an HMM are governed by a first order Markov assumption that says the current state depends on only the immediately previous state in time. The Particle Filter represents a probability distribution of the current state as a set of N state samples and associated scalar weights $\{(x^{(1)}, w^{(1)}), (x^{(2)}, w^{(2)}) \dots (x^{(n)}, w^{(n)})\}$. The weights sum to 1; larger weights indicate more likely states. Upon receipt of a new measurement z_t , a new set of particles can be computed. [10]

III. SYSTEM DESIGN

A. Simulation Setup

The image sequence consists of a bright foreground object against a dark background. The foreground object is simulated as an $S \times S$ square blob of high gray level values corrupted with a small amount of random Gaussian noise with mean 220 and variance 10. The background is simulated as a randomly fluctuating dark $N \times N$ grid of significantly lower gray level value than the foreground object. Random Gaussian noise

with mean 100 and variance 100 is added to the dark background at every time frame, to achieve the random fluctuating texture. Note that $S \ll N$. Figure 1 shows one frame of the simulation sequence.

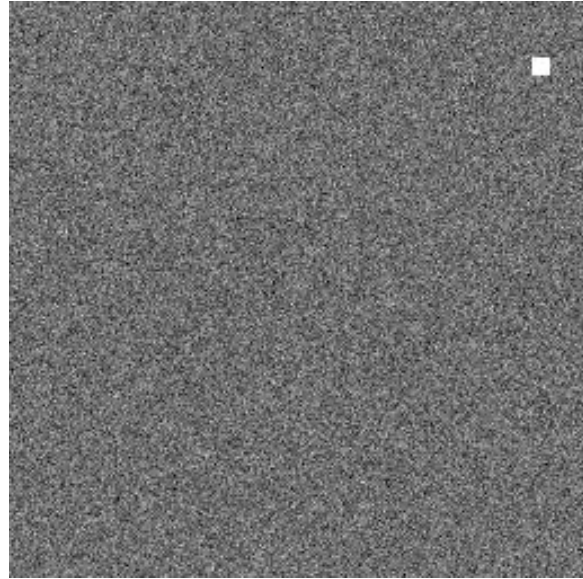


Figure 1: One frame of simulation sequence. The bright object can be seen in the top right corner.

B. Basic Approach

High level algorithm

In the algorithm, the following notation is used. O is the object to be tracked. G is the maximum foveal window size. G_m is the minimum size the foveal window is allowed to attain. (x_f, y_f) is the current foveal center. (x_o, y_o) is the current object center. F_{IMG} is the foveated image representation of a simulation frame F_s .

```

Set  $p \leftarrow 1$ 
Set  $F \leftarrow G$ 
Repeat
  Update  $(x_o, y_o)$   $p$  times and generate  $F_s$ 
  Construct  $F_{\text{IMG}}$ 
  Scan the foveal window for  $O$ 
  If  $O$  is found, then
    Set  $p \leftarrow 1$ 
    Set  $F \leftarrow \max(F-2, G_m)$ 
    Align  $(x_f, y_f)$  to  $(x_o, y_o)$ 

```

```

Update  $(x_f, y_f)$  using the motion model
Else
Scan perifoveal regions 1 to  $L$  for  $O$ 
If  $O$  is found in region  $r$  then
    Set  $p \leftarrow r$ 
    Set  $F \leftarrow G$ 
    Align  $(x_f, y_f)$  to  $(x_o, y_o)$ 
    Update  $(x_f, y_f)$  using the motion model
Else
    Randomly perturb  $(x_f, y_f)$  to  $(x_{f+\epsilon}, y_{f+\epsilon})$ 
While TRUE
    
```

n is a noise level parameter used to add jitter to the object trajectory
random is a random number generated from a Gaussian distribution with zero mean and unit variance

Figures 2 and 3 show a linear and an arbitrary trajectory respectively. Note that the black background and white foreground is for illustrative purposes only.

The steps of the algorithm are discussed in greater detail below.

Update Object Center

The object O is made to move on a trajectory defined by a set of equations. Changing the equation, changes the trajectory. Three types of trajectories have been tried for this project. The *linear trajectory* is defined by

$$y_l = mx_o + c \quad \text{-----} \quad (1)$$

while the *arbitrary trajectory* is a combination of linear, sinusoidal and parabolic trajectories defined by

$$y_o = y_l + y_s + y_p + y_n \quad \text{-----} \quad (2)$$

and

$$y_s = a \sin(\pi f x_o) \quad \text{-----} \quad (3)$$

$$y_p = \frac{b(x_o - h)^2}{4} + k \quad \text{-----} \quad (4)$$

$$y_n = n * random \quad \text{-----} \quad (5)$$

where x_o is the x -coord of the object's center
 y_o is the y -coord of the object's center
 m and c are linear trajectory parameters
 a and f are sinusoidal trajectory parameters
 b, h, k are parabolic trajectory parameters

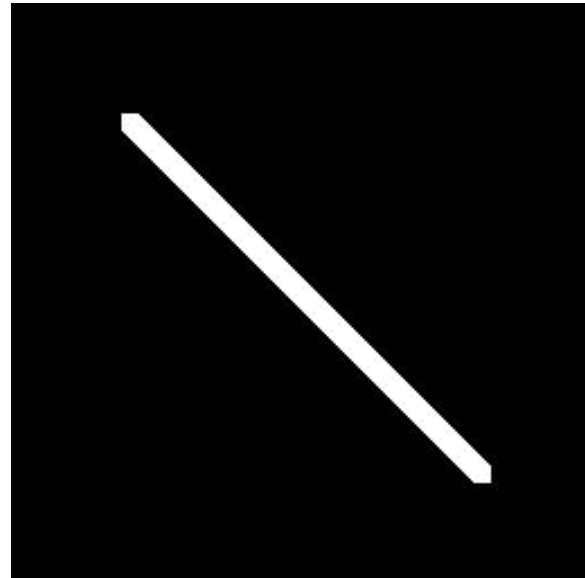


Figure 2: Linear trajectory

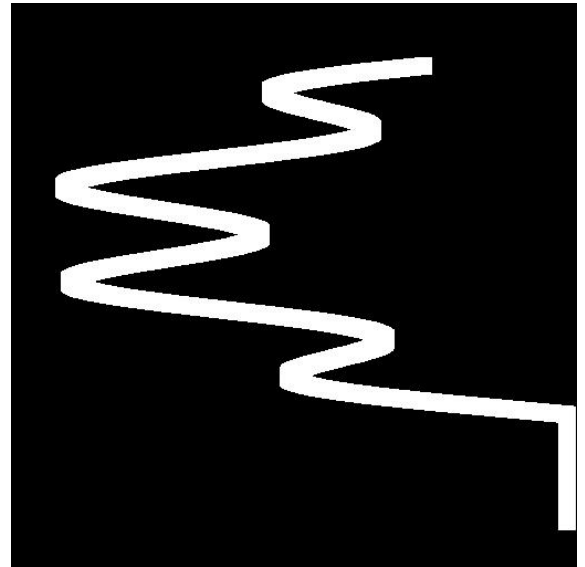


Figure 3: Arbitrary trajectory

The trajectory parameters can be used to simulate object maneuverability. Maneuverability refers to the object's ability to accelerate or decelerate rapidly and execute sharp turns. For instance, higher f values in equation (3) increases the object maneuverability. A more rapid increase in the x_0 values also results in higher object maneuverability. Figure 4 shows the same trajectory of figure 3 modified with a higher f and x_0 jump values. The number of turns the object executes has increased and the turns are now sharper. The rapid acceleration is evidenced by the larger gaps along the path. The trajectories in figures 2, 3 and 4 are called the linear, low-maneuverability and high-maneuverability trajectories respectively. Note that the black background and white foreground is for illustrative purposes only.

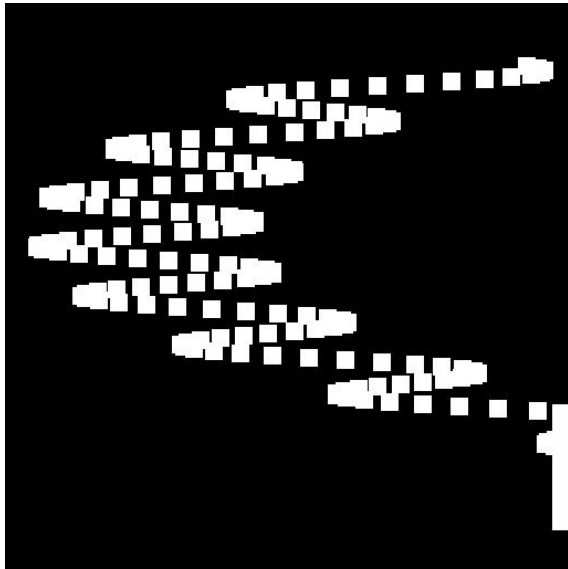


Figure 4: Modified arbitrary trajectory

Equation (5) is used to add random noise to the object trajectory. This is used to simulate non-ideal conditions including object motion unpredictability and foveal sensor noise. Figure 5 shows the same trajectory of figure 3, with noise added. The level of noise on this trajectory is 6, on a scale of 0 to 32. Note that the black background and white foreground is for illustrative purposes only.

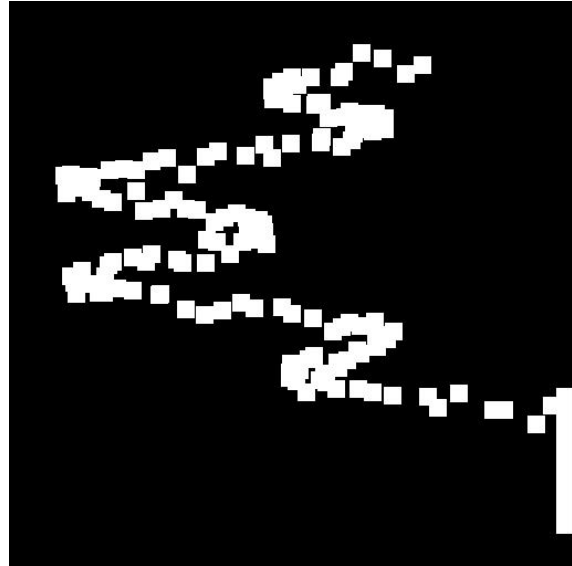


Figure 5: Trajectory with noise added

When the object center has been updated, the next frame in the simulation sequence is generated by updating the background gray level values and drawing the object O , centered at (x_0, y_0) .

A factor that must be considered is how to perform the object location update when the object has not been found in the foveal region at the preceding time instant. Not finding the object in the foveal region means that the L outer perifoveal regions have to be scanned. This means the object has that much more time to maneuver. Such a situation is taken care of in the simulation by allowing the object an extra p location updates if it has only been tracked in the p^{th} perifoveal region.

Construct foveated image representation

The Matrix-Pyramid representation is used to construct the foveated image of a simulation sequence frame. A matrix-pyramid is a set $\{M_L, \dots, M_0\}$ of $L+1$ images derived from a single $2^L \times 2^L$ gray level image M_L . M_{L-1} is derived by averaging four adjacent child nodes in M_L to compute one parent node. M_{L-1} is of dimension $2^{L-1} \times 2^{L-1}$.

¹www.cse.buffalo.edu/faculty/peter/cse573/lec/03.htm

M_L is at the highest possible resolution called foveal resolution. The resolution falls off by a factor of 2 for each successive layer in the pyramid.

The size of the foveal window, or gate, G_0 is the area of a frame viewed at maximum resolution. Successive perifoveal windows have sizes given by

$$G_l = G_0 2^{l-1} G_{l-1} \quad \text{-----} \quad (6)$$

where G_l is the size of l^{th} perifoveal window
 G_0 is the size of foveal region

Note that perifoveal window l is larger than perifoveal window $l-1$. This configuration was chosen to roughly mimic the perifoveal resolution drop-off in the primate eye.

The foveated image is constructed as follows. First the matrix-pyramid M for the current simulation sequence frame M_L is constructed. For each perifoveal level l , a window W_l of size G_l , centered at $(x_0, / 2^l y_0 / 2^l)$, is extracted from the image M_{L-1} and resized by a factor of 2^l . A composite image F_{IMG} is constructed using these windows by stacking the window W_{l-1} centered above W_l . Figure 6 shows a foveated image with the fovea at the top left.

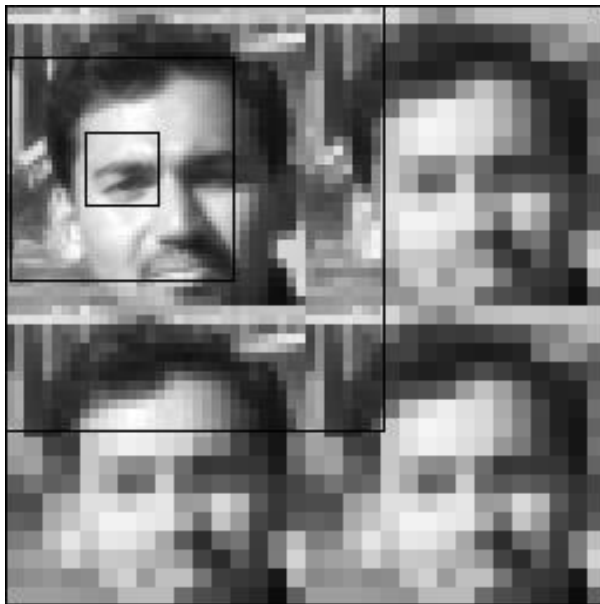


Figure 6: Foveated image

Scanning for the object (Object tracking)

There are a number of techniques mentioned in the literature to perform object tracking or for locating a given object in an image. Centroid tracking involves computing the centroid² (x_c, y_c) or center-of-mass of an object O in an image I where

$$x_c = \frac{1}{m_{00}} \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} iI(i, j) \quad \text{-----} \quad (7)$$

$$y_c = \frac{1}{m_{00}} \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} jI(i, j) \quad \text{-----} \quad (8)$$

and

$$m_{00} = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} I(i, j) \quad \text{-----} \quad (9)$$

This technique was used for tracking the object. However, it was realized that object tracking using centroid position estimation only works well for large objects taking up a significant area of the image window it is located in. For the case we have, of a small bright foreground object against an extensive, darker background, the tracker always gave false position estimates. The centroid of the entire window being searched was returned as opposed to the object center.

This prompted a shift to object-mask tracking. In this technique, a mask of the object is convoluted with the image window being searched. The pixels where the object and mask overlap, give a high output in the convolution process. So, finding this high output in a given window specifies the object center. In the foveal window, the object O of size $2^S \times 2^S$ is scanned for at maximum resolution. In the l^{th} perifoveal window, the object would be of size $2^{S-l} \times 2^{S-l}$ and hence the mask must also be of the same size. This technique is computationally more expensive than centroid tracking.

²www.cse.buffalo.edu/faculty/peter/cse573/lec/06.html

Foveal center update by using a motion model

For this project the constant velocity motion model was used. The estimate for the object center's location (x_t, y_t) at time t is given by

$$x_t = x_{t-1} + (x_{t-1} - x_{t-2})/\Delta t \quad \text{-----} \quad (10)$$

$$y_t = y_{t-1} + (y_{t-1} - y_{t-2})/\Delta t \quad \text{-----} \quad (11)$$

Since a constant frame rate is assumed, Δt is taken to be 1. The constant velocity motion model is used to update the foveal window's center at each time step. Using a motion model increases the probability of finding the object in the foveal window at each time step (for low object trajectory noise levels). If we did not estimate the foveal window center for the next instant, the object might be found only in perifoveal regions. This increases the time to re-foveate to the current object location and gives the object more time to maneuver, thus always staying one step ahead of the foveal tracking window.

If the object is found in the foveal region, it means the motion model is doing a respectable job of predicting where the object is going to appear at the next time instant. We can exploit this redundancy by dynamically shrinking the foveal window. This gives us less read-out time for the next simulation frame as we are reading a fewer number of pixels. By gradually decreasing the foveal window size, as is being done here, we are still allowing for random jitters in the object trajectory. If the foveal window size was suddenly reduced by a large value, in the next simulation frame, the object could execute an unexpected maneuver and jump out of the estimated foveal location. The foveal window will only reach its minimum allowed size if over the past k time instants, it has been able to correctly estimate the object location.

If the object is not found in the foveal region but it is found in one of the L perifoveal regions, the foveal window size is increased again to its maximum value. This ensures that

the object will be found in the foveal region at the next time instant, with a greater probability than if the foveal window size was small.

If the object is not found in the foveal region or in any of the L perifoveal regions, this means either the object has maneuvered out of the FOV of the sensor or the object-mask tracking algorithm has failed. In the case that the object has maneuvered out of the FOV, we cannot do anything but wait for it to re-appear. To eliminate the case that the object-mask tracking algorithm has made an error, we apply a small random perturbation ε to the foveal window center so as to look at a slightly different composite image, at the next time instant, to track the object.

C. Multiple Objects Tracking

The same approach described above was used to perform tracking of multiple objects in the FOV. Two objects were used, both of similar shape; bright foreground square blobs. The first object, however, was of size $S \times S$ and the second one, of size $S/2 \times S/2$. The objects were allowed to move on different trajectories. Each object was assigned its own fovea to track it as the simulation proceeded.

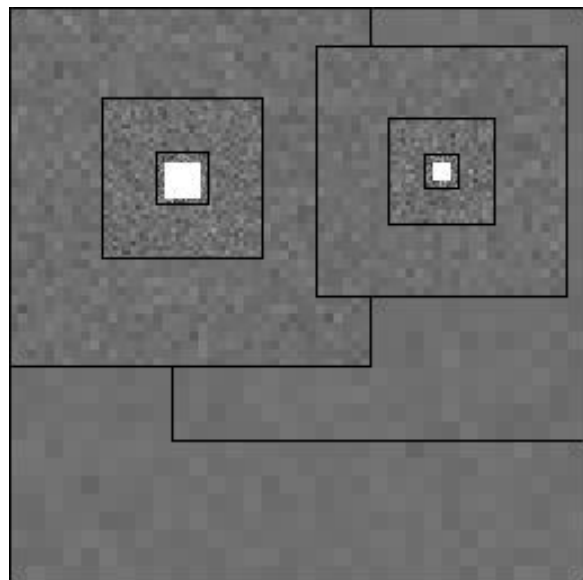


Figure 7: Multiple objects tracking

Figure 7 shows a snapshot of the simulation for tracking multiple objects, with the two bright objects clearly visible. The foveal windows were allowed to shrink and expand independently of each other as can be observed in the figure. Perifoveal boundary interaction issues were not explored in this project. The l^{th} perifoveal region for either object is always stacked above the $l+1^{\text{th}}$ perifoveal region, and hence smoothing of the boundary was not an issue.

IV. PERFORMANCE ANALYSIS

Single Object Tracking

For the performance curves described below, the following notation will be used. S is the size of the object being tracked and is taken to be 16. G is the maximum foveal window size and is taken to be 48 pixels wide. Note that tracking accuracy refers to the fraction of times the object is found in the foveal region. Figures 8 through 11 analyze the performance of the foveal configuration tracking system. Figures 12 and 13 and Table 1, compare the performance of the foveal configuration tracking system against the performance of the tunnel-vision configuration tracking system. In the tunnel-vision configuration system, the imager is constrained to have only a sible foveal region of high resolution and zero perifoveal regions. This means that, in the tunnel-vision system, if the object jumps out of the foveal region, the object is no longer present in the FOV of the system and the system goes “blind” from that point on.

The results below are taken from tests in which the initial foveal location is aligned with the initial object location. Results for the case where the foveal location is initialized randomly in the FOV are almost identical for the foveal configuration.

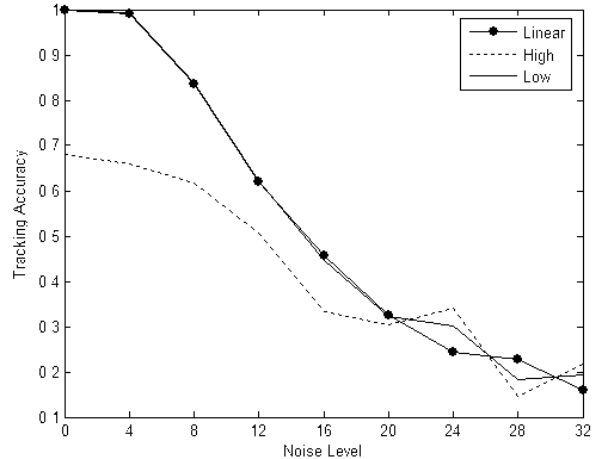


Figure 8: Tracking accuracy Vs. Noise

Figure 8 shows the plot of tracking accuracy versus noise added to the trajectory for different types of trajectories. The minimum foveal window size was set at $S + 8$.

We see that when the noise level is low, both the linear and the low-maneuverability trajectories perform much better in terms of keeping the object in the fovea as opposed to the high-maneuverability trajectory. As the noise level increases, beyond a certain threshold value ~ 20 , all trajectories fare equally bad. It is also noticed that the linear and low-maneuverability trajectories have near equal performance curves. This can be explained if we look at figures 2 and 3 again and realize that the low-maneuverability trajectory can be viewed as a combination of many separate linear trajectories joined together at the points of trajectory inflection. It is obvious that the motion model being used to update the foveal center follows the linear trajectory (at low noise levels) since they are mathematically equivalent. The fact that the arbitrary trajectory is reduced to a set of linear trajectories demonstrates the power in the simple constant velocity motion model. However, even though the high-maneuverability trajectory looks like it can be made up of a combination of separate linear trajectories, the high rates of acceleration and deceleration (see figure 4), act contrary to the prediction of the motion

model. This explains the sharp difference in performance between the linear (and low-maneuverability) trajectories at low noise levels.

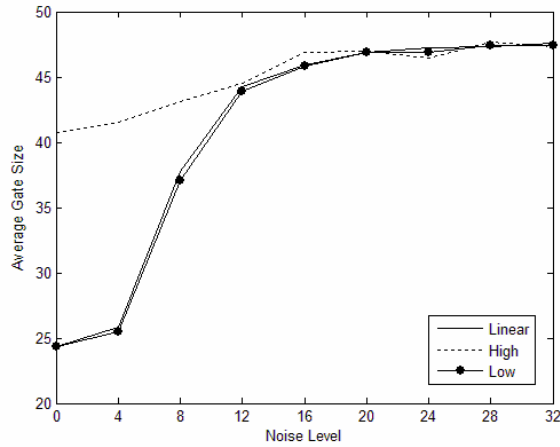


Figure 9: Avg Gate Size Vs. Noise

Figure 9 shows the plot of average gate size versus noise added to the trajectory for different types of trajectories. The minimum foveal window size was set at $S + 8$.

We see again that the curves for the linear and low-maneuverability trajectories are almost identical and they perform much better at maintaining a lower average gate size over the entire simulation as compared to the high-maneuverability trajectory curve. We also see an inverse relationship between the curves in figures 8 and 9. The lower average gate size is rendered possible since at lower noise levels, the motion model better predicts the object’s next position and the system dynamically reduces the gate size.

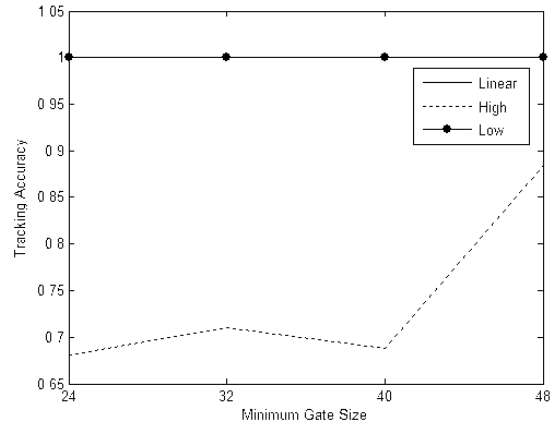


Figure 10: Tracking Accuracy Vs. Min Gate Size

Figure 10 shows the plot of tracking accuracy versus minimum gate size for different types of zero-noise trajectories.

We see that the tracking accuracy remains constant at 1 for the linear and low-maneuverability trajectories. This is expected since we are dealing with a zero-noise trajectory, something that the motion model accounts for very well. It is also observed that the tracking accuracy improves as the minimum gate size is increased from $S + 8$ to G for the high-maneuverability trajectory. We can explain this on the basis that increasing the gate size means more probability that the object would be found in the fovea at the estimated foveal position. This increased accuracy comes at a price as can be seen in figure 11.

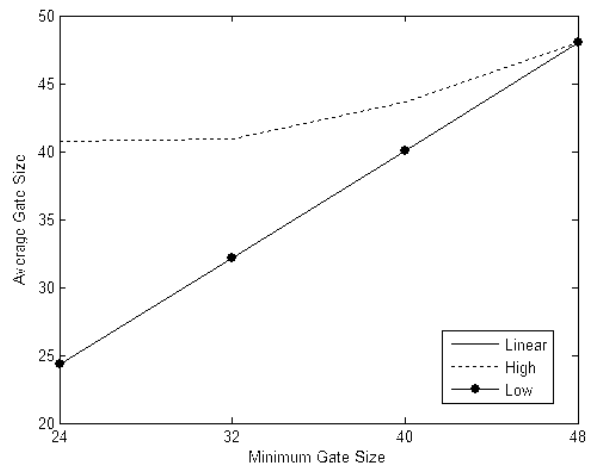


Figure 11: Avg Gate Size Vs. Min Gate Size

Figure 11 shows the plot of average gate size versus minimum gate size for different types of zero-noise trajectories.

Minimum gate size of 48 corresponds to no dynamic control of the foveal window. As the minimum gate size increases, the gate size averaged over the entire simulation also increases. This translates to more pixels being read-out at foveal resolution and calls for higher processing power. Thus a trade-off exists between the accuracy of tracking the object and the number of pixels read-out during the simulation. For the linear and low-maneuverability trajectories it makes sense to use the dynamic foveal window system as accuracy stays constant even when the foveal window size is increased.

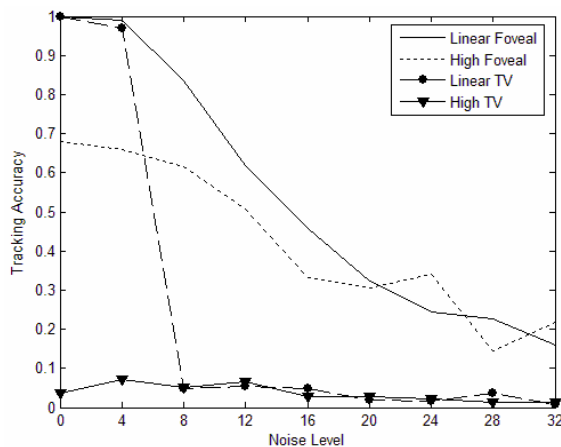


Figure 12: Tracking accuracy Vs. Noise for different tracking configurations

Figure 12 shows the plot of tracking accuracy versus noise added to the trajectory for the foveal configuration and the tunnel-vision configuration for the linear and high-maneuverability trajectories. The minimum foveal window size was set at $S + 8$.

We see that the foveal configuration outperforms the tunnel-vision configuration right from zero-noise levels for the high-maneuverability trajectory. When the object is on a linear trajectory, even at a noise level of 8 the performance drops drastically for the tunnel-vision configuration. This makes intuitive sense when we consider the fact that

the tunnel-vision tracker cannot see anything outside its highly constrained FOV, while the foveal configuration has more information to work with. Note that, in the case of the foveal configuration, even though the object is not in the foveal region, it is always found in at least one of the L perifoveal regions. This is in contrast with the tunnel-vision configuration where, if the object is not found in the FOV, the object is not in view at all.

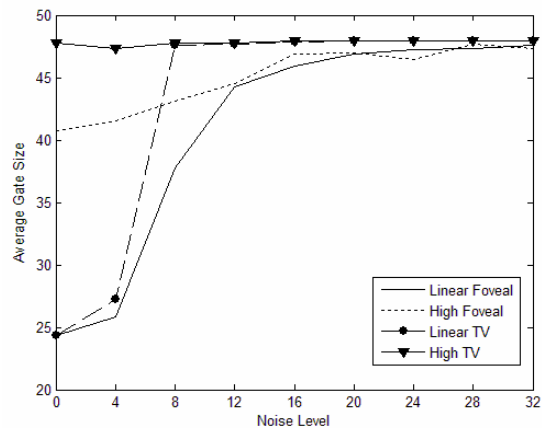


Figure 13: Avg Gate Size Vs. Noise

Figure 13 shows the plot of average gate size versus noise added to the trajectory for the foveal configuration and the tunnel-vision configuration for the linear and high-maneuverability trajectories. The minimum foveal window size was set at $S + 8$.

We see that both the foveal and tunnel-vision configurations start at low average gate sizes for the linear trajectory when the level of noise added is small. The tunnel-vision curve jumps to its maximum value soon afterwards, even as the average gate size for the foveal configuration increases much slower to saturate at its highest value beyond a certain noise threshold value ~ 20 . This can be explained based on the fact that tunnel-vision configuration loses the object easily at lower noise levels since it sees much less than the foveal configuration in terms of the available FOV.

For the high-maneuverability trajectory, once again, the foveal configuration does better at maintaining a lower average gate size over the entire simulation.

	Object			Random		
	L	HM	LM	L	HM	LM
Foveal	1	0.68	1	1	0.65	0.98
Tunnel	1	0.04	1	N/A	N/A	N/A

Table 1: Tracking Accuracy for different Foveal Locations and Trajectories. (L – Linear, HM – High Maneuverability and LM – Low Maneuverability)

Table 1 shows the tracking accuracy for different initial locations of the foveal center for zero-noise trajectories for the foveal and tunnel-vision configurations.

We see that, for the foveal configuration, the tracking accuracy drops only marginally when the foveal window is given a random initial location in the FOV. The N/A entries for the tunnel vision configuration indicate that reliable estimates could not be obtained when the foveal window is given a random initial location in the FOV for this configuration. Since the foveal window executes a movement in a random direction when it does not find the object in the FOV, once it latches onto the object (if found) it would behave like the object-centered-fovea case. However, for the tunnel vision configuration, the fovea finding the object is a random event.

Multiple Objects Tracking

As long as the object trajectories do not overlap, the multiple objects tracking problem reduces to two instances of the single object tracking problem. The issue that needs to be addressed is what to do when the trajectories overlap. It was noticed that when the objects come close and pass each other, both foveae start following the larger object and the smaller object is lost. This error could be attributed to the fact that the objects being tracked are not different enough.

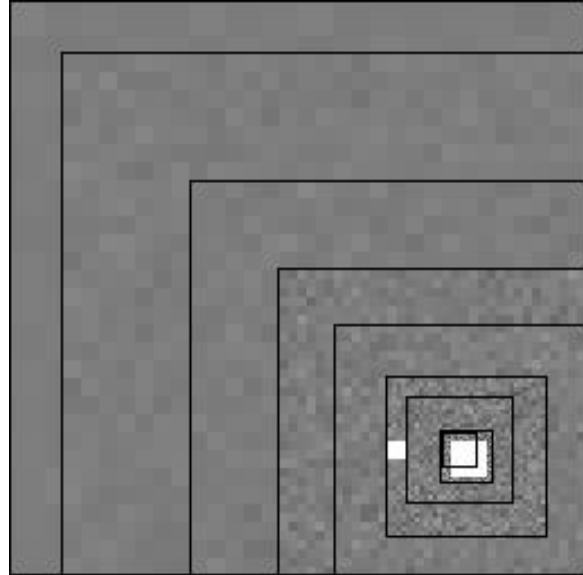


Figure 14: Tracking errors

More specifically, as can be seen in figure 14, since the fovea that is tracking the smaller object will find its target in the larger object (due to the object geometries), the tracking accuracy decreases from the point the trajectories overlap. We can clearly see both foveae tracking the larger object while the smaller object track has been lost.

An interesting observation noted was that, when the high-maneuverability trajectory was used for one object and the linear trajectory for the other, both foveae were able to successfully track their respective objects. This could be attributed to the fact that the momentum in the high-maneuverability trajectory ensures the objects do not overlap for more than a couple of simulation frames. When trajectories with lower maneuverability overlap, the foveae have more simulation frames of object overlap to get confused.

V. CONCLUSION AND FUTURE WORK

The following issues could be studied as an extension to the work carried out in this project. Here, only the simple constant velocity model was used to estimate the position of the object at the next time

instant. Incorporating more complex models such as the Kalman Filter or Particle Filters could have an impact on the tracking accuracy. A significant contribution would be a study of the interaction between various perifoveal regions as more than one object is being tracked by the system.

From the performance analysis section, we see that the foveal configuration system outperforms the tunnel-vision configuration system, both in terms of tracking accuracy as well as average gate size. Even when a small amount of noise is added to the trajectory, the tunnel-vision configuration system shows very bad performance. For the foveal configuration system, the tracking accuracy drops only when a much larger amount of noise is added.

VI. REFERENCES

- [1] Bedabrata Pain and Guang Yang, “*Real-Time-Programmable Reconfigurable-Vision Active-Pixel Sensors*”; NASA's Jet Propulsion Laboratory; <http://www.nasatech.com/Briefs/Jan02/NPO20866.html>
- [2] Capt. Amanda S. Birch, “*Biologically-Inspired Camera System Acquires and Tracks Targets at Multiple Resolutions*”; <http://www.afrlhorizons.com/Briefs/0009/MN0001.html>
- [3] W.T. Vestrand, K. Borozdin, S. P. Brumby, D. Casperson, E. Fenimore, M. Galassi, G. Gisler, K. McGowan, S. Perkins, W. Priedhorsky, D. Starr, R. White, P. Wozniak, and J. Wren, “*Searching for Optical Transients in Real-Time: The RAPTOR Experiment*”; Los Alamos National Laboratory
- [4] Laura Lit, Douglas Cochran, and Ross Martint, “*Target Tracking with an Attentive Foveal Sensor*”
- [5] Y. Xue and D. Morrell, “*Adaptive Foveal Sensor for Target Tracking*”; Conference Record of the 36th Asilomar Conference on Signals, Systems, and Computers, November 2002, pp. 848–852.
- [6] Ralph Etienne-Cummings, Jan Van der Spiegel, Paul Mueller, and Mao-zhu Zhang, “*A Foveated Silicon Retina for Two-Dimensional Tracking*”; IEEE Transactions on Circuits and Systems—II: Analog and Digital Signal Processing, Vol. 47, No. 6, June 2000
- [7] Winnifred Wong, Richard I. Hornsey, “*Calibration of a Saccadic Camera System to Adapt to Lens Distortions*”
- [8] Fengjun Xi and Darryl Morrell, “*Tracking Multiple Closely Spaced Targets Using an Adaptive Foveal Sensor*”
- [9] Ya Xue and Darryl Morrell, “*Target Tracking and Data Fusion using Multiple Adaptive Foveal Sensors*”
- [10] John Krumm, “*Probabilistic Inferencing for Location*”; Proceedings of the 2003 Workshop on Location-Aware Computing, October 2003