Tracking Subpixel Targets with Critically Sampled Optics

James Lotspeich and Mathias Klsch

Department of Computer Science, Naval Postgraduate School, 1 University Way, Monterey, CA, U.S.A.

Keywords: Subpixel, Maximum a Posterior, Tracking, Viterbi, Distance Transform, Pixel Matched Filter, Template Matching.

Abstract: In many remote sensing applications, the area of a scene sensed by a single pixel can often be measured in squared meters. This means that many objects of interest in a scene are smaller than a single pixel in the resulting image. Current tracking methods rely on robust object detection using multi-pixel features. A subpixel object does not provide enough information for these methods to work. This paper presents a method for tracking subpixel objects in image sequences captured from a stationary sensor that is critically sampled. Using template matching, we make a Maximum a Posteriori estimate of the target state over a sequence of images. A distance transform is used to calculate the motion prior in linear time, dramatically decreasing computation requirements. We compare the results of this method to a track-before-detect particle filter designed for tracking small, low contrast objects using both synthetic and real-world imagery. Results show our method produces more accurate state estimates and higher detection rates than the current state of the art methods at signal-to-noise ratios as low as 3dB.

1 PROBLEM DESCRIPTION

A subpixel target is one whose physical dimensions are smaller than the spatial resolution of the sensor. This definition assumes that the sensor is diffraction limited with pixel size determined by the Nyquist Rate. This critical value is commonly used in telescope and satellite optical sensor design to ensure proper sampling of the point spread function (PSF) of the sensor, which acts as a band-limited spatial filter. A lens that meets this criterion is referred to as critically sampled (Gonzalez and Woods, 2007).

The PSF of the optical system is a function of light intensity based on the diffraction of light waves as they pass through an aperture that acts as a spatial filter on an image, and results in blurring of target intensity over more than one pixel (DeCusatis et al., 2010). A staring sensor has no ego-motion relative to the scene, therefore any localized intensity changes in the scene are due to motion of objects in the scene, or noise in the sensor. To track the target over successive images, referred to as frames, we model the target's motion and sensor's output for a given target state.

This paper is organized as follows: in section 2, we discuss previous work in the fields of subpixel target detection, and very dim object tracking. Section 3 describes target, sensor and motion models, as well as defines our method for tracking subpixel targets, the Pixel Matched Viterbi (PMV) algorithm. Section 4 discusses experiments performed to test PMV in both synthetically generated and real-world data. Finally, section 5 examines the strengths and weaknesses of our method as well as provides suggestions for future work.

2 PRIOR WORK

A subpixel target is one whose physical dimensions are smaller than the spatial resolution of the sensor. The total contribution of an object to the final intensity of the pixel is based on the PSF of the sensor, $h(\cdot)$, the intensity of the target, α, measured in photon flux, and the total intensity of all background objects within a pixel's field of view. To track the target, it is necessary to identify which pixel contains the target, and estimate where the target is located within that pixel.

Samson, et al. addresses the detection of subpixel targets in a fixed scene (Samson et al., 2004). It provides a method to detect subpixel point targets using the sensor's PSF as an identifying characteristic. Using matched filtering theory, a measure is defined to

Lotspeich J. and Kolsch M. (2013).
Tracking Subpixel Targets with Critically Sampled Optics.
In *Proceedings of the 2nd International Conference on Pattern Recognition Applications and Methods*, pages 375-381
DOI: 10.5220/0004263903750381

determine the likelihood that a given pixel in an image contains a subpixel object. This measure is used to perform a detection decision by applying a threshold to the likelihood values. The Neyman-Pearson criterion to is used to determine the optimal threshold value for a given false positive rate. The resulting detector is shown to be greater than 70% accurate for images with SNR=14dB.

The primary contribution of Samson, et al. is the Generalized Pixel-Matched Filter (GPMF). This filter calculates the maximum likelihood estimate of the unknown target photon flux, α, and returns the minimum sum of squared error estimate for the template centered at each pixel in an image. The GPMF is calculated as:

$$\ell(z^{(x,y)}) = \frac{\left|(s^{\varepsilon_0})^T R^{-1} k(z^{(x,y)})\right|^2}{(s^{\varepsilon_0})^T R^{-1} s^{\varepsilon_0}} \qquad (1)$$

The variable s^{ε_0} is a 9×1 vector representation of a 3×3 pixel template. This template is generated by calculating the ensquared energy of the PSF for each pixel assuming a target with photon flux α is located in the center of the template. The variable R represents the sensor's noise covariance matrix, and $k(\cdot)$ is a function that returns a 9×1 vector representation of the 3×3 pixel patch with center pixel of (x,y) in the observation z. Since the sensor noise is assumed independent and identically distributed (IID), R becomes $I^9 r$ where I^n denotes an identity matrix with diagonal of size n.

The GPMF provides the likelihood that a subpixel target exists in any given pixel. It does not provide an estimate of the subpixel location of that target. Since GPMF assumes a template with the target located in the center of a pixel, objects offset from the center will produce a lower overall likelihood. In order to calculate the likelihood of a pixel in any subpixel position, Samson, et al. propose an Approximate Likelihood Ratio Test (ALRT) that uses the trapezoidal rule to approximate the integral:

$$\mathcal{L}(x,y) = \int_{-0.5}^{0.5} \int_{-0.5}^{0.5} \frac{1}{(s^{(i,j)} R^{-1} s^{(i,j)})^{0.5}} \exp\left(\ell(x,y)\right) \mathrm{d}i \mathrm{d}j \qquad (2)$$

While the ALRT method provides a measure of pixel-wise likelihood, it does not provide an estimate of subpixel location. To determine subpixel location, a second Maximum Likelihood (ML) estimate must be performed for each pixel likelihood exceeding the specified threshold. If a large number of pixels fall above the threshold limit, this will result in a high computational load. Additionally the subpixel position estimate does not incorporate any prior location probabilities or data from multiple observations over time. In order to use this method for tracking, it is necessary to specify a tracking method that can use detections provided by the ALRT method.

While not specifically focused on subpixel tracking, Track Before Detect (TBD) filters attempt to track very low signal-to-noise ratio (SNR) targets through a sequence of images. Since the total photon flux generated by a subpixel target is very small relative to the overall sensor size, these targets generally exhibit a very low SNR. Additionally, TBD filters focus on very small targets down to, but not smaller than, a single pixel and typically incorporate the PSF in their detection methods(Ristic et al., 2004; Rutten et al., 2005a).

The TBD filter attempts to track a target using multiple detection probabilities over time rather than tracking detections above a given threshold. This allows the filter to integrate motion information over time with multiple detection likelihoods in order to build a better estimate of a target's location. Since the sensor model is non-linear, TBD methods use a particle filter algorithm to perform estimation (Rutten et al., 2005b). Theoretical lower bound performance of these filters indicate the ability to perform with subpixel accuracy, however, while many applications achieve subpixel estimation accuracy, most applications in the literature fail to achieve the lower bound performance (Morelande and Ristic, 2009; Ristic et al., 2004). Since TBD filters use particle filter methods, they tend to be relatively fast, with computation time proportional to the number of particles used. The number of particles, however, also relates directly to the ability of the filter to converge on the correct estimate. As the number of dimensions or the size of the domain for any dimension increases, the number of particles required increases (Arulampalam et al., 2001).

The drawback of using TBD methods is that the filter performance degrades rapidly as the SNR falls below 7dB. As demonstrated by Rutten, et al. (Rutten et al., 2005b), the TBD method produces root mean squared (RMS) position error of no less than 1 pixels at 6dB, and no less than 3 pixels at 3dB. In contrast, the Cramer-Rao lower bound (CRLB) predicts the theoretical performance of the estimator to be better than 10^{-1} pixels at 4dB (Morelande and Ristic, 2009). While TBD methods perform close to the the lower bound above 7dB, their performance suffers below this level.

The PMV method presented here uses the Maximum A Posteriori (MAP) formulation of the target's state through a Hidden Markov Model (HMM) as the basis of estimation. As opposed to ML estimation used in TBD methods, the MAP estimator calculates the maximum probability path through a HMM given all observations from time $1 \ldots t$. This allows full utilization of all data regarding the target rather than performing an estimate for each time step based only on the previous estimation and current observation. For a large number of observations, we expect the ML and MAP to converge on the final position estimate, but the ML solution will usually contain higher error in its earlier estimates since those were made using only the data up to that time step. A smoothed ML estimate may provide reduced error for earlier estimates relative to the ML solution, however, it is necessary to determine a smoothing distribution. While this is possible, as demonstrated by Junkun, et al. in (Junkun et al., 2011), the final results are still well above the expected CRLB.

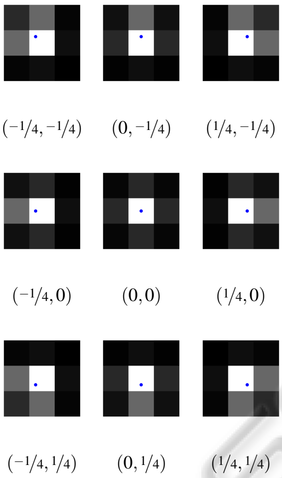
3 METHODOLOGY

The PMV algorithm estimates the MAP of the path of a subpixel object in three iterative steps:

1. Calculate the target's current state likelihood given the current observation
2. Calculate the probability of a state transition from the previous time step to the current for each possible target state
3. For each possible target state, select the maximum sum of the likelihood and state transition

3.1 Sensor Model

The sensor is assumed to consist of a discrete grid of square sensing elements, such as a Charge-coupled Device (CCD) or Complimentary Metal-oxide-semiconductor (CMOS) device. Furthermore we assume that the sensor is monochromatic, with each sensing element reporting the local photon flux detected during a fixed integration interval. Additionally, the sensor is assumed to be critically sampled. The width of the PSF is $1.22\lambda N$, where λ represents the average of the sensor's frequency range, and N is the ratio of aperture diameter to focal length. By the Rayleigh Criterion, the maximum spatial frequency that can be sensed is equal to the width of the PSF (DeCusatis et al., 2010). According to the Nyquist sampling theorem, the optimal sampling frequency for the PSF is $1/2.44\lambda N$. By setting the size of the sensor photo site to this optimal sampling value, the sensor is critically sampled. It is common practice to design satellite and optical sensors for remote sensing applications using this criteria (Olsen, 2007).


$(-1/4,-1/4)$ $(0,-1/4)$ $(1/4,-1/4)$

$(-1/4,0)$ $(0,0)$ $(1/4,0)$

$(-1/4,1/4)$ $(0,1/4)$ $(1/4,1/4)$

Figure 1: Templates generated by a subpixel target at $(-1/4,-1/4),\ldots,(1/4,1/4)$. The dot indicates the actual target position for each template.

An observation from the sensor consists of four elements: the effect of the sensor's PSF on the scene, the contribution of the target, the background, and sensor noise. These contributions are represented mathematically as

$$z = h \otimes (f(x) + B) + \omega \tag{3}$$

where h denotes the PSF of the optical system, $\otimes$ is the convolution operator, $f(x)$ is a function of target intensity over the scene given a target located at position x, B is a static function of the background scene intensity, and $\omega \sim \mathcal{N}(0,r)$ is IID for each pixel. Using the distributive property of convolution, eq 3 is rewritten

$$z = h \otimes f(x) + h \otimes B + \omega \tag{4}$$

Eq 4 separates the background contribution from the target contribution without the need to deconvolve the image first. This makes it possible to remove the background contribution from the scene using only estimates of the background determined from the observation frames. We use the median function to perform background subtraction, $\hat{B} = \text{median}(h \otimes B_{1\ldots t})$. As determined in (Lotspeich, 2012), this provides adequate subtraction for the PMV algorithm relative to other popular methods such as Mixture of Gaussians. The background subtracted observation is denoted

$$\bar{z} = z - \hat{B} \tag{5}$$

Using $\bar{z}$, we formulate a likelihood measure. Due to the PSF, the target signal will spread onto neighboring pixels. This additional information can be used to estimate the subpixel location of the target by comparing the expected intensity values of a pixel and its neighbors with the observed values. Figure 1 shows the ensquared energy of the PSF over a 3×3 pixel template for different subpixel positions. The additional information provided in neighboring pixels makes it possible to use an optimal template matched filter on the image to determine the matching error between a template and the 3×3 pixel patch centered at a given pixel. We use the GPMF developed by Samson, et al. as the matched filter. To determine the likelihood for each subpixel location, we generate a set of templates, S, with each template, $s^{\varepsilon} \in S$, representing a target located at a discrete subpixel point in the central pixel of the template. The central pixel is divided into $\rho \times \rho$ subpixel locations such that $\varepsilon \in \{(i,j) | i,j \in -{}^{1}/_{2}, -{}^{1}/_{2} + {}^{1}/_{\rho}, -{}^{1}/_{2} + {}^{2}/_{\rho}, \ldots, -{}^{1}/_{2} + {}^{\rho-1}/_{\rho}\}$. The likelihood density for each discrete subpixel location in the image is calculated for each pixel, $\bar{z}^{(x,y)}$, in the image.

$$\ell(\bar{z}^{(x,y)}|X) = \frac{\left|(s^{\varepsilon})^T R^{-1} k(\bar{z}^{(x,y)})\right|^2}{(s^{\varepsilon})^T R^{-1} s^{\varepsilon}} \tag{6}$$

3.2 Motion Model

The results presented in this paper assume a constant velocity target motion with additive Gaussian noise as described by Bar-Shalom in (Bar-Shalom et al., 2001). The Nearly Constant Velocity (NCV) model assumes that a target is moving at a fixed velocity with noisy, zero mean acceleration. The state for this model consists of position and velocity in the x and y dimensions. To avoid ambiguity between position and an instance of the random variable X, the state values are denoted using i and j instead.

$$X = \begin{bmatrix} i & \Delta i & j & \Delta j \end{bmatrix}^T \tag{7}$$

The state update equation is given as:

$$x_k = A x_{k-1} + \eta \tag{8}$$

With a linear transition matrix

$$A = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{9}$$

The noise, η, is additive Gaussian with covariance matrix Q.

$$\eta \sim \mathcal{N}(0, Q) \tag{10}$$

$$Q = \begin{bmatrix} \frac{q}{3}\Delta k^3 & \frac{q}{2}\Delta k^2 & 0 & 0 \\ \frac{q}{2}\Delta k^2 & q\Delta k & 0 & 0 \\ 0 & 0 & \frac{q}{3}\Delta k^3 & \frac{q}{2}\Delta k^2 \\ 0 & 0 & \frac{q}{2}\Delta k^2 & q\Delta k \end{bmatrix} \tag{11}$$

Q represents the covariance matrix of the NCV noise. The variable $q = 0.01$ is used in all tests described in section 4. The variable Δk represents the time increment between frames and is set to 1 for all experiments.

Since η is a Gaussian, the motion probability for this model is:

$$p(X_t|X_{t-1}) = (2\pi)^{-2}|Q|^{-\frac{1}{2}} \exp^{-\frac{1}{2}(X_k - AX_{k-1})^T Q^{-1}(X_k - AX_{k-1})} \tag{12}$$

3.3 Calculating MAP

Using sensor model likelihood in eq 6 and motion probability in eq 12, the MAP is calculated as

$$\mathrm{MAP}(X_{1\ldots t}) = p(X_0) \prod_{k=1}^{t} \operatorname*{argmax}_{X_k} \ell(\bar{z}_k|X_k) p(X_k|X_{k-1}) \tag{13}$$

The naive implementation of this calculation requires $O(n^t)$ time where $n = \rho^2 wh$, when each frame has size $w \times h$. This implementation quickly becomes computationally intensive for more than a few frames. Since the number of possible states has been discretized, the Viterbi algorithm can be used to optimally calculate the MAP in $O(tn^2)$ time (Viterbi, 1967).

The Viterbi algorithm is a dynamic programming method that calculates the optimal path through an acyclic directional graph given node and edge costs. The HMM representing each possible discrete target state can be viewed as an acyclic directional graph where each state at time t connects to each state at time $t+1$. The node value for each state is calculated using eq 13, and edge weights from t to $t+1$ are calculated using eq 12. The Viterbi algorithm calculates the MAP of the states by selecting the maximum inbound edge for each successive set of states. The states' weights are then updated by multiplying the weight of the inbound edge with the current state likelihood. Outbound edges are calculated as the product of the motion model probability and the node weight. The maximum probability node in the last time sequence represents the final node of the MAP sequence. A simple backtracking step follows the maximum inbound edge for each node, providing the final sequence of states in the MAP.

3.3.1 Distance Transform

Assuming the noise in eq 12 is additive Gaussian noise, we can use the distance transform (DT) algorithm developed by Felzenswalb and Huttenlocher (Felzenszwalb and Huttenlocher, 2004) to further reduce the computation time from $O(tn^2)$ to $O(tn)$. The DT algorithm calculates the surface described by the maximum value of a set of functions $S = \{f_1(x), f_2(x), \ldots, f_n(x)\}$ over each value of x.

The distance transform algorithm calculates the maximum or minimum manifold of a set of parabolic or conic functions over a given domain. The set of functions consists of tuples (h,k), where each tuple represents a parabola in the form

$$f(x) = a(x-h)^2 + k \tag{14}$$

The variable a determines the size and direction of the parabola. To apply the distance transform algorithm, the value of a must be constant over all parabolas in the set.

The distance transform algorithm only works in one dimension, however, since Q is separable (i.e. it can be expressed as the outer product of two vectors, v and h), it is possible to run the distance transform once for each dimension of the state - in this case the i and j axis of the target position. Since the probability of a target state can be expressed as $f(x) = \log(p(X_k|X_{k-1})) + \log(\ell(\bar{z}_k|X_k))$, $f(x)$ represents the Gaussian distribution of the target's state at time k given that $X_k = (i,j)$. Since $f(x)$ is Gaussian, it is separable. Substituting $X_k = (i,j)$ into the motion model in eq 12 and the likelihood in eq 6, separating the Gaussian distribution into the marginal distributions for i and j locations and simplifying yields:

$$f(i) = \frac{-1}{2\sigma_1^2}(i-\mu_1)^2 + \log(p(\hat{X})) + \log\left(\frac{1}{\sqrt{2\pi\sigma_1^2}}\right) \tag{15}$$

$$f(j) = \frac{-1}{2\sigma_3^2}(j-\mu_3)^2 + \log(p(\hat{X})) + \log\left(\frac{1}{\sqrt{2\pi\sigma_3^2}}\right) \tag{16}$$

with $\begin{bmatrix} \mu_1 & \mu_2 & \mu_3 & \mu_4 \end{bmatrix}^T = Ax_{k-1}$, $\sigma_d = Q_{dd}$, for each dimension $d \in \{1,3\}$. The DT algorithm is performed for the set of functions over the i and j dimensions. The resulting surface is the MAP estimate of target states for the time step k.

4 RESULTS

To test the performance of the proposed method, we estimated the target path over a series of simulated image sequences at various SNR values from 1dB to 20dB. The SNR is calculated as

$$\text{SNR} = 20\log\left(\frac{\alpha}{\sigma}\right) \tag{17}$$

The estimated paths were then compared with the known target path to determine the RMS distance error in pixels. The TBD filter described by Ristic, et al. was run on the same data set with 10000 particles, with p(birth)= 0.05, p(death)= 0.05, and $\Delta x = \Delta y = 1$. The PSF width, Σ, was set to $1.22/2\sqrt{2\ln 2}$, which matches the PSF width of a critically sampled system with $\lambda N = 1$. The detection threshold for TBD is set to 0.7. This means that the target is considered detected if greater than 70% of the particles are in the "continuation" chain of the jump Markov model.

The TBD and PMV filters were each run 20 times on synthetic imagery from 1dB to 20dB, and the RMS distance error was calculated for each estimated path. In the case of the TBD filter, distance error is only calculated for those points of the path that exceed the detection threshold. To measure the computational time for each filter, images of size 30×30, 85×85, and 200×200 where tested. Figure 2 shows the average RMS over 20 runs for each SNR value, and table 1 shows the running time for both algorithms on the different data sets.

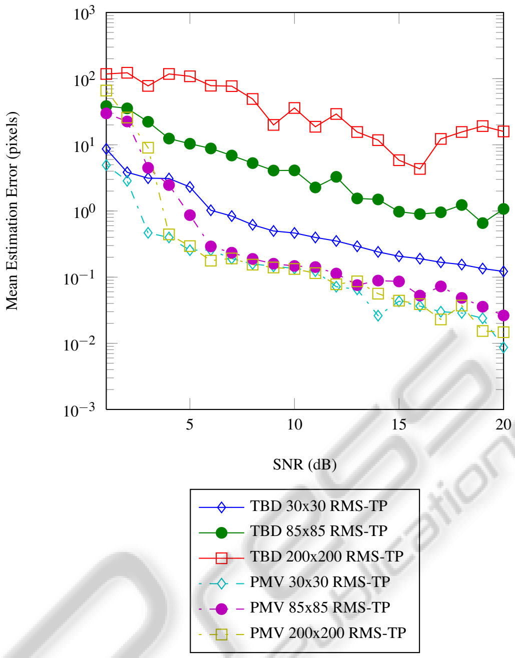


Figure 2: Average RMS over 20 runs for SNR of 1-20dB for TBD and PMV.

Table 1: Run times for PMV and TBD for synthetic data sets. Mean run times are reported over 20 separate tests.

	Image Size	Mean Run Time (s)	Std of Run Time (s)
PMV	30x30	17.9897	2.4852
	85x85	58.3830	1.7788
	200x200	403.7682	8.3409
TBD	30x30	1.2973	0.0488
	85x85	1.9445	0.0711
	200x200	6.3925	0.6091

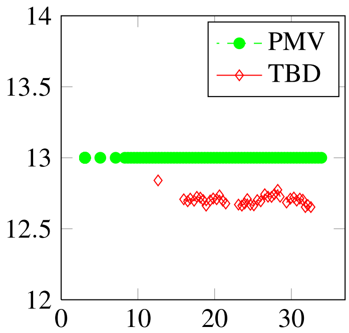


Figure 3: Close-up of the PMV estimated path and TBD estimated path in the real-world data set.

Using the mean RMS, PMV outperforms TBD in every data set for SNRs greater than or equal to 4dB. At SNR values from 1 to 4dB, the methods are relatively comparable, however, both methods fail to achieve subpixel estimation error. Defining detection rate as the number of estimated points that fall within a 1 pixel distance of the ground truth, PMV achieves greater than 80% detection rate for SNR of 3dB and greater, with TBD producing greater than 80% at 7dB for 30x30 images, and 12dB for 85x85. TBD fails to achieve greater than 80% for the 200x200 data set at any tested SNR. In relation to TBD, which is dependent on image size, PMV does not appear to reduce quality with image size. However, as expected, the linear growth of PMV with respect to the number of pixels results in much larger computation time.

To validate PMVs performance against real-world data, we captured video of a 14mm diameter LED emitting at the 3μm light frequency, mounted on a robot platform moving at a constant 0.4m/s speed from a distance of 97m using a FLIR 8200 mid-wave infrared (MWIR) camera with 50mm lens and f/4 aperture setting. The camera's PSF is calculated to have a full width at half-maximum (FWHM) of 16.448μm on a sensor with pixel pitch of 18μm. The spectral range of the camera is $3-5\mu m$ wavelength frequencies. This results in a pixel sampling rate of $r_c = 1.36$, and a Nyquist sampling distance of $^1/_{1.36\cdot 4\mu m\cdot 4}$. This means the camera is under sampled for the PSF size, however, since the full width of the spot is greater than the pixel pitch, the target intensity is still spread to more than one pixel. To avoid adding multiple targets to the image, the final sequence of images was cropped to remove the robot from the test imagery. The real-world image sequence consists of 126 frames of 59×14 sized images.

Figure 3 shows the estimated paths for PMV and TBD. The breaks in the TBD-estimated path are due to the filter falling below the 0.7 threshold value during portions of the sequence. Due to the difficulty in determining the transformations necessary to convert real-world target coordinates to camera coordinates, ground truth target location is not known. Instead, the resulting paths are overlaid on the actual image sequence as seen in figure 4.

The smoothness of the PMV estimate is due to the use of the entire path in estimation. Low probability estimates at the beginning of the sequence may still be included in the final solution. For TBD, on the other hand, there is no mechanism for keeping track of prior path estimates when the filter has not converged. The estimate tends to have high variance at first, with the variance diminishing as further observations are made. As a consequence of this behavior, estimates made prior to convergence will result in larger error. Additionally, the TBD population of particles have a minimum variance after convergence that results in a shifting mean. While the mean stays close to the actual target location, the fluctuation produces poorer results than the PMV estimate.

While the low noise characteristics of the sensor used to the capture the real-world data set preclude testing at lower SNR values, the observed results are consistent with the predicted results from the synthetic data set. At a peak SNR of ˜20dB, both methods are expected to produce subpixel position estimates with PMV providing a lower RMS error. As seen in figure 3, the paths produced by each method are very close to each other, with the smoother path of the PMV more closely matching the actual motion of the target.

5 CONCLUSIONS

While the results show that PMV performs better than the TBD method, they also indicate areas for further research. For example, the computational requirements of PMV precludes its use in real-time imagery. A number of techniques may reduce overall processing time without reducing estimation quality. For example, adaptively adjusting the value of ρ based on likelihood values has the potential to reduce the total number of states considered by the Viterbi algorithm. Also, alternative Viterbi solution methods such as Lazy (Feldman et al., 2002), A-Star (Klein and Manning, 2002), or Tree Viterbi (Nelson and Roufarshbaf, 2009) have all been shown to increase the speed of the Viterbi algorithm by evaluating fewer states than the proposed method. While initial experiments show the distance transform providing superior results, this conclusion has not been thoroughly tested.

Another possible improvement is the development of a likelihood function for color optical sensors. The

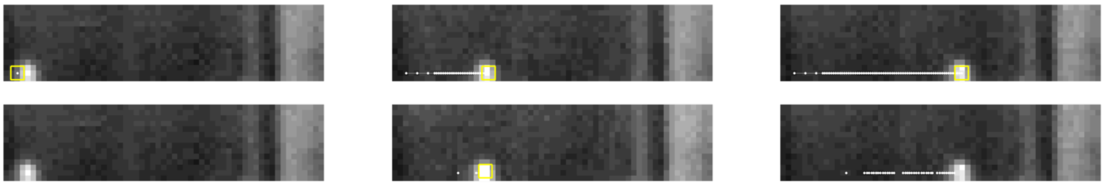


Figure 4: Three frames for the real-world dataset. The top row shows the PMV estimated path up to the current frame for frames 47, 87, and 126 with a rectangle placed at the current estimated location. The bottom row shows the TBD estimated path up to the current frame for frames 47, 87, and 126. A rectangle is placed over the current estimate if TBD has passed the detection threshold for the frame (frames 47 and 126 do not have rectangles because TBD has fewer than 70% of particles tracking the target. Picture intensity in these images is adjusted to highlight the target.

use of a color filter over different pixels of an optical sensor, or a prism in the case of 3 CCD cameras results in different PSFs for each pixel of the image. While this complicates the likelihood function, it may also provide another indicator of target location. Additionally, since the width of the PSF is dependent on the wavelength of light, different color filters will provide more or less information in neighboring pixels. By matching the likelihood function to the sensor in use, it may be possible to increase accuracy of PMV.

This paper has demonstrated a method for tracking the path of subpixel objects in image sequences captured by a critically sampled optical sensor. A likelihood method is developed using a pixel matched optimal filter. The problem is formulated using the MAP solution for HMMs, and the motion model is mapped to a distance transform problem that reduces the overall complexity from $O(tn^2)$ to $O(tn)$. We compared the performance of PMV to a current state of the art method and show that our method outperforms TBD in all data sets used. Finally, we provide real-world validation of the results observed for the synthetic data set. To the best of our knowledge, PMV is the first subpixel target tracking method proposed in the literature.

REFERENCES

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

Bar-Shalom, Y., Li, X. R., and Kirubarajan, T. (2001). *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons, Inc., New York, NY, USA.

DeCusatis, C., Enoch, J., Lakshminarayana, V., Li, G., MacDonald, C., Mahajan, V., and Stryland, E. V. (2010). *Handbook of Optics*, volume 4. McGraw Hill, 3 edition.

Feldman, J., Abou-Faycal, I., and Frigo, M. (2002). A fast maximum-likelihood decoder for convolutional codes. In *Vehicular Technology Conference, 2002. Proceedings. VTC 2002-Fall. 2002 IEEE 56th*, volume 1, pages 371 – 375 vol.1.

Felzenszwalb, P. F. and Huttenlocher, D. P. (2004). Distance transforms of sampled functions. Technical report, Cornell Computing and Information Science.

Gonzalez, R. C. and Woods, R. E. (2007). *Digital Image Processing (3rd Edition)*. Prentice Hall, 3 edition.

Junkun, Y., Hongwei, L., Xu, W., and Zheng, B. (2011). A track-before-detect algorithm based on particle smoothing. In *Radar (Radar), 2011 IEEE CIE International Conference on*, volume 1, pages 422 –425.

Klein, D. and Manning, C. D. (2002). A* parsing: Fast exact Viterbi parse selection. Technical Report 2002-16, Stanford InfoLab.

Lotspeich, J. (2012). *Tracking Subpixel Targets With Critically Sampled Optical Sensors*. PhD thesis, Naval Postgradute School, Monterey, CA.

Morelande, M. and Ristic, B. (2009). Signal-to-noise ratio threshold effect in track before detect. *Radar, Sonar Navigation, IET*, 3(6):601 –608.

Nelson, J. and Roufarshbaf, H. (2009). A tree search approach to target tracking in clutter. In *Information Fusion, 2009. FUSION '09. 12th International Conference on*, pages 834 –841.

Olsen, R. (2007). *Remote Sensing from Air and Space*. The International Society for Optical Engineering, Monterey, CA.

Ristic, B., Arumluampalam, S., and Gordon, N. (2004). *Beyond the Kalman Filter-Particle Filters for Tracking Applications*. Artech House, Boston, MA.

Rutten, M., Gordon, N., and Maskell, S. (2005a). Recursive track-before-detect with target amplitude fluctuations. *Radar, Sonar and Navigation, IEE Proceedings -*, 152(5):345 – 352.

Rutten, M., Ristic, B., and Gordon, N. (2005b). A comparison of particle filters for recursive track-before-detect. In *Information Fusion, 2005 8th International Conference on*, volume 1, page 7 pp.

Samson, V., Champagnat, F., and Giovannelli, J.-F. (2004). Point target detection and subpixel position estimation in optical imagery. *Applied Optics*, 43(2):257–263.

Viterbi, A. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *Information Theory, IEEE Transactions on*, 13(2):260 –269.