Title

Tracking highly maneuverable targets with unknown behavior

Journal

Proceedings of the Ieee, 92(3)

ISSN

0018-9219

Authors

Schell, C
Linder, S P
Zeidler, J R

Publication Date

2004-03-01

Peer reviewed

Tracking Highly Maneuverable Targets With Unknown Behavior

CHAD SCHELL, MEMBER, IEEE, STEPHEN P. LINDER, MEMBER, IEEE, AND JAMES R. ZEIDLER, FELLOW, IEEE

Invited Paper

Tracking of highly maneuvering targets with unknown behavior is a difficult problem in sequential state estimation. The performance of predictive-model-based Bayesian state estimators deteriorates quickly when their models are no longer accurate or their process noise is large. A data-driven approach to tracking, the segmenting track identifier (STI), is presented as an algorithm that operates well in environments where the measurement system is well understood but target motion is either or both highly unpredictable or poorly characterized. The STI achieves improved state estimates by the least-squares fitting of a motion model to a segment of data that has been partitioned from the total track such that it represents a single maneuver. Real-world STI tracking performance is demonstrated using sonar data collected from free-swimming fish, where the STI is shown to be effective at tracking highly maneuvering targets while relatively insensitive to its tuning parameters. Additionally, an extension of the STI to allow its use in the most common multiple target and cluttered environment data association frameworks is presented, and an STI-based joint probabilistic data association filter (STIJPDAF) is derived as a specific example. The STIJPDAF is shown by simulation to be effective at tracking a single fish in clutter and through empirical results from video data to be effective at simultaneously tracking multiple free-swimming fish.

Keywords*—Probabilistic data association, state estimation, tracking.*

I. INTRODUCTION

A common application of sequential state estimation is the tracking of targets as they move through a sensor's field of view. Many of the filters commonly used for target tracking, such as the Kalman filter and its relatives, are predictive model-based Bayesian state estimators. These filters achieve improved state estimates through the use of a predictive model that describes the evolution of the target state through time. The model consists of a state transition function that describes the evolution of the state in the absence of (unknown) external inputs, and a process noise that represents unknown changes to the state not described in the state transition function.

However, state evolution is often not easily modeled in a predictive fashion. This can happen either when the system being studied is not well understood, or when the random changes in the state are large enough to dominate the predictable changes. In these situations predictive filters can become ineffective and an alternative method is required, one that does not rely on prediction. An alternative is to use parameter estimation or curve fitting techniques to estimate the target state directly from the data. In order to use this approach with a relatively simple state vector and fitting function, the data must be broken into segments over which the simple fitting function can adequately describe the data. Thus, the problem now consists of two parts, segmenting the data and then estimating the state, or the parameters, of the individual segments.

This paper demonstrates the performance advantages of one such data-driven approach to tracking highly maneuverable targets with unknown behavior: the segmenting track identifier (STI), first introduced by Linder [1]. The STI's advantages are demonstrated by comparison against Kalman and extended Kalman filters (EKFs) for the tracking of free-swimming fish. Additionally, an extension of the STI algorithm is presented, which enables its use in the common data association frameworks for multiple target tracking, and an STI-based joint probabilistic data association filter (STIJPDAF) is developed as a specific example. The effectiveness of the STIJPDAF is demonstrated using simulations and empirical results from fish-tracking experiments.

Manuscript received February 13, 2003; revised October 28, 2003. This material is based in part upon work supported under a National Science Foundation Graduate Fellowship.

C. Schell was with the San Diego Electrical and Computer Engineering Department and the Scripps Institution of Oceanography, University of California, La Jolla, CA 92093 USA. He is now with Rincon Research Corporation, Tucson, AZ 85711 USA (e-mail: chad@schells.com).

S. P. Linder is with the Department of Computer Science, Dartmouth College, Hanover, NH 03755-3510 USA (e-mail: spl@cs.dartmouth.edu).

J. R. Zeidler is with the San Diego Electrical and Computer Engineering Department, University of California, La Jolla, CA 92093 USA (e-mail: zeidler@ece.ucsd.edu).

Digital Object Identifier 10.1109/JPROC.2003.823151

II. Segmenting Track Identifier

The STI is a data-driven tracking algorithm that achieves improved state estimates by partitioning the track data into segments which contain only a single maneuver and then performing least-squares fitting of the motion model to each track segment. It is similar in some ways to algorithms designed to recognize curves and lines in images and freehand drawings [2]–[4]. However, these image processing methods generally rely on a high sampling density and relatively low measurement noise as their purpose is to reconstruct or recognize elements of images which already look approximately like lines or arcs, while the STI algorithm is designed to operate not only in low-noise, high data rate situations, but also in low-noise, low data rate and high-noise, high data rate situations.

As the STI is a data-driven rather than a predictive algorithm, no process noise is required to handle target maneuvers. Maneuvers are represented by a segmentation of the data, and segmentation decisions are based only on knowledge of the system measurement errors, a quantity that is very likely known regardless of the type of target being tracked. Additionally, segmentation also allows rapid response to large abrupt maneuvers as it makes a clean break from the previous segment, which is helpful in low data rate environments where predictive filters may be slow to respond to such maneuvers. However, one drawback to the STI's lack of prediction is that STI in its simplest form cannot be easily used in the most common data association frameworks for tracking in clutter or for tracking multiple targets.

This section presents the original single-target STI algorithm, and as a specific example details the development of a motion model for a target performing constant speed coordinated turns. After presenting the original single-target version of the STI, an extension to the STI algorithm that computes a measurement prediction and its covariance is presented. This extension allows the STI to now be used in many of the common data association frameworks including assignment methods and multiple hypothesis tracking. Finally, as an example, the STIJPDAF is developed in detail. A probabilistic data association (PDA) algorithm was chosen as the example because using the STI in a PDA algorithm requires some additional consideration to support the two-way communication required between the PDA algorithm and the STI.

A. STI Algorithm

The STI algorithm dynamically partitions a target track into M segments, S_1: S_M, where M is an index that starts at one and grows as the algorithm determines new segments are needed. The sequential target state is estimated by recursively calculating the segment parameter vectors, $\mathbf{x}_1$: $\mathbf{x}_M$, that minimize the pairwise sums of the segment least-squares cost functions χ_1: χ_M. A recursive, pairwise minimization is used in place of a global minimization to keep the problem computationally tractable.

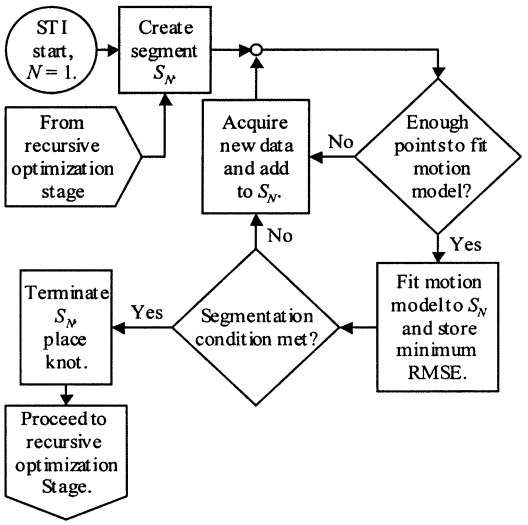

Fig. 1. STI fit and segmentation stage flowchart.

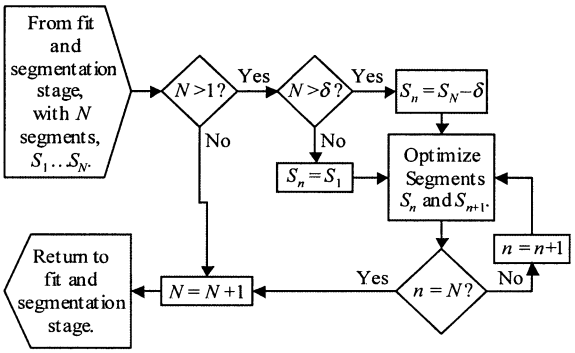

Fig. 2. STI recursive optimization stage flowchart.

Segments are the base element on which the STI algorithm operates. Each segment represents a single maneuver within a track, or more specifically a period in which the sequential evolution of the target state can be described completely by a state transition function $\mathbf{f}_{\mathrm{m}}$ and an initial state (at the start of the segment) $\mathbf{x}_{\mathrm{m}}$. A track's measurements are subdivided into overlapping subsets, such that each pair of consecutive segments share a measurement as a common end point, referred to as a knot.

With the exception of a brief initialization procedure, the STI is a recursive two-stage algorithm. The first stage performs the initial fit and segmentation of the measurements as each new measurement becomes available. The second stage recursively optimizes the fit and segmentation of the previously acquired measurements. The second stage is a smoothing algorithm that uses future measurements to perform the optimization of past fits. The two stages are illustrated in flow charts in Figs. 1 and 2.

The STI algorithm is initialized by assigning the first measurement to the start of the first segment, S_1, and setting the total number of segments $M = 1$. Associated with each segment S_m are the following: Y_m, the set of measurements assigned to the segment; L_m, a count of the number of assigned measurements; $\mathbf{x}_m$, a parameter vector that describes the parameters at the start of the segment; and $\psi_{m,\min}$, a record of

the lowest root-mean-squared fitting error achieved for the segment. Note that the values of $\mathbf{x}_m$ and $\psi_{m,\min}$ are undefined until the segment has actually been fit to the motion model, which occurs in the fit and segmentation stage.

1) Fit and Segmentation Stage: Each new measurement is added to the current segment S_M. If, after adding the measurement, $L_M < L_{\min}$, where $L_{\min}$ is the minimum number of points required to perform a fit to the motion model, the algorithm returns to the start of this stage. Otherwise, the parameter vector $\mathbf{x}_M$ is estimated to minimize the least-squares cost function

$$\chi_M = \|\mathbf{c}(\mathbf{x}_M, Y_M, S_{M-1})\|_2^2 \tag{1}$$

where $\mathbf{c}(\mathbf{x}_M, Y_M, S_{M-1})$ is the vector valued function that calculates both the continuity knot costs between S_M and S_{M-1} and the measurement residuals for Y_M for a given $\mathbf{x}_M$. The continuity knot costs maintain continuity of motion between two segments. For segment S_1, the knot costs are identically zero as there is no previous segment S_0. The root-mean-squared error (RMSE), $\psi_M = \sqrt{\chi_M/\gamma_{\mathbf{c}}}$ is also calculated, where γ_c is the length of the vector $\mathbf{c}$, which is equal to the dimension of the measurement vector times L_M, plus the dimension of the knot costs. Each time the segment is fit, $\psi_{M,\min}$ is updated as

$$\psi_{M,\min} = \begin{cases} \psi_M, & L_M = L_{\min} \\ \min(\psi_M, \psi_{M,\min}), & \text{otherwise} \end{cases}. \tag{2}$$

This add-and-fit procedure is repeated for each new measurement until a segmentation condition is met. A segmentation condition occurs when any of the following are true:

$$\begin{aligned} \psi_M &> \kappa_\psi \psi_{M,\min} \\ \psi_M &> \kappa_\sigma \sigma_v \\ \psi_{M,\tau} &> \kappa_\sigma \sigma_v \end{aligned} \tag{3}$$

where σ_v is the measurement noise standard deviation, and κ_ψ and κ_σ are tuning parameters that determine the thresholds used for segmentation. The value $\psi_{M,\tau}$ represents the root-mean-squared measurement residual for the last τ measurements, and is helpful in detecting break conditions in long segments, where the poor fit from new measurements after a maneuver is obscured in ψ_M by the averaging across the entire length of the segment. The value τ is also a tuning parameter of the algorithm, but it is restricted to integers between one and $L_{\min}$; otherwise, it would not exist for all fit segments as the segments could be shorter than the tail.

Once a segmentation condition has occurred, the segment S_M is terminated and the most recently added measurement is removed from the segment. A new segment S_{M+1} is started, and the last measurement of S_M and the measurement that caused the segmentation condition are assigned as the first two measurements of S_{M+1}. The shared measurement is the location of the knot between segments S_M and S_{M+1}. The algorithm then proceeds to the recursive optimization stage.

2) Recursive Optimization Stage: This stage recursively optimizes the fit and segmentation of past segments, performing the optimization on a pair of segments at one time. The optimization requires that the number of fitted segments be $M \geq 2$ and begin with segment S_B, $B = \max[1, M-u]$, where u is the optimization depth, the maximum number of previous segments to reoptimize. If $M < 2$, no optimization is performed, and the algorithm returns to the start of the fit and segmentation stage with $M = M + 1$.

The optimization algorithm works as follows. Starting with segment $S_m = S_B$, form $Y_{m,m+1}$, the union of the measurements from Y_m and Y_{m+1}. The set $Y_{m,m+1}$ will contain $L_{m,m+1} = L_m + L_{m+1} - 1$ measurements, as the knot will only appear once in the union. Define the notation $Y_{m,m+1}[a, b]$ as the subset of $Y_{m,m+1}$ containing the ath through bth measurements when listed in ascending time order of arrival. The goal of the algorithm is to replace S_m and S_{m+1} by the optimal segmentation, S_m^{opt} and S_{m+1}^{opt}, of the combined data set $Y_{m,m+1}$ defined as the knot location L_{opt}, and the optimized segment parameter vectors, $\mathbf{x}_m^{\text{opt}}$ and $\mathbf{x}_{m+1}^{\text{opt}}$, such that the cost

$$\begin{aligned} \chi_{\text{opt}} = &\left\|\mathbf{c}\left(x_m^{\text{opt}}, Y_{m,m+1}[1, L_{\text{opt}}], S_{m-1}\right)\right\|_2^2 \\ &+ \left\|\mathbf{c}\left(x_{m+1}^{\text{opt}}, Y_{m,m+1}[L_{\text{opt}}, L_{m,m+1}], S_m^{\text{opt}}\right)\right\|_2^2 \end{aligned} \tag{4}$$

is minimized subject to the constraint that $L_{\min} \leq L_{\text{opt}} \leq L_{m,m+1} - L_{\min} + 1$, where $\mathbf{c}$ is the same as in (1). After the optimal segmentation has been found, S_m and S_{m+1} are replaced by their optimal counter parts, and the algorithm proceeds to optimize the next pair of segments, starting with $S_m = S_{m+1}$ (one of the segments just replaced). This loop continues up to $S_m = S_{M-1}$. After the optimization loop is completed, the algorithm returns to the start of the fit and segmentation stage with $M = M + 1$.

Because the STI optimization procedure operates only on pairs of segments, it is expected that large optimization depths will produce diminishing returns because the location of the knots between pairs of segments will tend to settle to constant values for segments that have been reoptimized multiple times. Experience with the STI optimization algorithm has shown that there is practically no change for values of $u > 4$, so $u = 4$ is suggested as a constant and was used in this study.

3) STI Motion Models and Knot Costs: This section presents the format for STI motion models. They consist of a measurement generating function, $\mathbf{g}(\mathbf{x}_m, n_m)$, and a continuity knot cost function $\mathbf{k}(\mathbf{x}_m, S_{m-1})$. These functions are used together to form the STI cost function

$$\mathbf{c}(x_M, Y_M, S_{M-1}) = \begin{bmatrix} \mathbf{k}(\mathbf{x}_M, S_{M-1}) \\ \mathbf{y}_M(1) - \mathbf{g}(\mathbf{x}_M, n_M(1)) \\ \cdots \\ \mathbf{y}_M(L_M) - \mathbf{g}(\mathbf{x}_M, n_M(L_M)) \end{bmatrix} \tag{5}$$

where $\mathbf{y}_M(a)$ is the ath measurement of the set Y_M, and $n_M(a)$ is the elapsed time of the same measurement relative to the first measurement in Y_M. [For example, $y_M(1) = 0$.] As a specific example, the constant speed coordinated turn model used in a study of fish behavior [5] is presented.

The minimum number of measurements required to compute a fit for this model is $L_{\min} = 3$, and the parameter vector $x_M = [\varepsilon_0\ \eta_0\ \theta_0\ s_h\ \omega]^T$ estimated for each segment consists of the target's position (ε_0, η_0) and course (θ_0) at the start of the segment, and the target's speed (s_h) and turn rate (ω), both constant throughout the segment. The measurement generating function $\mathbf{g}(\mathbf{x}, n)$ is given by

$$\mathbf{g}(\mathbf{x}, n) = \begin{bmatrix} \varepsilon(n) \\ \eta(n) \end{bmatrix} = \begin{cases} \begin{bmatrix} \varepsilon_0 + \frac{2s_h}{\omega} \sin\left(\frac{\omega n}{2}\right) \cos\left(\theta_0 + \frac{\omega n}{2}\right) \\ \eta_0 + \frac{2s_h}{\omega} \sin\left(\frac{\omega n}{2}\right) \sin\left(\theta_0 + \frac{\omega n}{2}\right) \end{bmatrix}, & \omega \neq 0 \\ \begin{bmatrix} \varepsilon_0 + s_h n \cos(\theta_0) \\ \eta_0 + s_h n \sin(\theta_0) \end{bmatrix}, & \omega = 0 \end{cases} \tag{6}$$

which in the case of $\omega \neq 0$ is the time parameterized equation of an constant speed, constant turn rate arc, and in the case of $\omega = 0$ is the time parameterized equation of a constant speed straight line segment. The continuity knot cost function $\mathbf{k}(\mathbf{x}_M, S_{M-1})$ is

$$\begin{aligned} &\mathbf{k}(\mathbf{x}_M, S_{M-1}) \\ &\quad = \begin{cases} \kappa_C (L_M + L_{M-1})[\Delta_{\varepsilon,\eta} \quad \Delta_\theta]^T, & M > 1 \\ [0], & M = 1 \end{cases} \\ &\Delta_{\varepsilon,\eta} = \sqrt{\begin{array}{l} \{[\mathbf{g}(\mathbf{x}_M, 0) - \mathbf{g}(\mathbf{x}_{M-1}, n_{M-1}(L_{M-1})]^T \\ \times [\mathbf{g}(\mathbf{x}_M, 0) - \mathbf{g}(\mathbf{x}_{M-1}, n_{M-1}(L_{M-1})]\} \end{array}} \\ &\Delta_\theta = \theta_M - (\theta_{M-1} + n_{M-1}(L_{M-1})\omega_{M-1}) \end{aligned} \tag{7}$$

where κ_C is the knot cost multiplier, a tuning factor that affects how important continuity in position and heading at the knots is relative to the fit between the motion model and the measurements. The factor $(L_M + L_{M-1})$ insures that the proportional weight of the knot cost remains relatively constant even as the total length of the two segments increases. $\Delta_{\varepsilon,\eta}$ is the distance between the positions at the start of S_M and the end of S_{M-1}, and Δ_θ is the difference in course, measured in radians and ranging from zero (same course) to π (directly opposite course) radians.

B. STI Data Association

The required modifications to the single-target STI tracker for its use in the most common data association frameworks are presented in this section. The first modification, the calculation of the measurement prediction for the STI is trivial, as the measurement generating function, $\mathbf{g}(\mathbf{x}_M, N_M)$, of the STI model already serves this purpose. All that is required is to extend the time of the final measurement in the segment $n_M\ (L_M)$, which is measured relative to the time of the start of the segment, to the time of the desired measurement prediction, N. Then the predicted measurement is given by

$$\hat{\mathbf{y}}(N) = \mathbf{g}(\mathbf{x}_M, N). \tag{8}$$

The second modification, calculation of the measurement prediction covariance, is more difficult, but with the assumption that the STI is an unbiased estimator one can generate a covariance for the STI state parameters for a given segment using the Cramer–Rao lower bound (CRLB). The measurement prediction covariance is then generated using the CRLB and a process noise that characterizes the possible maneuvers of the target.

The CRLB represents the lowest possible covariance that can be obtained using any filter for a given set of measurements. Although it is used here to generate the state covariance for a segment, it is not meant to imply that the STI algorithm actually obtains this lower bound. It is simply used as a means of generating a state covariance that can be compared relative to the state covariance of segments from other tracks.

The process noise for the STI algorithm is added to the measurement prediction covariance rather than the state covariance, because the process noise represents the uncertainty in a target's predicted location that comes from target behavior between the last measurement and the prediction time. Without the process noise, the covariance estimate would approach zero with increasing information (increasing segment length), resulting in a zero-volume search location for the next measurement. The use of the process noise to model small unpredictable motions is not important to the underlying STI algorithm as it does not use the state covariance in fitting and segmentation.

Using the definition of the CRLB, the minimum state covariance for the latest segment, S_M, with estimated parameter vector $\mathbf{x}_M$, representing the state at the start of the segment is

$$\mathbf{K}_M = E\{[\mathbf{x}_M - \mathbf{x}_{M0}][\mathbf{x}_M - \mathbf{x}_{M0}]^T\} \geq \mathbf{J}^{-1} \tag{9}$$

where $\mathbf{x}_{M0}$ is the true target parameter vector (unknown for real data sets). $\mathbf{J}$ is the Fisher information matrix (FIM) given by

$$\begin{aligned} \mathbf{J} &= E\{[\nabla_{\mathbf{x}} \ln \Lambda(\mathbf{x}_M, Y_M)][\nabla_{\mathbf{x}} \ln \Lambda(\mathbf{x}_M, Y_M)]^T\}\,|_{\mathbf{x}_M = \mathbf{x}_{M0}} \\ &= E\{[\nabla_{\mathbf{x}} \lambda(\mathbf{x}_M, Y_M)][\nabla_{\mathbf{x}} \lambda(\mathbf{x}_M, Y_M)]^T\}|_{\mathbf{x}_M = \mathbf{x}_{M0}} \end{aligned} \tag{10}$$

with the expectation carried out over Y_M, the set of measurements associated with the segment. $\Lambda\ (\mathbf{x}_M, Y_M)$ is the likelihood function of $\mathbf{x}_M$, defined as

$$\begin{aligned} \Lambda(\mathbf{x}_M, Y_M) &= p[Y_M | \mathbf{x}_M] \\ &= p[\mathbf{y}(1), \mathbf{y}(2), \ldots, \mathbf{y}(L_M) \,|\, \mathbf{x}_M] \\ &= \prod_{i=1}^{L_M} p[\mathbf{y}(i) \,|\, \mathbf{x}_M] \end{aligned} \tag{11}$$

where $p[a \,|\, b]$ is the probability of a given b. $\lambda\ (\mathbf{x}_M, Y_M) = -ln\Lambda\ (\mathbf{x}_M, Y_M)$ is the log likelihood of $\mathbf{x}_M$.

Given a measurement noise vector, $\mathbf{v}\ (n_M)$, assumed to be a zero-mean Gaussian white sequence with known covariance matrix $\mathbf{\Sigma}_v\ (n_M)$, represent the actual noisy measurement as

$$\mathbf{y}(n_M) = \mathbf{g}(\mathbf{x}_{M0}, Y_M) + \mathbf{v}(n_M). \tag{12}$$

The likelihood function is then

$$\Lambda(x_M) = \prod_{n_M=1}^{L_M} ce^{-0.5\{[\mathbf{y}(n_M)-\mathbf{g}(\mathbf{x}_M,n_M)]^T\boldsymbol{\Sigma}_v^{-1}(n_M)[\mathbf{y}(n_M)-\mathbf{g}(\mathbf{x}_M,n_M)]\}} \tag{13}$$

and the log likelihood of $\mathbf{x}_M$, dropping the unnecessary normalization constants represented by c in (13), is

$$\lambda(x_M, Y_M) = 0.5\sum_{n_M}^{L_M}\left(\begin{array}{c}\{[\mathbf{y}(n_M)-\mathbf{g}(x_M,n_M)]^T\boldsymbol{\Sigma}_v^{-1}(n_M)\\ \times[\mathbf{y}(n_M)-\mathbf{g}(x_M,n_M)]\}\end{array}\right). \tag{14}$$

Inserting (14) into (10) and using the whiteness of the measurement errors $\mathbf{v}\,(n_M)$, one reaches the formula for the FIM as shown in (15), at the bottom of the page.

Taking the inverse of (15) evaluated at $\mathbf{x}_M = \mathbf{x}_{M0}$ provides the absolute minimum state covariance for $\mathbf{x}_M$ given the set of measurements Y_M. Despite the fact that this bound is practically unrealizable for any filter operating on a real data set, computing this bound at the estimated state $\mathbf{x}_M$ provides a relative estimate of the covariance of the segment's state vector. Consequently, in the implementation of data association algorithms using STI, the estimated state covariance of $\mathbf{x}_M$ at time N is given by $\mathbf{K}_M\,(N) = \mathbf{J}^{-1}$, where $\mathbf{J}$ is calculated according to (15).

With the state covariance calculated, the measurement prediction covariance at time N, $\boldsymbol{\Sigma}_{\hat{y}}(N)$, can be calculated as

$$\boldsymbol{\Sigma}_{\hat{y}}(N) = \{\nabla_x[\mathbf{g}(\mathbf{x}_M,N)^T]\}^T\mathbf{K}_M(N)n_M(L_M) \times\{\nabla_x[\mathbf{g}(\mathbf{x}_M,N)^T]\} + \boldsymbol{\Sigma}_v(N) + \Sigma_d(N) \tag{16}$$

where

$$\boldsymbol{\Sigma}_d(N) = E\{[\mathbf{g}'(\mathbf{x}_M(L_M),\mathbf{d},\Delta_N) - \mathbf{g}(\mathbf{x}_M(L_M),\Delta_N)] \times[\mathbf{g}'(\mathbf{x}_M(L_M),\mathbf{d},\Delta_N) - \mathbf{g}(\mathbf{x}_M(L_M),\Delta_N)]^T\} \tag{17}$$

is the measurement prediction process noise, which represents the uncertainty in the measurement prediction resulting from possible changes in the target motion parameters between the last measurement time $n_M\,(L_M)$ and the time of the predicted measurement N. $\mathbf{x}_M\,(L_M)$ is the target state at the *end* of the segment (rather than the beginning). $\Delta_N = N - n_M\,(L_M)$ is the elapsed time between the last measurement and the time of the desired prediction. The expectation in (17) is taken over the elements of the process noise vector $\mathbf{d}$, which are random variables that characterize the possible changes in the target state motion parameters during the interval Δ_N. The function $\mathbf{g}'(\mathbf{x}_N(L_N),\mathbf{d},\Delta_N)$ is the behavior modified measurement generating equation that calculates the predicted measurement for a given value of $\mathbf{d}$, or in other words it is the function that expresses the effects of the change in state on the predicted measurements.

Calculation of $\boldsymbol{\Sigma}_d\,(N)$ can be difficult for nonlinear motion models, often requiring the evaluation of complicated integrals with no closed form solution. However, if the changes in target state are limited to entering only as additive impulse functions added at the last measurement time $k_M\,(L_M)$, such as an instantaneous change in turn rate or speed, and the elements of $\mathbf{d}$ are assumed to be zero-mean Gaussian variables with known covariance matrix $\boldsymbol{\Sigma}_{\mathrm{dd}}$, $\boldsymbol{\Sigma}_d\,(N)$ can be calculated simply as

$$\boldsymbol{\Sigma}_d(N) = \{\nabla_{\mathbf{x}}[\mathbf{g}(\mathbf{x}_M(L_M),\Delta_N)^T]\}^T \times\boldsymbol{\Sigma}_{\mathrm{dd}}\{\nabla_{\mathbf{x}}[\mathbf{g}(\mathbf{x}_M(L_M),\Delta_N)^T]\}. \tag{18}$$

As the process noise is a design parameter to keep the prediction window large enough to cover target maneuvers (and not an exact representation of knowledge about the target's motion), the simplified computation of (18) is reasonable in many cases.

This completes the evaluation of the measurement prediction and its covariance, but there is still one more small change required to the STI in order to use it in data association frameworks. The STI normally waits until there are enough measurements in a segment to fit the desired motion model before estimating the segment's parameters. Thus, when the current segment has too few points to fit the motion model, no measurement prediction or covariance would be available. To compensate for this, a simpler motion model must be fit for segments that do not have enough measurements to fit the ultimate desired motion model. A good model for this purpose is a simple straight line constant velocity model, in whatever dimensions is appropriate for the data set (one-dimensional, two-dimensional, etc.), as this model can be fit for two measurements, and there are always at least two measurements in a segment after the track has been initialized.

With this modification, the STI algorithm can now be used in data association algorithms that only require one-way communication from the tracking filter to the association algorithm, such as most assignment methods and multiple hypothesis tracking [6]. Additional work is required to use the STI algorithm in PDA algorithms because storing the measurement uncertainty in a way that will be recognized by the STI is required. A method for accomplishing this is presented in the following section.

$$\mathbf{J} = 0.25\sum_{n_M=1}^{L_M}\left\{\begin{array}{c}E[(\nabla_x\{[\mathbf{y}(n_M)-\mathbf{g}(\mathbf{x}_M,n_M)]^T\boldsymbol{\Sigma}_v^{-1}(n_M)\\ \times[\mathbf{y}(n_M)-\mathbf{g}(\mathbf{x}_M,n_M)]\})\\ \times(\nabla_x\{[\mathbf{y}(n_M)-\mathbf{g}(\mathbf{x}_M,n_M)]^T\boldsymbol{\Sigma}_v^{-1}(n_M)\\ \times[\mathbf{y}(n_M)-\mathbf{g}(\mathbf{x}_M,n_M)]\})^T]\end{array}\right\}. \tag{15}$$

C. STI Probabilistic Data Association

Implementation of a PDA algorithm using the STI requires some special consideration. Typically, PDA algorithms using model-based Bayesian state estimators as their tracking filters generate a synthetic innovation and associated covariance from a track's predicted measurement and all the measurements possibly associated with the track, and then return this to the tracking algorithm for use in its update procedure [6]. As the synthetic innovation has a higher covariance than that of a single, unambiguously assigned measurement, its use in the update process increases the covariance of the Bayesian state estimate.

As the STI algorithm does not use or store a state covariance matrix, the uncertainty cannot be passed back to the STI in this fashion. This section presents a method to represent the increased uncertainty from multiple measurements and less than unity probability of detection (data association uncertainty) in an STI-based PDA algorithm. The synthetic innovation is replaced by a synthetic measurement and its covariance, essentially the effective measurement covariance. This effective measurement covariance is stored with the measurement for use in computing the measurement prediction covariance.

This procedure is illustrated by presenting the derivation of the STIJPDAF in its entirety. This presentation closely parallels the presentation of the Kalman filter-based JPDAF algorithm of Bar-Shalom and Li [6]. The STIJPDAF makes the following assumptions.

1) There are a known number of targets currently under track.
2) The past is summarized by the information in the STI filters for the active tracks, including the associated effective measurement covariances.
3) The true measurements are Gaussian distributed around the measurement predictions with known covariance matrix $\mathbf{R}$.
4) The underlying model of the current STI segment for each target is true and correct.

The STIJPDAF computes the measurement to target association probabilities jointly for all targets, and does so only for the current measurement time N. The joint association probabilities are generated by computing the probabilities for all feasible joint association events. A feasible joint association event is one where the following conditions are met.

1) Each measurement is associated with at most one track.
2) Each track has at most one measurement associated with it.
3) The measurement associated with each track lies within that track's validation gate.

The last assumption exists to eliminate the evaluation of highly unlikely tracks. Given a track with predicted measurement $\hat{\mathbf{y}}$ and measurement prediction covariance $\mathbf{\Sigma}_{\hat{y}}$, a measurement $\mathbf{y}$ falls inside the track validation gate when

$$(\mathbf{y}-\hat{\mathbf{y}})^T\mathbf{\Sigma}_{\hat{y}}^{-1}(\mathbf{y}-\hat{\mathbf{y}}) \le \Theta \tag{19}$$

for a given Θ chosen such that the probability of a target lying in the gate is a desired value.

Given a set Y of H measurements at time N, the probability of a particular event $E(i)$ is

$$p(E(i)\,|\,Y) = \frac{\phi!}{cN!}\mu_F(\phi)V_{\text{obs}}^{-\phi}\prod_{j=1}^{H}a(\mathbf{y}_j) \times\prod_{t=1}^{K_T}P_{D_t}^{\delta(t)}(1-P_{D_t})^{(1-\delta(t))} \tag{20}$$

where K_T is the number of tracks, V_{obs} is the sensor observation volume, and P_{D_t} is the probability of detection for track t. The quantity $\delta(t)$ is a binary track detection indicator, whose value is one if track t is detected in $E(i)$, and zero otherwise, and

$$\phi = K_T - \sum_{t=1}^{K_T}\delta(t) \tag{21}$$

is the number of measurements not originating from a track.

The probability mass function $\mu_F(\phi)$ describes the probability of observing a given number of clutter (non-target-originated) measurements. Typical distributions for $\mu_F(\phi)$ are either the Poisson distribution

$$\mu_F(\phi) = \frac{e^{-\lambda V_{\text{obs}}}(\lambda V_{\text{obs}})^{\phi}}{\phi!} \tag{22}$$

which is characterized by the clutter density level λ, or the diffuse prior distribution of [7], $\mu_F(\phi) = \upsilon$, where υ is an unimportant constant that cancels out when the diffuse prior is used. The diffuse prior represents a situation where the clutter density is unknown; it is also referred to as the uninformative prior.

The function $a(\mathbf{y}_j)$ is the likelihood function of the measurement defined by (23), assuming that false alarm measurements are uniformly distributed within the observation volume

$$\begin{aligned} a(\mathbf{y}_j) &= N(\mathbf{y}_j;\hat{\mathbf{y}}_t,\mathbf{\Sigma}_{\hat{y}_t}), \quad \mathbf{y}_j \text{ is associated with} \\ &\qquad \text{track } t \text{ in event } E(i) \\ &= V_{\text{obs}}^{-1}, \quad \mathbf{y}_j \text{ is a false alarm in event } E(i) \end{aligned} \tag{23}$$

where $N(\mathbf{y},\mathbf{m},\mathbf{P})$ is the probability of observing value $\mathbf{y}$ from a Gaussian distribution with mean $\mathbf{m}$ and covariance matrix $\mathbf{P}$.

The normalization constant c in (20) is chosen such that the probabilities of the feasible joint events, representing a mutually exclusive and exhaustive set of the possible data associations, sum to one

$$c = \sum_i p(E(i)\,|\,Y). \tag{24}$$

With the calculation of the probabilities of the joint events completed, for each track t calculate a set of weights β_{jt}

which represent the probability that measurement j is associated with the track t. Also calculate β_{0t}, the probability the track t was not detected at time N. These weights are calculated as

$$\beta_{0t} = \sum_i p(E(i)\,|\,Y)\left(1 - \sum_{j=1}^{H} w_{jt}(i)\right)$$
$$\beta_{jt} = \sum_i p(E(i)\,|\,Y) w_{jt}(i) \tag{25}$$

where the value $w_{jt}\,(i)$ is a binary track to data association indicator equal to one if measurement j is associated with track t in event $E\,(i)$, and zero otherwise.

Up to this point, the STIJPDAF is exactly the same as the Kalman filter-based joint PDA filter (JPDAF), but now instead of calculating a synthetic innovation and its covariance for each track, and then performing the Kalman filter update procedure, the STIJPDAF will calculate a synthetic measurement and the effective measurement covariance for each track. It will then call the STI algorithm to perform the fit and segmentation stage with the synthetic measurement as the latest measurement. For each track t, calculate the synthetic measurement $\mathbf{ys}_t$ and its covariance $\mathbf{\Sigma}_{y\_\text{eff}}$ as follows:

$$\mathbf{ys}_t = \beta_{0t}\hat{\mathbf{y}}_t + \sum_{j=1}^{H} \beta_{jt}\mathbf{y}_j \tag{26}$$

$$\mathbf{\Sigma}_{y\_\text{eff}} = \beta_{0t}\mathbf{\Sigma}_{\hat{y}_t} + \sum_{j=1}^{H} \beta_{jt}\mathbf{\Sigma}_v + \sum_{j=0}^{H} \beta_{jt}\mathbf{y}_j\mathbf{y}_j^T - \mathbf{ys}_t\mathbf{ys}_t^T. \tag{27}$$

The synthetic measurement and its covariance are stored with the track, and are used as $\mathbf{y}(n_M)$ and $\mathbf{\Sigma}_v(n_M)$ in (15) when $n_M = N$ for purposes of computing the segment state covariance. Otherwise, the original implementation of the STI algorithm remains unaltered, with $\mathbf{ys}_t$ used as the measurement and $\mathbf{\Sigma}_{y\_\text{eff}}$ ignored.

Although it is possible to use $\mathbf{\Sigma}_{y\_\text{eff}}$ as a weighting coefficient during the STI least-squares fitting operation, effectively reducing the importance given to measurements that have a high degree of uncertainty, there are drawbacks to such an option. When target maneuvers are large, weighting the synthetic measurement will reduce the STI's ability to respond to changes in target motion, as measurements after initial maneuver onset will likely have a large effective measurement covariance as the spread of the means between the predicted and actual measurement will be large. This large covariance will reduce the effect the measurement has on increasing the model fitting residual and possibly prevent the STI algorithm from properly detecting the need for segmentation, as the segmentation criteria rely on the increase in fitting residual caused by target maneuvers. Essentially, using the effective measurement covariance in this way eliminates the STI's independence from a process noise. Additionally, scaling the measurement residuals makes it more difficult to compensate for the scaling of knot costs versus measurement fitting costs because of the ever-changing weights of the measurement residuals.

III. Single Fish-Tracking Performance Evaluation With Perfect Data Association

In this section the performance of the STI at tracking free-swimming fish is compared against the performance of a Kalman filter, a Kalman smoother, and an EKF. First some background on the particulars of the fish-tracking problem and the performance metrics used in the evaluation are presented. Next the experimental setup that generated the data sets and the data sets themselves are described, followed by a description of the algorithms tested and the tuning parameters of each algorithm. Finally, the results of the comparison are presented and discussed.

A. Fish-Tracking Background and Performance Metrics

Tracking fish for the purposes of ecological studies provides an example of a difficult tracking problem with characteristics not commonly encountered when tracking man-made systems or objects. The goal in these ecological studies is to develop increased understanding of fish behavior and to represent that behavior in a simple fashion that lends itself to behavior classification, statistical analysis, and insertion into forward models of fish behavior and energetics. Unfortunately, the behavior that one wishes to learn, such as fish swimming patterns and maneuver levels, is generally an important input into the models used in sequential state estimation algorithms, and its lack of availability requires us to track targets with unknown behavior. This means that any algorithms that are used must be able to function without using *a priori* information of target behavior (such as maneuver-based process noise levels).

The tracking problem is further complicated by the extreme maneuverability of free-swimming fish, and the need for very accurate swimming speed and acceleration estimates to accurately estimate fish energy expenditure. As an example of fish maneuvering capabilities, Fig. 3 illustrates the fish horizontal swimming speed and turn rate cumulative distribution functions for the data used in the study presented in this section. It shows that fish are capable of turn rates exceeding 100°/s. To put this number in perspective, consider the turn capabilities of aircraft, which are limited by the forces involved in maneuvering at high velocities. An aircraft moving at 300 m/s, a value commonly used in aircraft simulations [8], [9], experiences a force of three times the pull of gravity (3 g's) when turning at only 5.9°/s. Even at 10 g's, pushing the limits of aircraft and their pilots, the turn rate at 300 m/s is only 19°/s. Additionally, fish are capable of bursts of very high linear acceleration, several times their average swimming speed, and they can come to a dead stop or swim backward. Very highly maneuvering targets such as these can present problems for predictive-model-based algorithms because high process noise levels are required to represent the high uncertainty in motion created by the maneuverability, and a large process noise places most of the emphasis on

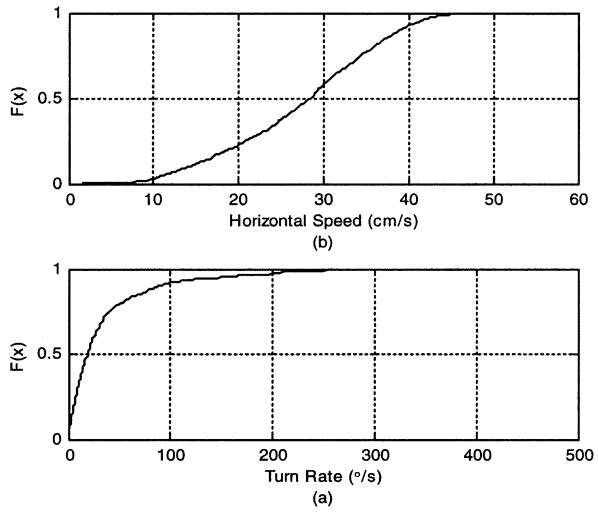

Fig. 3. Cumulative distribution of: (a) swimming speed and (b) turn rates for 100 fish tracks as estimated using the stereo video-camera system.

the measurement in the algorithm's update process, effectively rendering the filter inoperative and creating unnecessarily large errors during periods when the target is not maneuvering.

Very accurate swimming speed estimates are required to estimate fish energy expenditure because the energetic cost of fish swimming increases exponentially with increasing speed due to the nonlinear effects of drag [10], [11]. Thus, small errors in estimating speeds, such as a failure to capture a brief high-speed period or a bias in the distribution of swimming speeds over time, can lead to large errors in estimating fish energy expenditure. Recent work [12]–[14] has also shown that turns and accelerations play a large role in fish energetics, with swimming patterns consisting of several turns or accelerations having an energetic cost several times higher than that of a swimming pattern with steady course and speed.

Given these characteristics of the fish-tracking problem, a set of performance metrics to evaluate the use of tracking algorithms to process fish motion were chosen. The performance metrics stress accurate estimation of the state elements that directly represent behavior (speeds and accelerations) over accurate estimates of fish positions. Additionally, they seek to verify that an algorithm's estimates are accurate not only at individual samples, but also at capturing the overall distribution of values over time, as an error at either level would lead to inaccurate estimates of energy expenditure. Finally, each algorithm's sensitivity to its tuning parameters was tested by evaluating the performance metrics over a wide range of tuning parameter values. This final criterion represents the effects of unknown target behavior on algorithm performance. Given the lack of *a priori* knowledge to use for tuning, algorithms must perform well over a wide range of tuning parameters that might be reasonably chosen based on a rough understanding of target behavior. Otherwise, the value of the filtered state estimates would be in question and the processed data would be of little practical use.

A few final notes before the specific performance metrics are presented. In the desire to keep the data extracted from the study as simple as possible for ready use in ecological models, state estimates were limited to fish position, heading, swimming speed, and turn rate. An emphasis on constant speed, constant angular acceleration (coordinated turn) motions was stressed in an attempt to allow the motion to be described as a set of maneuvers whose length could be determined (again due to the exponential relationship between speed and energy expenditure) and whose motion could be easily classified. Although estimates of tangential accelerations (in the direction of travel) were also desirable, it has been demonstrated [15] that due to the extreme burst accelerations of fish, accurate estimates of fish tangential accelerations from position only measurements require very high frame rates (hundreds of frames per second) and very high accuracy, neither of which is provided by the sonar used in this study. Also, this paper only addresses the study of fish motion in the horizontal plane, as fish physiology [13] often restricts vertical motion to more or less constant speed maneuvers confined to a narrow horizontal plane. This limited vertical motion is effectively tracked using a Kalman smoother and a piecewise constant white acceleration motion model [7]. For details of the vertical motion analysis, see [5].

Five performance metrics were used to evaluate algorithm performance: RMSE for position and speed estimates, median absolute deviation (MAD) for turn rate, and the two-sample Kolmogorov–Smirnov (KS) test probabilities that the estimated speed and turn rate distributions were statistically similar to the ground truth distributions. The RMSE and MAD metrics evaluate the performance of an algorithm on individual samples, with lower values being better. MAD was used in place of RMSE for turn rate as it is more robust to the outliers often created when an algorithm leads or lags a sharp change in turn rate. These outliers can create a high RMSE for an algorithm that performs well other than a small lead or lag on sharp turns. The KS probability evaluates the algorithm's ability to capture the overall distribution of speed and turn rate, with higher values being better and any value over 0.01 considered acceptable.

B. Experimental Setup and Data

A laboratory study was conducted to generate a data set that was used to evaluate the performance of tracking algorithms for sonar-based fish studies. Full details and results of the experiment are given in [5]. The experiment provided a data set of 100 fish tracks simultaneously recorded using a multibeam sonar and a stereo video-camera system in a cylindrical tank 5 m in diameter and 3 m deep. All tracks were selected such that no data association ambiguity existed. The video data, with its higher frame-rate and accuracy, was used as the "ground truth" against which the output of the tracking algorithms processing the sonar data was compared.

Sonar data was recorded at a frame rate of 4 Hz, and the sonar polar measurement errors were calculated empirically as the sample standard deviations of the error between the ground truth estimates and sonar measurements for all measurements for all 100 tracks. The resulting sample means μ

and standard deviations σ are $\mu_R = 0.35$ cm, $\sigma_R = 2.05$ cm for range, $\mu_B = -0.61^\circ, \sigma_B = 0.39^\circ$ for bearing angle, and $\mu_E = 0.04^\circ, \sigma_E = 0.56^\circ$ for elevation angle. Sonar measurements were converted to Cartesian coordinates using the unbiased conversion technique of [16], which also calculates an appropriate Cartesian measurement error covariance matrix given the above polar measurement standard deviations. These converted measurements and associated covariance matrices were supplied to the tracking algorithms.

The video data was recorded at a 29.97-Hz frame rate, and had a mean spherical distance error of 1.08 cm with a standard deviation of 0.70 cm. "Ground truth" reference values were obtained by smoothing the video position measurements along each Cartesian axis with a fifth order Savitzky–Golay filter [17] over 29 data points, and then calculating velocities along each Cartesian axis by differentiating the smoothed video positions. Horizontal speed was calculated as the absolute vector sum of the two Cartesian velocities. Target course was calculated as the arctangent of the two Cartesian velocities, and these course estimates were again smoothed with a fifth-order Savitzky–Golay filter then differentiated to produce the turn rate estimates. Finally, all estimates were down-sampled to 4 Hz to correspond to the sonar measurements.

C. Algorithms and Tuning Parameters

This section provides a brief description of the models used in the Kalman filter algorithms used in this comparison, as well as a description of the tuning parameters associated with these algorithms and the STI. For full details on the algorithms and models, see [5].

The Kalman filter and smoother both used the piecewise constant white acceleration model of [7]. This model filters motion along each Cartesian axis independently and assumes that the motion is constant velocity motion perturbed by small random accelerations which are piecewise constant during each sampling period. The accelerations enter by way of the process noise, which is assumed to be zero-mean, white, Gaussian distributed with known covariance. The smoother was implemented following the presentation of [18], and the smoothing was performed over the entire length of the track. The Kalman filter and smoother both have only one tuning parameter, the standard deviation of the process noise acceleration for each Cartesian axis, set identically equal in this analysis. It was varied from 1 cm/s^2 to 100 cm/s^2 in 1-cm/s^2 increments.

The EKF used a variation of the polar constant speed coordinated turn model of [19]. The original model of [19] assumed the target did not change speed and, thus, supplied a process noise only for angular acceleration, while our model used two independent process noise accelerations, tangential and angular, to allow for changes in both speed and turn rate. The standard deviations of these two process noise accelerations are the tuning parameters of the EKF. The angular acceleration process noise was swept from 10°/s^2 to 100°/s^2 in 10°/s^2 increments, and the tangential acceleration process noise was swept from 1 cm/s^2 to 50 cm/s^2 in 1-cm/s^2 increments.

The STI used the constant speed coordinated turn model described in Section II. A special consideration for the STI algorithm was the choice of the noise standard deviation σ_v used in the STI break condition computations of (3), given that σ_v was not actually constant because the base sonar measurements were in spherical coordinates. The value used was that generated by a measurement which lies at the middle range value of the field of view, 337 cm, and at the middle of the angular limits, 4° in both bearing and elevation, on the basis that this represents approximately the midpoint of the Cartesian errors. Specifically, the Cartesian measurement error covariance for a measurement at this position was calculated using the unbiased spherical to Cartesian coordinated conversion algorithm [16], and the diagonals of the covariance matrix were then used as the measurement variances for each Cartesian axis. The radial measurement standard deviation was used, $\sigma_v = (\sigma_\varepsilon^2 + \sigma_\eta^2)^{1/2} = 3.09$ cm.

The STI was evaluated against the four tuning parameters mentioned in Section II. As these tuning parameters do not have physical analogs like the Kalman tuning parameters, the reasoning behind the selection of their values will be discussed. The first tuning parameter is the noise threshold κ_σ; it should typically be near one so that the segmentation occurs when the fitting cost is worse than expected according to the measurement noise standard deviation σ_v. Varying it gives an idea of the sensitivity to errors in the choice of noise threshold. κ_σ was varied from 0.6 to 2 in increments of 0.2.

Next is the RMSE ratio κ_ψ, which is used to tune track segmentation based on the ratio of the current fit to the best previous fit (when the segment had fewer data points associated with it). If κ_ψ is chosen too large, most segmentation will occur because of the κ_σ break condition. If κ_ψ is close to one (it must be greater than or equal to one or else segmentation will occur for situations where the current fit is *better* than past fits), segmentation will occur too frequently as slightly better previous fits will result in a new segment. κ_ψ was tested for values of two, three, and four.

The third tuning parameter is the tail length τ. τ determines the sensitivity to new maneuvers when the current segment is long. As τ is limited to being an integer between one and the minimum number of measurements required to fit the model, it was natural to sweep τ over the allowable values, $\tau = 1, 2, 3$.

Finally, there is the knot cost tuning parameter κ_C. κ_C determines the relative importance of knot continuity versus the fitting of the measurements within each segment. Choosing very small values implies that continuity is of little importance, whereas choosing very large values is impractical, as it forces all segments to be continuous and essentially ignores the measurements to fit a continuous curve. κ_C was swept from 0 to 2 in increments of 0.2.

D. Single Fish-Tracking Performance Evaluation Results

The results of the performance evaluation are presented as a collection of sensitivity plots for the four algorithms, shown in Figs. 4 through 6, and a table of worst case performance values for each algorithm for each metric shown in

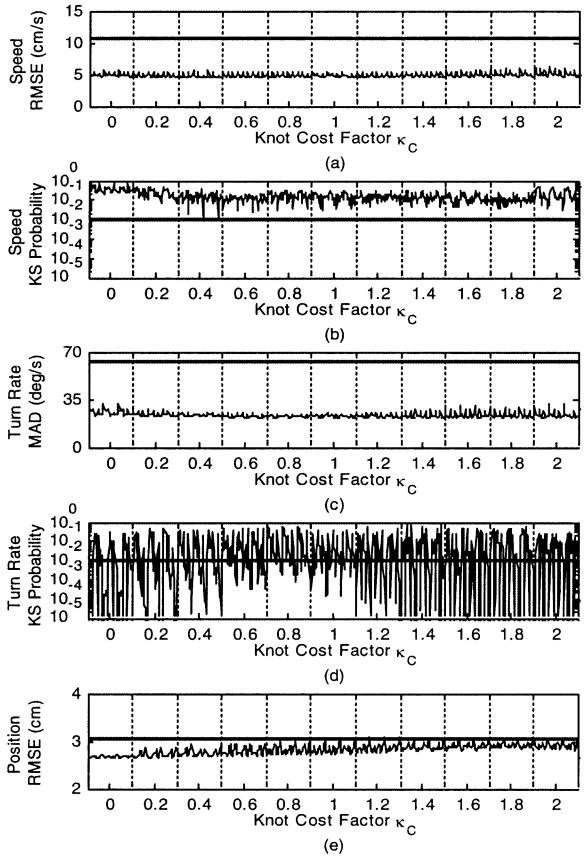

Fig. 4. STI performance and tuning sensitivity plots. The five plots show the STI's performance and sensitivity to five performance metrics. (a) Speed RMSE. (b) Speed KS probability. (c) Turn rate MAD. (d) Turn rate KS probability. (e) Position RMSE. The horizontal bars represent the baseline performance: values obtained without filtering for (a), (c), and (e), and the desired 0.01 probability level for (b) and (d). Lower values are better for (a), (c), and (e), while higher values are better for (b) and (d). Tuning parameters were iteratively varied using nested for-loops, first κ_C, then τ, κ_ψ, and finally κ_σ.

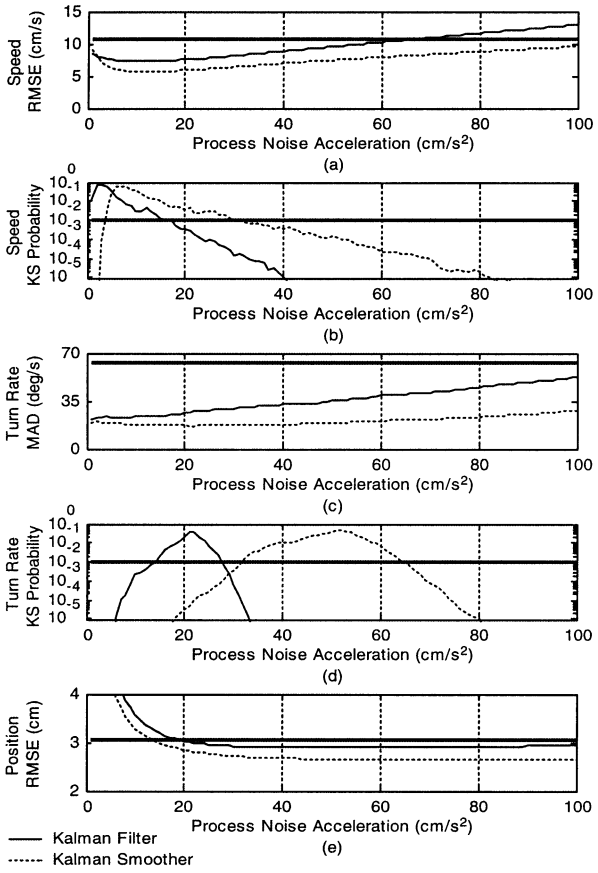

Fig. 5. Kalman filter and smoother performance and tuning sensitivity plots. Format is the same as Fig. 4. Their only tuning parameter, the process noise acceleration standard deviation, was varied from 1 to 100 cm/s^2.

Table 1. The sensitivity plots for each algorithm were generated by calculating the performance metrics for all 100 tracks for a given set of tuning parameters, and then plotting those metrics against the tuning parameter values such that the abscissa of each plot represents the Cartesian product of that algorithm's tuning parameters. The Cartesian product is expressed by varying the individual tuning parameters along the abscissa in nested for-loops. The plot for an ideal algorithm that is completely insensitive to its tuning parameters would consist of a flat horizontal line for all performance metrics. This format provides an easy way to visually evaluate the overall sensitivity of an algorithm to its tuning parameters, although in the case of the STI with its four tuning parameters, it is admittedly somewhat difficult to evaluate the effects of each individual parameter.

Baseline reference values are provided as horizontal lines in the sensitivity plots. For the RMSE and MAD statistics, the reference values shown are the results obtained by processing the sonar data equivalently to how the video data was processed to obtain the ground truth, but with no smoothing. This is equivalent to no filtering at all, and a good algorithm should provide errors whose values are below these reference values. The reference value for the KS probabilities is 0.01, a commonly accepted level for a good match when using the KS test. A good algorithm would provide probabilities that are higher than 0.01.

Examining Figs. 4 through 6, it can be seen that no algorithm completely meets the desired goal of being insensitive over the entire tuning parameter space while outperforming the baseline. The STI performs best, achieving the lowest speed RMSE and presenting remarkable insensitivity to the choice of tuning parameters for all but the turn rate KS test. The STI algorithm's turn rate MAD performance is second to that of the Kalman smoother, and, as shown in Table 1, the STI algorithm is the only algorithm that always outperforms the baseline in all speed and turn rate performance metrics, regardless of the choice of tuning parameters. As expected, very high knot continuity costs (emphasizing smoothness over fit of the measurements) result in position RMSEs that slightly exceeds the RMSE of the unfiltered sonar measurements (Maximum STI RMSE of 3.10 cm versus 3.04 cm for the measurements.)

Only for the turn rate KS test is the STI especially sensitive to tuning parameters. From the turn rate KS plot in Fig. 4, it is obvious that the knot continuity cost κ_C can have a large effect on turn rate performance, with values at

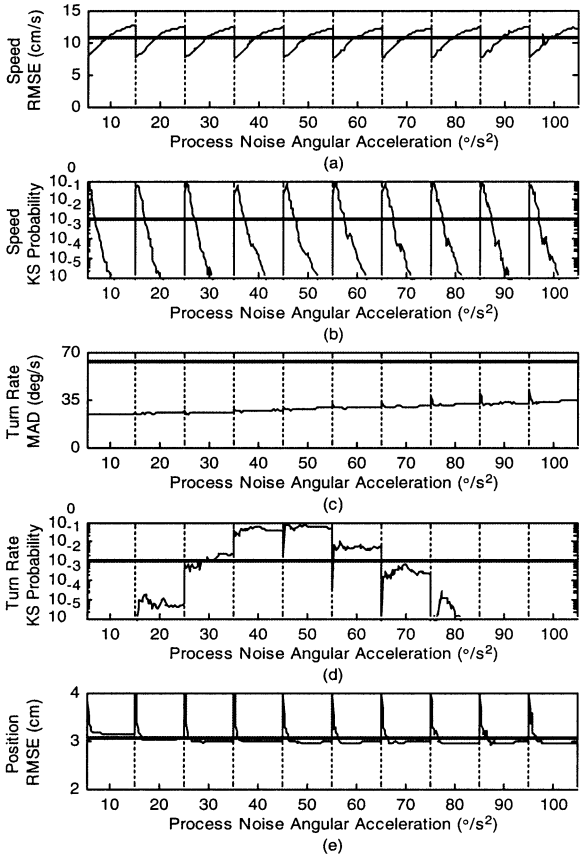

Fig. 6. EKF performance and sensitivity tuning sensitivity plots. Format is the same as Fig. 4. Tuning parameters were iteratively varied using nested for-loops, first process noise angular acceleration, and then process noise tangential acceleration.

Table 1 Single Fish-Tracking Worst Case Results—Lower Values Are Better for RMSE and MAD, Higher Values Are Better for KS Probabilities

Algorithm	Speed RMSE (cm/s)	Speed KS Prob.	Turn Rate MAD (°/s)	Turn Rate KS Prob.	Position RMSE (cm/s)
No Filtering	10.58	6.58e-8	63.29	2.93e-18	3.04
STI	6.25	1.20e-2	32.24	3.40e-15	3.10
Kalman Filter	12.90	4.23e-13	52.49	4.59e-51	9.31
Kalman Smoother	9.56	2.62e-11	28.16	2.92e-142	9.04
EKF	12.63	3.05e-8	41.78	6.67e-20	4.49

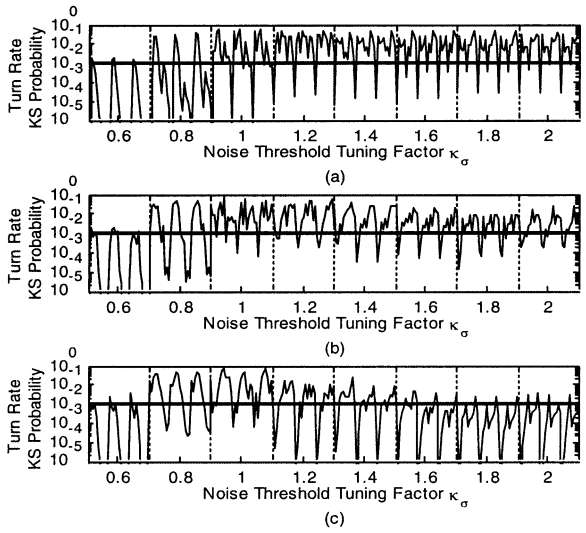

Fig. 7. STI turn rate KS probability performance and tuning parameter sensitivity for: (a) $\kappa_\psi = 2$; (b) $\kappa_\psi = 3$; (c) $\kappa_\psi = 4$ is shown as a function of the remaining three tuning parameters which were iteratively varied using nested for-loops, first κ_σ, then τ, and finally κ_C. The horizontal bars represent the desired 0.01 probability level. These plots present an alternative view to that of Fig. 4(d) to more readily illustrate the sensitivity to individual tuning parameters.

or slightly below one being the best choice. An alternate view of what is happening with the turn rate KS probabilities for the STI algorithm is shown in Fig. 7, which presents the variation in turn rate KS probability versus the noise segmentation threshold parameter κ_σ for all three values of the RMSE ratio break level tuning parameter κ_ψ. When the noise segmentation level is set correctly, $\kappa_\sigma = 1$ (correct because the measurement standard deviation is known and supplied correctly), performance is generally good regardless of the value of κ_ψ. If κ_σ is too low, performance is always poor, as segmentation occurs too often. As κ_σ goes higher, the lower values of κ_ψ improve results because the lower break ratio allows proper segmentation despite the noise segmentation threshold being set too high. When both values are high, performance is again poor. The dependence on κ_C can still be seen, with values near one being the most consistent performers, as low values of κ_C tend to be best when κ_σ is small, and higher values of κ_C are best when κ_σ is larger. The tail length tuning parameter τ plays very little role, as can be seen by the repeating triples of performance on each graph between the major grid lines. These triples are created by the variation of τ from one to three.

Although the STI's KS turn rate probability performance varies considerably with changing tuning parameters, it is not quite as bad as it seems from the graphs; for reasonable performance, one needs only either knowledge of the sensor's measurement error characteristics (represented by $\kappa_\sigma = 1$), or proper selection of the knot continuity cost, with values of knot continuity cost at or slightly below one being good choices. If either of these conditions is met, the variance with the remaining three tuning parameters is relatively small.

All the Kalman algorithms are sensitive to the choice of tuning parameters and have difficulties achieving simultaneously good performance on each of the five performance metrics. The constant velocity Kalman filter and smoother both suffer from the problems that the process noise acceleration levels that produce the best results on the speed performance metrics produce poor position results, and similarly there is little overlap between the process noise levels that provide good KS probabilities for the speed and turn rate distributions. The smoother clearly outperforms the filter, demonstrating the performance gains available through smoothing. The smoother also achieves the lowest turn rate MAD of all algorithms, and its peak horizontal speed RMSE is second only to the STI algorithm. Why the smoother outperforms the STI in turn rate MAD is uncertain, but it may indicate that the more restrictive constant speed, constant turn rate model of the STI limits its ability to estimate the fine structure of the

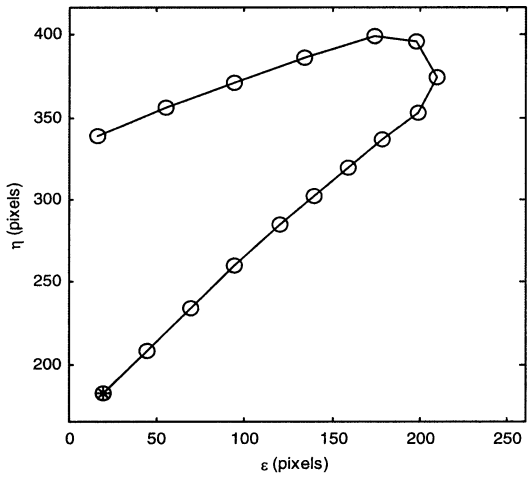

Fig. 8. Simulated fish track 1 of a fish performing a sharp U-turn. The graph axes are camera pixel coordinates. ◯ represents a measurement, and * represents the start of the track.

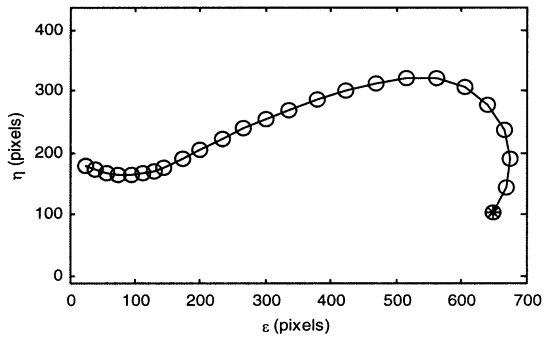

Fig. 9. Simulated fish track 2 of a fish performing three relatively slow consecutive linked turns. The graph axes are camera pixel coordinates. ◯ represents a measurement, while * represents the start of the track.

fish turn rate compared to the independent two-axis motion used for the Kalman smoother. However, it should be noted that the independent two-axis motion model makes breaking fish motion down into easily classified individual maneuvers more difficult, and the ability to achieve such a breakdown was one of the goals of the study. Additionally, the inability of the Kalman smoother to simultaneously provide quality estimates of both swimming speed and turn rate tends to diminish the usefulness of the gains it provides in turn rate MAD.

The EKF never achieves the low levels of speed RMSE obtained by the STI and constant velocity Kalman smoother, and is very dependent on proper selection of the tangential acceleration process noise level for accurate speed estimation. Additionally, it often performs worse than the baseline achieved without filtering on the speed and position RMSE tests.

The worst case results in Table 1 provide an idea of the level of confidence one could have in the filtered output in a blind study, one without a baseline data set to validate the results or to use to tune the algorithm. This table clearly demonstrates how an improperly tuned predictive filter can provide results far worse than no filter at all, essentially destroying the utility of a data set. Here is where the advantage of a data-driven algorithm such as the STI becomes clearly evident when tracking targets with unknown behavior. The STI outperforms the no filtering option on every parameter except position RMSE for the entire broad range of tuning parameters over which it was tested. And even in position RMSE, the worst case penalty imposed is only 0.06 cm, a rather small error compared to the typical sensor measurement error of 3.09 cm.

IV. Verification of the STIJPDAF

This section presents results obtained using the STIJPDAF algorithm developed in Section II. These results are only meant to demonstrate that the STIJPDAF filter performs essentially as one would expect any JPDAF algorithm to operate, thus demonstrating the ability to install the extended STI algorithm into existing data association frameworks in situations where it may be more suitable to the underlying single-target tracking problem than other filters. Data association problems are generally complicated and require specific solutions tailored to specific problems. The authors are aware that a more complex data association algorithm, such as one that accounted for limited sensor resolution and merged targets, would likely perform better, especially in the case of the tracking of actual fish video data.

A. Single Fish in Clutter Tracking

The effectiveness of the STIJPDAF at tracking a single fish in clutter is compared against a standard JPDAF [6] using an EKF tracking filter, termed the EKFJPDAF. The comparison is performed using Monte Carlo simulations of two representative fish tracks in clutter. The fish tracks were extracted from video data, and then preprocessed by fitting to a coordinated turn model and modifying them so that there were no discontinuities in target course between segments, eliminating the discontinuities that give the EKF difficulty. The resulting tracks are shown in Figs. 8 and 9.

A set of simulations were run for each simulated track, with the following parameters varied for each simulation: measurement noise variance σ_v^2; Poisson clutter density parameter λ; and target probability of detection PD. Specifically, simulations for each track were performed for all combinations of the parameters as follows:

$$\sigma_v^2 = \{2, 5, 10\}$$
$$\text{PD} = \{1, 0.9, 0.8\}$$
$$\lambda = \{1\times10^{-7}, 5\times10^{-7}, 1\times10^{-6}, 5\times10^{-6}, \ldots, 1\times10^{-4}\}.$$

Noisy Cartesian coordinate measurements were generated by adding mutually independent white Gaussian noise of variance σ_v^2 to each axis of the simulated track positions. Missing measurements were simulated by randomly removing measurements according to the probability of detection. The amount of clutter (non-target-originated measurements) generated for each sampling time was determined according to a Poisson distribution model

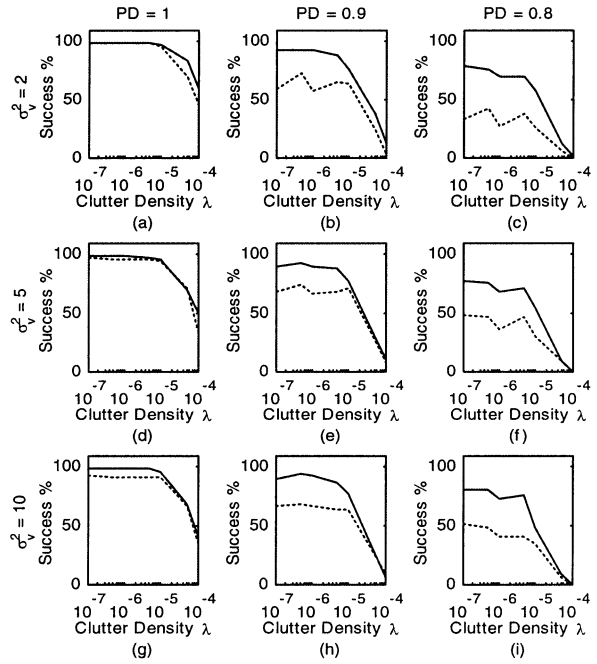

Fig. 10. Percentage of 100 Monte Carlo simulations of track 1 (Fig. 8) where the target was successfully tracked from start to finish for various levels of Poisson clutter density parameter λ, probability of detection (PD), and measurement noise variance σ_v^2 (in pixels^2). Solid line is the STIJPDAF; dashed line is the EKFJPDAF. (a) PD $= 1, \sigma_v^2 = 2$. (b) PD $= 0.9, \sigma_v^2 = 2$. (c) PD $= 0.8, \sigma_v^2 = 2$. (d) PD $= 1, \sigma_v^2 = 5$. (e) PD $= 0.9, \sigma_v^2 = 5$. (f) PD $= 0.8, \sigma_v^2 = 5$. (g) PD $= 1, \sigma_v^2 = 10$. (h) PD $= 0.9, \sigma_v^2 = 10$. (i) PD $= 0.8, \sigma_v^2 = 10$.

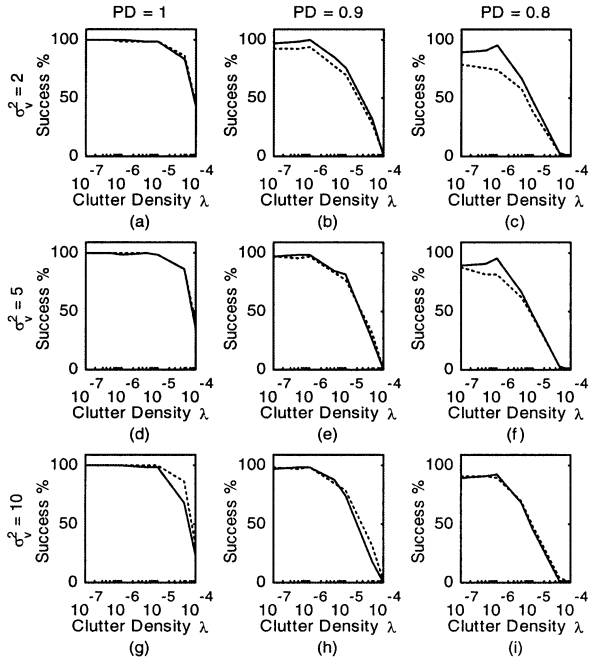

Fig. 11. Percentage of 100 Monte Carlo simulations of track 2 (Fig. 9) where the target was successfully tracked from start to finish for various levels of Poisson clutter density parameter λ, probability of detection (PD), and simulated measurement noise variance σ_v^2 (in pixels^2). Solid line is STIJPDAF; dashed line is the EKFJPDAF. (a) PD $= 1, \sigma_v^2 = 2$. (b) PD $= 0.9, \sigma_v^2 = 2$. (c) PD $= 0.8, \sigma_v^2 = 2$. (d) PD $= 1, \sigma_v^2 = 5$. (e) PD $= 0.9, \sigma_v^2 = 5$. (f) PD $= 0.8, \sigma_v^2 = 5$. (g) PD $= 1, \sigma_v^2 = 10$. (h) PD $= 0.9, \sigma_v^2 = 10$. (i) PD $= 0.8, \sigma_v^2 = 10$.

with density parameter λ. The clutter measurements were distributed uniformly throughout out the entire observation volume (720×480 pixels).

Both JPDAF algorithms used the diffuse prior distribution for their clutter density mass function, and both were supplied with the correct measurement noise variance and probability of detection for each simulation. The validation gate for each algorithm was a four-sigma gate [$\Theta = 16$ in (19)]. Each algorithm was initialized using the first three measurements, which were always detected and supplied clutter and noise free (perfect initialization).

The STI motion model is the constant speed coordinated turn model described previously, with tuning parameters $\kappa_C = 1, \kappa_\psi = 3, \kappa_\sigma = 1, \tau = 3$. The measurement prediction process noise for the STIJPDAF was generated using (18) with the process noise covariance matrix

$$\begin{aligned}\mathbf{Q}_{zz} &= \mathrm{diag}[0 \quad 0 \quad 0 \quad \delta_s \quad \delta_\omega] \\ &= \mathrm{diag}[0 \quad 0 \quad 0 \quad 65 \quad 0.1\pi] \end{aligned} \tag{28}$$

where $\mathrm{diag}[\cdot]$ represents a square matrix whose values are zero except on the diagonal, where the values are as provided inside the brackets in order from the upper left corner to the bottom right corner of the matrix, δ_s is the linear acceleration impulse change and δ_ω is the impulse change in turn rate.

The EKF used the coordinated turn model discussed in the previous section. However, the process noise was expanded to include process noise in all state parameters not just in the speed and turn rate, as performance was greatly improved on track 1 with the additional process noise. The total additive process noise used in the EKF state covariance prediction equation is

$$\begin{aligned}\mathbf{Q} &= \mathrm{diag}[\delta_\varepsilon \quad \delta_\eta \quad \delta_s \quad \delta_\theta \quad \delta_\omega] \\ &= \mathrm{diag}[15 \quad 15 \quad 20 \quad 0.02\pi \quad 0.05\pi]. \end{aligned} \tag{29}$$

The process noise for each filter was selected by tuning the filters to achieve good performance on both tracks in the absence of clutter.

The performance metric used to compare the two algorithms was the percentage of runs in which the target was successfully tracked from start to finish. A track was declared successful using the criterion of [20] and [21]; a track was successful if at no point along the track did the estimated position error exceed $10\sqrt{2}\sigma_v$. The results for the simulations are shown in Figs. 10 and 11 for tracks 1 and 2, respectively. The plots show that the STIJPDAF was at least as effective as the EKFJPDAF in these simulations. The STIJPDAF performed better for track 1 because of the STI's overall better ability to respond to the sharp turn compared to that of the EKF. Both algorithms performed very similarly on the smoother second track.

As mentioned at the beginning of this section, the point of the simulations is to show that the STIJPDAF has similar abilities to the EKFJPDAF, not to make a claim as to which is better in general. It is likely that a set of tracks that are relatively smooth but with high measurement noise could be created to show the EKFJPDAF outperforming the STIJPDAF.

B. Multiple Fish Tracking

Approximately 1 h of video from the top camera was processed using the STIJPDAF to test its ability to track multiple targets in a low-clutter, low-noise environment. We used the video data instead of the sonar data because in our experimental setup the sonar was unable to resolve individual fish in schools, resulting in an intractable multiple target tracking scenario. Instead the sonar data rates were simulated by extracting data from every 8th video frame for a sampling rate of $29.97/8 \approx 3.75$ Hz.

Fish were detected in each video frame by first subtracting a frame with no fish (subtracting the background) from the current frame, converting the remaining video red–green–blue (RGB) values to black and white by thresholding, dilating and eroding the black and white data to group points close together, and then accepting as a target any group of white pixels with an area greater than 10 pixels. The centroid of the pixel group was used as the location the measurement.

The measurements supplied to the STIJPDAF algorithm were Cartesian coordinate measurements in camera pixel coordinates, with the estimated measurement noise variance along each axis used by the STIJPDAF equal to 5 pixels. The distributed prior distribution was used for the clutter density mass function. The STI motion model was the same coordinated turn model as described previously, with tuning parameters $\kappa_C = 0.5, \kappa_\psi = 3, \kappa_\sigma = 1, \tau = 3$. A low value of κ_C was used so that the measurement prediction would be strongly based on the new measurements rather than unduly guided by continuity in heading from the previous segment. Continuity and quality fitting in the completed tracks was maintained by using $\kappa_C = 1$ for the recursive optimization stage. As all segments were ultimately fit using recursive optimization, using two different values of κ_C allowed for good prediction after sudden sharp turns, without any degradation in the final track estimate from too low of a knot continuity cost factor.

In order to implement the STIJPDAF on real data, additional logic for track initiation and track drop was required. For this study, track initiation and drop logic were designed to be conservative, with missed tracks being more acceptable than false alarms. The reason for this choice is that in a biological study aimed at determining animal behavior, it is better to have a smaller but correct data set than a larger data set with a lot of false information.

Tracks were initiated using a "two out of three" detection logic. Valid points for starting new tracks were those that were not associated with an existing track in the most likely joint event. Any single-point track initiator that did not have a second point in its search window at the proceeding sampling time was dropped. Once a two-point preliminary track was established the "two out of three" detection logic was activated. A two-point preliminary track was promoted to a valid track if at least two detections occurred in its validation gate within the next three sampling times. Thus, a minimum of four samples was required to initiate a track.

Preliminary tracks were able to share points with other preliminary tracks, but not with confirmed tracks. When a preliminary track was promoted to a confirmed track, all preliminary tracks that shared points with the new confirmed track were eliminated. If two preliminary tracks that shared points were both promoted to confirmed tracks in the same sampling period, the track with the smallest STI fitting cost was confirmed and the other track eliminated.

Track drop logic was an augmented "three out of five" system. Tracks were dropped if there were no measurements inside their validation gate during three out of the last five sampling times, with situations where the track's probability of a missed detection was greater than 95% counted as missed detections for purposes of track drop logic. Additionally, tracks were dropped if their predicted location was outside the field of view. This condition was the primary means of dropping tracks as all fish eventually swam out of the field of view.

The STIJPDAF-generated tracks were validated by overlaying them onto their corresponding video frames and viewing the results frame by frame to determine if each track was valid or not. Valid tracks were taken as those tracks which followed only a single fish from the start of the track to end of the track, but it was not required that the fish be tracked over the entire time it was in the field of view.

The performance metrics are the probability of detection and the false alarm rate. The probability of detection is defined as the number of true tracks that were detected (true positives) divided by the total number of true tracks. The false alarm rate is the percentage of detected tracks that were false, or the number of false tracks divided by the total number of detected tracks.

A total of 710 valid fish tracks were detected, with 37 instances of a fish swimming through the field of view going undetected (missed tracks), for a probability of detection of 95%. A large portion of the undetected tracks consisted of tracks where the fish swam across a corner of the field of view and was only detected for fewer samples than the track detection logic would accept. The total number of false tracks was 126, with seven of them resulting from nonfish objects (shadows, surface debris), and 54 of them representing tracks that were made up of partial or complete tracks from two different fish, in other words tracks that swapped from one fish to another once, but were otherwise valid. With 126 out of 836 (710 + 126) tracks being false tracks, the false alarm rate is 15%.

These statistics are considered to be quite good, as this data set presents a very difficult multiple target tracking scenario. The difficulty comes from the fish's tendency to alter their motion when approaching another fish, performing such actions as slowing, stopping, changing direction, or some combination of the above. While it is not surprising that fish alter their behavior in response to approaching another fish, this is a problem for position-only measurements data association where the only information available to make the association decision is prediction based on previous motion. The need for predictable motion is most critical when two fish cross paths, as this is when the association decisions are difficult,

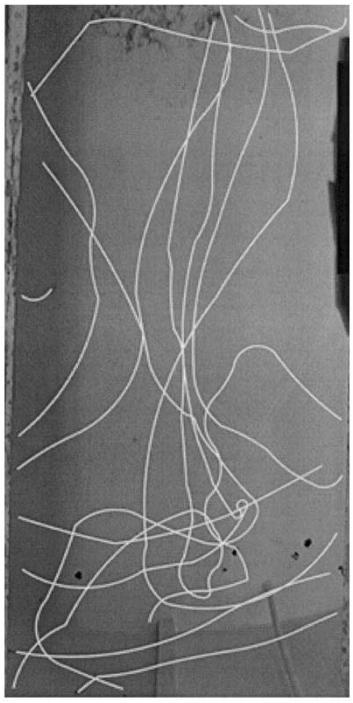

Fig. 12. Composite video image showing 14 fish tracks recorded at ∼3.75 Hz during a 25-s sequence of video data. All tracks were successfully tracked using the STIJPDAF.

but this is also the time when the fish are least likely to behave in a predictable fashion.

An example of the complexity presented in this data set is shown in Fig. 12, which shows the 14 fish tracks that were present in approximately 25 s of video data (94 frames at ∼3.75 Hz). All 14 tracks were tracked successfully by the STIJPDAF during this sequence.

V. Conclusion

It is well known that tracking highly maneuverable targets with unknown behavior presents difficulties for predictive-model-based Bayesian state estimators. There are essentially two main problems in using these estimators in this situation. The first problem concerns the inability to properly tune the filter due to a lack of knowledge about the system process noise. The fish-tracking experiments in this paper clearly demonstrated the poor results these filters can provide when the process noise is selected improperly.

The second problem comes from the high maneuverability of the targets, because high process noise levels are required to represent the high uncertainty in motion created by the maneuverability. A large process noise places most of the emphasis on the measurement in the algorithm's update process, effectively rendering the filter inoperative and creating unnecessarily large errors during periods when the target is not maneuvering. Although this paper has focused on the use of a data-driven method to track maneuvering targets, a wealth of literature exists that addresses the problem of tracking maneuvering targets with predictive-model-based Bayesian state estimators. Bar-Shalom and Li [7] discuss various adaptive solutions including adaptive noise levels, least-squares input estimation, and variable state dimension filters, although they do not rate their performance very highly. More popular are the switching multiple model approaches, such as the interacting multiple model (IMM) approach [22], very powerful for its computational complexity, and the Gaussian wavelet estimator (GWE) [23], [24], which becomes especially useful at low sampling rates [25], as well as particle filter-based multiple model methods [26], [27] helpful in cases where some of the models are nonlinear or the Gaussian distributed merged estimates of filters like the IMM and GWE are inappropriate. One caveat on multiple model methods is that Li [28] has shown them to be very sensitive to proper model selection. This indicates that the use of multiple model methods to attack the problem of unknown behavior is likely to prove difficult.

Another alternative are the various variable structure multiple model (VSMM) filters [8], [9], [28]–[32]. These VSMM models may be more suitable to covering the wide range of motion presented by highly maneuvering targets, although to achieve stability their adaptation rates must be limited. This may limit their effectiveness for problems with low sampling rates, such as the sonar fish-tracking problem.

The use of data-driven methods for target tracking has not received as much attention in the tracking literature as the predictive Bayesian methods. This paper has demonstrated the potential advantages to data-driven methods for tracking highly maneuvering targets with unknown behavior, focusing specifically on a curve segmentation and fitting approach. This approach relies more on knowledge of the system measurement errors than on knowledge of the process noise. Its use of segmentation to represent maneuvers frees it from the need to select a process noise to represent maneuvers not directly accounted for in the state transition function, and also provides an ability to respond very rapidly to abrupt changes in target motion while still achieving accurate results during the quiescent periods of target motion. Additionally, model selection for algorithms based on this approach is reduced to simply choosing the parameter space on which fitting is performed.

One algorithm that follows this approach, the STI, was discussed in this paper. It works by partitioning a track into segments that each represent a single maneuver and then performing a least-squares fit of a motion model-based cost function to the data in that segment to estimate the target state. The STI algorithm was shown to be very effective at tracking free-swimming fish. It produced accurate estimates of swimming speeds and turns rates while maintaining insensitivity to the choice of its tuning parameters. This insensitivity is a critical aspect of any algorithm that must provide reliable data for scientific purposes when little or no *a priori* information is available for use in tuning. The tuning parameter sensitivity was especially low when the sensor measurement error characteristics were modeled correctly.

An extension to the STI to generate a measurement prediction and its covariance was presented for use in data association frameworks, and an STI-based JPDAF algorithm using this extension was developed. The derived STIJDPAF was shown through simulations to be effective at tracking single

fish in clutter and through the use of real-world video data to be effective at tracking multiple free-swimming fish.

ACKNOWLEDGMENT

C. Schell would like to thank his thesis adviser J. S. Jaffe for providing support and lab resources to perform the fish-tracking experiment described in this paper.

REFERENCES

[1] S. P. Linder, M. D. Ryan, and R. J. Quintin, "Concise track characterization of maneuvering targets," presented at the AIAA Conf. Guidance, Navigation, and Control, Montreal, QB, Canada, 2001.

[2] P. L. Rosin and G. A. W. West, "Segmenting curves into elliptic arcs and straight lines," in *Proc. 3rd Int. Conf. Computer Vision*, 1990, pp. 75–78.

[3] S. Hsin-Teng and H. Wu-Chih, "Multiprimitive segmentation of planar curves-a two-level breakpoint classification and tuning approach," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 21, pp. 791–797, Aug. 1999.

[4] L. Wenyin and D. Dori, "Incremental arc segmentation algorithm and its evaluation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 20, pp. 424–431, Apr. 1998.

[5] C. Schell, "Advanced tracking algorithms for the study of fine scale fish behavior," Ph.D. dissertation, Univ. California, San Diego, 2003.

[6] Y. Bar-Shalom and X. R. Li, *Multitarget-Multisensor Tracking: Principles and Techniques*. Storrs, CT: YBS, 1995.

[7] ——, *Estimation and Tracking: Principles Techniques, and Software*. Storrs, CT: YBS, 1998.

[8] V. P. Jilkov, D. S. Angelova, and T. A. Semerdjiev, "Design and comparison of mode-set adaptive IMM algorithms for maneuvering target tracking," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 35, pp. 343–350, Jan. 1999.

[9] E. Semerdjiev, L. Mihaylova, and X. R. Li, "Variable- and fixed-structure augmented IMM algorithms using coordinated turn model," in *Proc. 3rd Int. Conf. Information Fusion*, vol. 1, 2000, pp. 25–32.

[10] J. R. Brett and T. D. D. Groves, "Physiological energetics," in *Fish Physiology*. New York: Academic, 1979, vol. 8, pp. 279–352.

[11] P. W. Webb, "Hydrodynamics and energetics of fish propulsion," *Bull. Fisheries Res. Board Canada*, vol. 190, pp. 1–158, 1975.

[12] M. Tang, D. Boisclair, C. Menard, and J. A. Downing, "Influence of body weight, swimming characteristics, and water temperature on the cost of swimming in brook trout (Salvelinus fontinalis)," *Canadian J. Fisheries Aquatic Sci.*, vol. 57, pp. 1482–1488, July 2000.

[13] P. W. Webb, "Composition and mechanics of routine swimming of rainbow trout oncorhynchus-mykiss," *Canadian J. Fisheries Aquatic Sci.*, vol. 48, pp. 583–590, 1991.

[14] D. Boisclair and M. Tang, "Empirical analysis of the influence of swimming pattern on the net energetic cost of swimming in fishes," *J. Fish Biol.*, vol. 42, pp. 169–183, 1993.

[15] D. G. Harper and R. W. Blake, "A critical analysis of the use of high-speed film to determine maximum accelerations of fish," *J. Exp. Biol.*, vol. 142, pp. 465–472, 1989.

[16] L. Mo, X. Song, Y. Zhou, K. Sun Zhong, and Y. Bar-Shalom, "Unbiased converted measurements for tracking," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 34, pp. 1023–1027, July 1998.

[17] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C, The Art of Scientific Computing*, 2nd ed. New York: Press Syndicate Univ. Cambridge, 1996.

[18] S. Roweis and Z. Ghahramani, "A unifying review of linear gaussian models," *Neural Comput.*, vol. 11, pp. 305–345, Feb. 1999.

[19] F. Gustafsson and A. J. Isaksson, "Best choice of coordinate system for tracking coordinated turns," in *Proc. 35th IEEE Conf. Decision and Control*, vol. 3, 1996, pp. 3145–3150.

[20] S. R. Rogers, "Diffusion analysis of track loss in clutter," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 27, pp. 380–387, Mar. 1991.

[21] D. Avitzour, "Stochastic simulation Bayesian approach to multitarget tracking," in *IEE Proc. F, Radar, Sonar Navigat.*, vol. 142, Apr. 1995, pp. 41–44.

[22] H. A. P. Blom and Y. Bar-Shalom, "The interacting multiple model algorithm for systems with Markovian switching coefficients," *IEEE Trans. Automat. Contr.*, vol. 33, pp. 780–783, Aug. 1988.

[23] D. D. Sworder, J. E. Boyd, R. J. Eliott, and R. G. Hutchins, "Data fusion using multiple models," in *Conf. Rec. 34th Asilomar Conf. Signals, Systems and Computers*, vol. 2, 2000, pp. 1749–1753.

[24] C. T. Leondes, D. D. Sworder, and J. E. Boyd, "Multiple model methods in path following," *J. Math. Anal. Appl.*, vol. 251, pp. 609–623, Nov. 15, 2000.

[25] D. D. Sworder and J. E. Boyd, "Measurement rate reduction in hybrid systems," *J. Guidance Control Dynamics*, vol. 24, pp. 411–414, Mar.–Apr. 2001.

[26] R. Karlsson and N. Bergman, "Auxiliary particle filters for tracking a maneuvering target," in *Proc. 39th IEEE Conf. Decision and Control*, vol. 4, 2000, pp. 3891–3895.

[27] A. Doucet, N. J. Gordon, and V. Krishnamurthy, "Sequential simulation-based estimation of jump Markov linear systems," in *Proc. 39th IEEE Conf. Decision and Control*, 2000, pp. 1166–1171.

[28] X. R. Li and Y. Bar-Shalom, "Multiple-model estimation with variable structure," *IEEE Trans. Automat. Contr.*, vol. 41, pp. 478–493, Apr. 1996.

[29] X. R. Li, Z. Xiaorong, and Z. Youmin, "Multiple-model estimation with variable structure. III. Model-group switching algorithm," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 35, pp. 225–241, Jan. 1999.

[30] X. R. Li and Z. Youmin, "Multiple-model estimation with variable structure. V. Likely-model set algorithm," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 36, pp. 448–466, Apr. 2000.

[31] X. R. Li, Z. Youmin, and Z. Xiaorong, "Multiple-model estimation with variable structure. IV. Design and evaluation of model-group switching algorithm," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 35, pp. 242–254, Jan. 1999.

[32] X. R. Li, "Multiple-model estimation with variable structure. II. Model-set adaptation," *IEEE Trans. Automatic Control*, vol. 45, pp. 2047–2060, Nov. 2000.

Chad Schell (Member, IEEE) received the B.S. degree in electrical engineering from the University of New Mexico, Albuquerque, in 1996, and the M.S. and Ph.D. degrees in electrical engineering (applied ocean science) from the University of California, San Diego, La Jolla, in 2000 and 2003 respectively.

From 1991 to 1996, he worked at Sandia National Laboratories in the Intelligent Systems and Robotics and the Microsensors divisions. He currently conducts research on geolocation systems at Rincon Research Corporation in Tucson, AZ, and operates his own consumer electronics business.

Dr. Schell received a National Science Foundation fellowship.

Stephen Paul Linder received the B.S. degree in mechanical engineering from the Massachusetts Institute of Technology, Cambridge, in 1982 and the M.S. degree in computer systems engineering and the Ph.D. degree in electrical and computer systems engineering from Northeastern University, Boston, MA, in 1996 and 1998, respectively.

He was with the Applied Research Laboratory, Pennsylvania State University, University Park, working on target tracking and sensor data fusion, and a Faculty Member in computer science at the State University of New York, Plattsburgh. He is currently in the Computer Science Department, Dartmouth College, Hanover, NH, and is working on projects that include the track prediction of bouncing ball using limited data rate cameras, tracking of cell phones using antenna arrays, and dynamic health assessment of first responders. He also teaches classes in mountaineering or sea kayaking when not sitting in front of the computer. See http://alum.mit.edu/www/spl for more details.

James R. Zeidler (Fellow, IEEE) received the Ph.D. degree in physics from the University of Nebraska, Lincoln, in 1972. From 1974 to 2003, he was a Scientist at the Space and Naval Warfare Systems Center, San Diego, CA, and from 1988 to 2003, he was an Adjunct Professor of Electrical and Computer Engineering (ECE) Department at the University of California, San Diego (UCSD). He is currently a Senior Research Scientist/Senior Lecturer in the ECE department at UCSD and a faculty member of the Center for Wireless Communications at UCSD and of the California Institute for Telecommunications and Information Technology at UCSD and the University of California, Irvine. He has written more than 200 technical publications and holds 12 U.S. patents.

Dr, Zeidler received the IEEE Frederick Ellersick best paper award at the 1995 IEEE Military Communications Conference, the Lauritsen-Bennet award for achievement in science in 2000, and the Navy Meritorious Civilian Service Award in 1991. He was an Associate Editor of the IEEE TRANSACTIONS ON SIGNAL PROCESSING from 1991 to 1994.