Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2013, Article ID 160718, 11 pages
http://dx.doi.org/10.1155/2013/160718

Research Article

Radar Tracking System Using Contextual Information on a Neural Network Architecture in Air Combat Maneuvering

Antonio Navidad Pineda, Luis Usero Aragonés, José Raúl Fernández del Castillo Díez, and Miguel Ángel Patricio Guisado

Departamento de Ciencias de la Computación e Inteligencia Artificial, Universidad de Alcalá de Henares, Edificio Politécnico, Campus Universitario, Alcalá de Henares, Madrid, P.C. 28871, Spain

Correspondence should be addressed to Antonio Navidad Pineda; antonio.navidad@uah.es

Received 18 February 2013; Revised 4 June 2013; Accepted 2 July 2013

Academic Editor: Jesús García

Copyright © 2013 Antonio Navidad Pineda et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Air surveillance radar tracking systems present a variety of known problems related to uncertainty and lack of accurately in radar measurements used as source in these systems. In this work, we feature the theoretical aspects of a tracking algorithm based on neural network paradigm where, from discrete measurements provided by surveillance radar, the objective will be to estimate the target state for tracking purposes as accuracy as possible. The absence of an optimal statistical solution makes the featured neural network attractive despite the availability of complex and well-known filtering algorithms. Neural networks exhibit universal mapping capabilities that allow them to be used as a control tool for capturing hidden information about models learned from a dataset. We use these capabilities to let the network learn, not only from the received radar measurement information, but also from the aircraft maneuvering context, contextual information, where tracking application is working, taking into account this new contextual information which could be obtained from predefined, commonly used, and well-known aircraft trajectories. In this case study, the proposed solution is applied to a typical air combat maneuvering, a dogfight, a form of aerial combat between fighter aircraft. Advantages of integrating contextual information in a neural network tracking approach are demonstrated.

1. Introduction

Tracking algorithms have a widespread use and increased sophistication in aerial control and surveillance systems (both military and civilian) where current scenarios demand a great capability on the *tracking* and *surveillance* of a large number of objects moving across a vast aerial space. Measurement data will be obtained from many and diverse sensors generating information related to those objects. Tracking, in the context of law enforcement or military activities, implies also noncooperating parties whereas in commercial activities usually leverages active exchanges among parties to facilitate identification and minimize risks.

Inferring the value of a quantity of interest from a series of indirect, inaccurate, and uncertain measurements is called *estimation* [1]. Moreover it can be seen as the process of selection of a point from a continuous multidimensional space, the *best estimate*.

Figure 1 shows a typical flow chart depicting a state estimation process. First two stages are usually *black boxes* enclosing the underlying (hidden) true state of the system, while the last stage is fed by state measurements where a noise component is incorporated, causing uncertainty and lack of accuracy which should be minimized by the estimation process that conforms the estimation problem.

Estimation is used as a mechanism to reduce possible uncertainty and inaccuracy, generating information that has the following properties:

(i) quality (i.e., accuracy, reliability) higher in the estimate than the raw measurements;

(ii) contain information not directly available in the measurements.

Decision can be viewed as the selection of one of a set of discrete alternatives, the *best choice* from a discrete space.

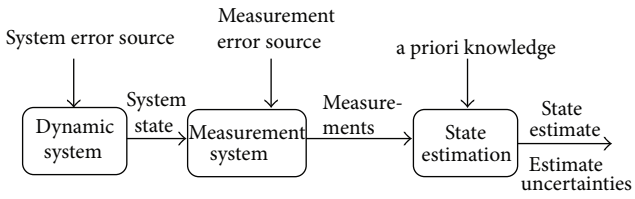

Figure 1: State estimation scheme.

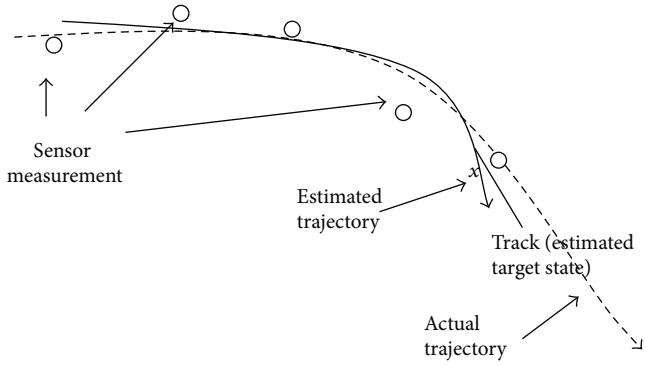

Figure 2: Tracking problem scheme.

However, decision can also provide a metric to evaluate a probability of various alternatives.

Tracking [1] is the process of estimation of the state of a moving object. This is done using one or more sensors to provide measurements.

Tracking is wider in scope than estimation. Tracking not only uses all the tools from estimation but it also requires the use of statistical decision theory to improve the description of objects and inferring of future change in their properties.

More specifically, in the concrete case of *air surveillance systems*, *tracking* consists of processing measurements to gather and maintain precise knowledge on target state, physical properties (position and its derivatives) about moving targets, as presented in Figure 2.

The *measurements* are always affected by error sources in form of noise, derived from subtle variations of the process dynamics, the underlying environment, or sensors imprecision. Those phenomena alter the sensing and measurement of an object position (usually distance, azimuth, and altitude).

Those measurements incorporate to tracking process *uncertainty* (measurement association to targets, false detections,...) and *inaccuracy* (target position error) [2].

The estimator can make use of a priori knowledge about

(i) physical system dynamics (evolution of the variable),

(ii) sensor characteristics (measurement system),

(iii) probabilistic models of certain *random* factors (uncertainties) and the prior information, where models about the target behavior could be considered.

Modeling the possible behavior with a priori knowledge about the target movement can play a significant role in the estimation process. To incorporate a priori knowledge to the estimation process, significant difficulties will be added to this process.

An *Optimal filter* can be seen as an algorithm that processes observations, providing estimate of a variable of interest minimizing a certain error criterion (usually called cost function). A possible weakness of these optimal filters is their sensibility to errors existing in the models and excessive computational cost preventing their use for *real time object tracking*.

A number of tracking algorithms have been reported in the literature [3–7] over the years each mainly differing in the way a dynamical system model of maneuvering target being tracked is obtained.

The classical purely statistical algorithms, derived upon stochastic filtering, like α-β-γ [8, 9] filter or state-model filters like *Kalman* [7, 9, 10] filters, cannot always perform well when the model is error prone, unrealistic, or there is high noise in the measurement process, even when the possible target movement modeling has not been taken into account in benefit of other typical cases. So that, while both of the above approaches have certain strong points and can deliver good tracking performance in specific scenarios, there are some shortcomings as well. Some of them are computational complexity and difficulty in accurately modeling faster-turn and coordinated-turn maneuvers and the difficulty in including additional input features for the estimation process.

This computational complexity is considered critical in typical cases of air control centers where tracking system must estimate hundreds of targets (aircraft) with thousands of measurements in a short period of time, 3–10 seconds, in close coordination with other multiple functions related to this [11]. Accordingly this deadline is passed, and results obtained are considered obsolete.

Alternate approaches that can deliver improved tracking performance are of particular interest.

Artificial neural networks (NN) can generate arbitrary mappings allowing the identification of complex or unknown processes and prediction of future values, one step ahead, capturing complex behaviors when rule-based systems cannot be used or are not available. This property is especially useful when it is required to work in environments where prediction rules cannot be easily defined or where uncertainty or inaccuracy data are used.

Specifically in the case of a radar tracking system, NN training information can be set up from data extracted directly from the provided radar measurements, but also this information can be augmented by *contextual information*, deriving knowledge from a domain expert from context situation where tracking algorithm is being executed.

In this work, such contextual information can not only enclose multiple sensor data but also know expected patterns in air maneuvering which probably will define the target behavior.

The ability to efficiently fuse information of different forms for facilitating intelligent decision making is one of the major capabilities of trained multilayer NN that is being recognized in recent times [12–14].

Such approach results in an overall nonlinear tracking filter which has several advantages over the popular efforts

at designing nonlinear estimation algorithms for tracking applications, the principal one being the reduction of mathematical and computational complexities.

As a case of application it is proposed to consider and augment contextual information with a common case of dogfight, fighter air combat maneuvers used from World War I, allowing the NN to estimate future position values for complex trajectories, especially those that arise when two or more fighter aircraftsare in close combat.

In a tracking system, these dogfight techniques are hard to incorporate to classical statistical filters or particle filters, basically because these algorithms do not incorporate contextual information beyond pure statistical variables.

In this document, a proposal of neural network tracking algorithm (in a scope of radar air surveillance) dealing with this contextual information, in addition to the classical radar measurement information considered in this kind of algorithms, will be presented.

2. Contextual Real Time Neural Tracking

2.1. Framework. The scheme shown below tries to mimic the general workflow used by experts of the domain (i.e., air traffic controllers) while trying to infer a proper estimate of actual position for an unknown aerial mobile object based on radar surveillance data acquired on a regular basis.

(i) Taking into account the last measurements, estimating velocity (speed and direction), a prediction of the next *approximate* position, is inferred for the tracked object (*predicted estimation*).

(ii) From this *predicted estimation*, considering *contextual information* (i.e., trajectory followed by this target, relative positions between close radar targets, and existence and location of a possible enemy,. . .), along with a *priori knowledge* such as modeling of basic maneuverings, combat tactics [15] and other available data in the field, like Rules of Engagement (RoE, rules or directives to military forces that define the circumstances, conditions, degree, and manner in which force or actions which might be construed as provocative,), may be applied which will allow us to make corrections to the initial *predicted estimation*, getting a *filtered estimation* [16].

As an example of basic maneuvering model, in this application case, we will use a specific dogfight *pursuit curves* [17] scenario.

A pursuit curve depicts trajectories followed by two objects, the "rabbit" that tries to escape from the action radius of the "fox" that tries to catch the former. These trajectories imply two factors.

(1) The fox moves always in the direction of the rabbit, trying to catch it.

(2) The fox varies its speed accordingly (proportional) to the rabbit.

This trajectory is commonly known as "pure pursuit" in the dogfight argot, and it is used when the pursuer wants to maintain aiming of its weapons directly at the pursued aircraft. Both aircraft move on a similar path based on the premise of trying to hit and avoiding to be shot.

The pursuer trajectory will be conditioned to the pursued trajectory, taking into account the factors previously described.

This information of how threatened aircraft responds can be used as a priori knowledge while attempting to estimate future values of position and velocity for both type of aircraft (pursued and pursuer).

A common characteristic of many tracking systems, in context of air surveillance, is to consider them as real time systems: their requirements specification includes timing restrictions in the form of deadlines. Informally, a safety real time system can be defined as one in which the damage incurred by a missed deadline is greater as time passes beyond the deadline [11, 18]. These real time systems are needed in a number of application domains including air traffic control.

In a real-time system timely availability [19] of the results is as important as accuracy. Future actions depend on both, but it is even more important to have result in time to allow the system to react to event than having too accurate results but too late, when the environment has changed. This will condition the proposed solution as an attempt of getting an optimum relation "time of response/quality of response".

To guarantee proper operation of a real-time system we need to predict (to some extent) the general conditions to meet with a real time constrains in order to ensure the system will be responsible in the stated terms for the whole time of operation [20].

2.2. Neural Tracking: State of the Art. Open literatures on target maneuvering estimation from noisy position data from radar data are very scanty. A detailed literature survey has been carried out by Ananthasayanam et al. [21] and Amoozegar [22].

Different tracking algorithms have been developed as α-β filter, α-β-γ filter [8, 9] and Kalman filter [7, 9, 10]. Due to the faster computing speed of current computers, more and more systems use Kalman filter to track the target. In practical systems there are many factors originating from the tracked targets and the tracking system that lead to target loss (i.e., a sudden maneuver of a target implies a tracking system that it is accelerating unexpectedly, causing a bias in the measurement sequence, will result in a target loss).

Though the Kalman filtering is a fundamental building block for target maneuvering estimation, the NN techniques have also been used by some researchers to improve the estimation accuracy of Kalman estimates. In several cases [23, 24] a neural network-aided Kalman filter tracker has been proposed to improve the accuracy of EK-estimated position and velocity. Improvement in estimation using NN in planar situation has been demonstrated. It has been also reported that by aiding NN with Kalman filter, estimation accuracy in both position and velocity improves considerably.

So, NN techniques (several along with fuzzy techniques) were been used to maneuver detection or system variance

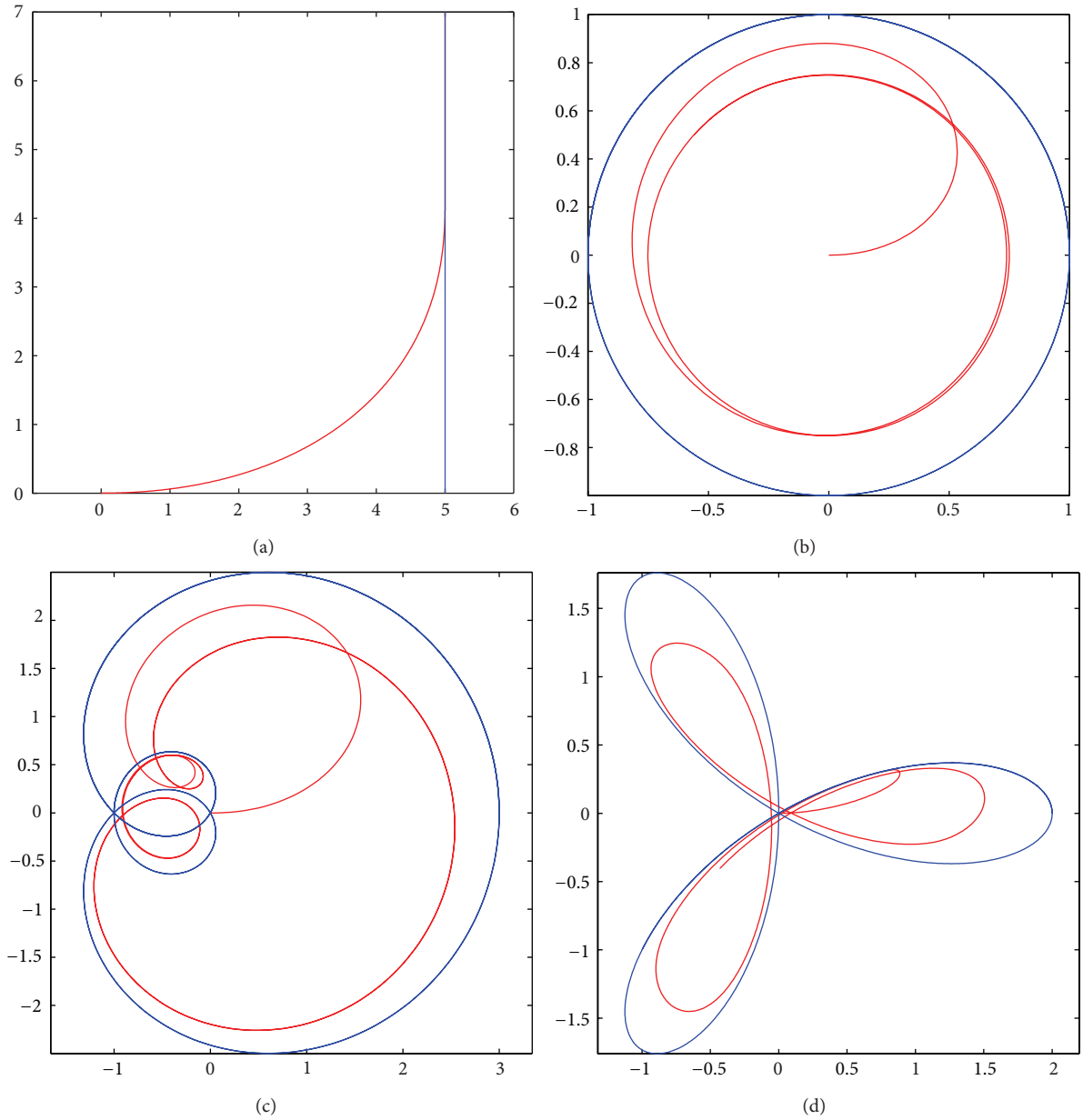

Figure 3: "Pursuit curves" (in blue pursued, in red pursuer).

adjustment of the filter [25] which help to improve the estimation obtained by a Kalman filter.

Chin [23] employed the backpropagation neural network to aid the Kalman filter to reduce the estimation error. The output of a trained NN is used to compensate the state estimation. This algorithm does not change the structure or parameter of the standard Kalman filter.

The defect for these approaches is that they will not be easy to compensate well enough when large errors are generated by the Kalman filter state.

Sundareshan [12] demonstrates the ability of a NN-based tracking solution to realize improved tracking under maneuver conditions using additional data, collected from a diverse set of sensors. In this case the tracking solution is improved using information which is possible to extract from the measurements from imaging sensors (such as laser radar, infrared sensor, or a sensor operating in the visible frequency ranges).

Kong et al. [26] test whether a NN can be used to replace the function of Kalman filter. After some modifications have been made to the input/output vectors of the NN, the problem of the radar tracking system using only measured data is solved. No additional information is used in this solution.

On the other hand, NN have been used for learning air combat maneuvers, not with the aim of solving a tracking problem but also to improve air combat maneuvering strategies, providing training value to the users of simulators [27].

Schvaneveldt et al. [28] explore the applicability of NN models in the domain of air combat maneuvering, incorporating in these models knowledge about air combat maneuvers and components of maneuvers as well as rudimentary knowledge about maneuver planning and situational awareness. The authors provide a review of Air Combat Expert Simulation, a presentation of the implementation and testing of the NN models and an overview of a software system developed.

It is very usual to develop tracking systems, especially applied to visual scope, which employ contextual information as inputs to neural networks to address specific problems and improve their performance [29, 30].

The *main contribution* of this paper is to integrate in a tracking algorithm based on NN technique the available knowledge on air combat maneuvering strategies. In this way, data from the environment, taken as *contextual information*, which should be decisive in the trajectory decision for an aircraft implied in a combat, are fused in a NN architecture to obtain a new state estimation. The considered data are derived, not only from the own radar measurements for the target, but also from the overall global situation where combat is taken place.

2.3. Proposed Neural Tracking Algorithm. In order to achieve the objective of our case study, we separated estimation and contextual enhancement into two different stages, both using artificial neural networks as a basis to mapping and information fusion.

(i) *Basic estimation neural network (prediction NN)*. This network operates as a classic particle filter allowing inferring an initial rough estimate (predicted state) of a tracked aircraft, where approximately our target might be found.

(ii) *Enhancement neural network (smoothing NN)* [31, 32]. This second network augments previous estimate with a priori information modeled upon basic maneuvering, *contextual information* which will allow us to obtain a more accurate estimate.

The artificial neural network used in this study is the well-known multilayer perceptron (Figure 4) with a supervised training framework based on the technique of error back-propagation [33], and the number of input elements of both networks is constrained to the bare minimum number of neurons needed to characterize the input space and the desired output while the number of layers will be constrained to three using a unique hidden layer to connect input with output space.

In our case study, as a simplification of the problem, the input space considered will be 2D. So, radar measurements, generally referenced to its own position, will be range and azimuth to the detected target. It is usual to transform those coordinates into Cartesian ones (X, Y), related again to the sensor (radar) ground position. In our case, we will use these Cartesian coordinates.

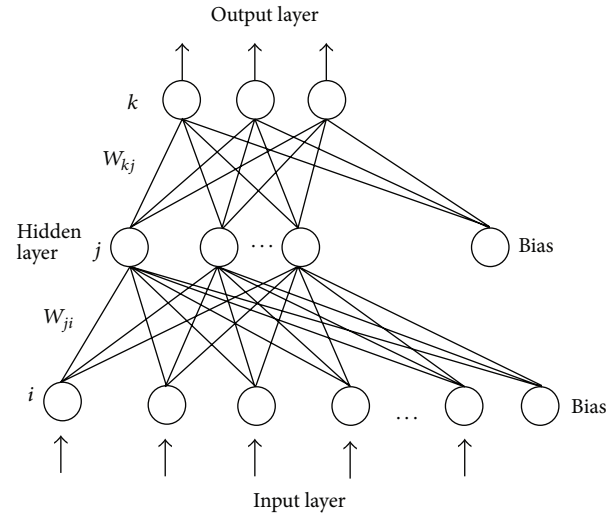

Figure 4: Multilayer perceptron network. i, j, k: layers, W_{xy}: link weights.

In the proposed solution, different NNs, with the same scheme but different training, will be applied independently to coordinates X and Y.

For an "ideal" situation of air space using, considering the target as an independent statistical variable, where no previous condition exists on its movement, the NN used for the components X and Y would be exactly the same, insomuch as its behavior in the NN will be exactly the same.

Nevertheless, in a real world, taking into account that an aircraft trajectory is influenced by diverse external items (i.e., its takeoff point), it can be observed that different instances of NN must be used for each component.

We set the position measured from the radar data at time t as

$$y(t) = x(t) + v(t), \tag{1}$$

where $y(t)$ is obtained from the "true" $x(t)$ position that should be ideally measured plus additive noise $v(t)$ from a known probability distribution.

As input to the *prediction NN* which is used the last samples provided from the sensor (radar), using Cartesian coordinates: $y(t), y(t-1), y(t-2), y(t-3), \ldots X$ or Y component $[yx(i), yy(i)]$ will be used each one in its own NN.

Using absolute coordinates for X and Y components makes trajectories that look almost equal to be very different when they start from far locations. Thus it makes training to be very slow.

To overcome this limitation, in our prediction NN, we use incremental sampling [26] $[\Delta y(t) = y(t) - y(t-1)]$. Therefore the network will infer also estimates as an increment $[\Delta yxpe(t+1), \Delta yype(t+1)]$, instead of absolute locations (Figure 5). This allows greater convergence rates and makes estimates invariant to absolute positions, being the response given by the NN conditioned only by the maneuver previously learnt, where $\Delta yxpe(t+1)$ will be the component X predicted for the measurement that should be received in time $t+1$.

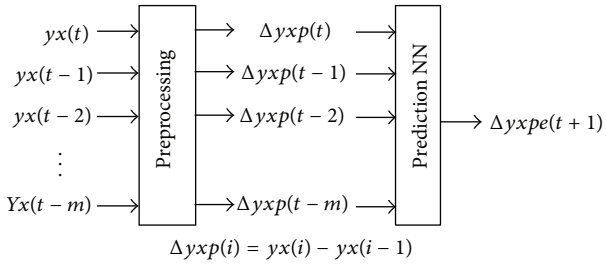

Figure 5: Prediction NN.

Training data for this stage, using coordinates X, Y in different NN, can be generated upon simulated pursuit trajectories, where sampling incorporates additive gaussian noise $v(t)$ to mimic real data as those gathered from real devices or from actual data provided by a real radar.

While it is desirable to have a proper output data to validate network results, generally aircraft real position $x(t)$ is impossible to know. Yet, considering an off-line *training* for our NNs, we can try two different approaches to this problem.

(1) Simulated data make possible to know this information as we can store samples ($x(t)$) before noise ($v(t)$) addition just to use them for *validation*.

(2) Using an off-line processing with an optimal filtering algorithm, without real time requirements, accurate enough to get estimates from a set of measurements that could be used as a validating source during the training phase.

In this study, option 1 was chosen as the good enough alternative given the current scope of this work.

For the next stage, *smoothing NN,* the previous estimate is adjusted using a second network where *contextual information* extracted from the relative aerial situation of aircraft is used as inputs to this NN. It means that estimated features, location, or speed components of an aircraft status will be estimated to be conditioned by this contextual information. This is the great advantage of using this NN for tracking in maneuvers previously leant, considering contextual information that in other case (classical algorithms) will be impossible, or at least computationally very expensive, to be taken into account.

In dogfight-pursuit case, a key factor for the pursuer will be the *angle* to be maintained between it and the pursued. In close combat scenarios, viability of weapons is restricted to short-range defensive guns. Modern fighters have only a forward firing fixed gun that can be used in short-range dogfight. Thus pursuit trajectories are usually one of "lead pursuit" "pure pursuit" and "lag pursuit" designed to position the attacker in the optimal firing position while avoiding overshooting the other aircraft and risk being exposed to shooting instead.

This *angle* between pursuer and pursued trajectories, as well as other features as relative distance, relative speed, and so forth. can be incorporated as inputs to the *smoothing NN*, and therefore consider this information as *contextual information* that will be used in the tracking process.

In the same way that in prediction NN, the smoothing NN will receive as inputs the previously predicted increments $\Delta yxpe(t)$, $\Delta yype(t)$ (by prediction NN) and trajectory form being tracked, but we augment the information the network will use with the components for distance between both aircraft ($\delta xp(t)$ and $\delta yp(t)$), a key feature that will allow the network to make a better estimate for a pure pursuit style.

$\delta xp(t)$ and $\delta yp(t)$ will be calculate as the difference in absolute position (X or Y component) between the pursuer and pursued estimated position (it is an evidence that tracking algorithm does not know actual position for pursuer and pursued).

Absolute position (X or Y component) will be obtained adding the successive $\Delta yxs(t)$, $\Delta yys(t)$ increments to the previously absolute position of the target (track initialization is not considered in this paper), where $\Delta yxs(t)$ is obtained as the output of smoothing NN (Figure 6).

This contextual information has been considered as useful in this kind of trajectories, by the proper definition of these curves. Nevertheless, in other trajectories, the contextual information to be applied must be considered, taking into account its appropriate parameterization to be used as a proper input for smoothing NN. In fact, the inputs in the smoothing NN should be defined according to this contextual information used in this tracking process.

The combined two stages for *neural tracking* (Figure 6) make final estimates more accurate and robust than trying to make the combined work into a single neural network. Also training times showed to be lower than using a single network.

Starting from actual radar measurements, a neural-based tracking algorithm has been defined to incorporate typical noise as well as define contextual information useful in this case of dogfight pursuit curves.

In simulation results for this case of study (Section 3 of this paper), it is observed that considering the last four measurements received from radar and using a single hidden layer with five neurons, the estimates obtained from this tracking system considerably improves the estimates for the α-β filter [8, 9].

Since the structure of both neural networks is clearly defined (previous paragraph), the computational cost for this approach can be perfectly dimensioned and evaluated according to the requirements proper of a real time system. The time deadline for each execution of this tracking process is totally bounded, and so it can be real time scheduled, enabling us to evaluate if reacting to data received will be within time intervals dictated by the environment [34]. Taking into account that training is executed offline, the training time must not be considered for these real time requirements.

3. Simulation Results

To train neural networks used in this case, a set of pursuit curves, with random errors added to the measurements

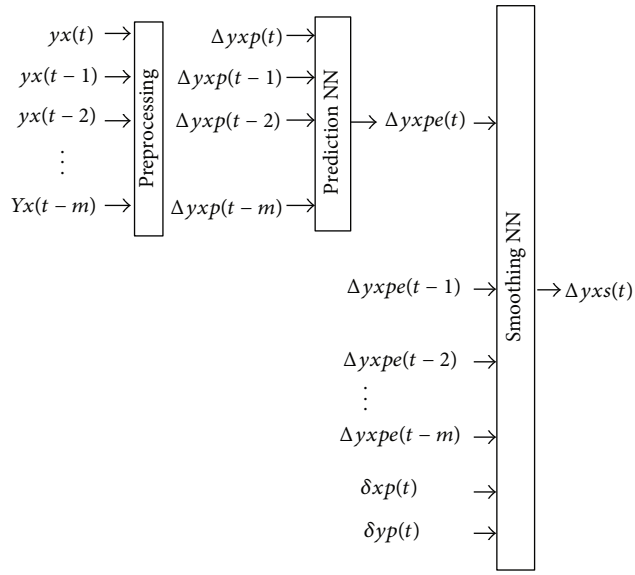

Figure 6: Dual stage neural tracking (X component position).

obtained for each curve, are used. This set of pursuit curves includes different curves (Figure 3) and similar curves changing parameters in their definition. Nevertheless random noise is generated for each specific case, so that no case is equal to other, even when the same curve has been taken as reference.

To validate the trained neural networks new pursuit curves are designed, with different mathematical definition or changing parameters in curves previously defined. As it was said before, in any curve definition new random noise is added to this curve so that the data for any new curve will always be different in each new case.

In any case, to train and to validate neural networks data with random noise will be used, using "perfect" data (without noise) only to compare the result curve with the "optimal" estimate.

In Figure 7, results obtained by the NN tracking algorithm proposed in this document are shown as representative graphical examples. Previously NNs have been trained with a certain set of pursuit curves (Figure 3), defined by different parameters. Any case, the curves used and defined for training and validation are always affected by a random noise.

Along with estimated curve, other useful curves are presented. Next, what each curve, depending on its color, represents is explained.

(i) Blue. Trajectory for the *pursued* plane. In this path *no error* has been added. It is not used in tracking process.

(ii) Black. Trajectory for the *pursuer* plane. In this path *no error* has been added. This trajectory is conditioned by a *pursuit curve* determined by the *pursued* plane (blue trajectory). It is used as reference position to calculate errors in estimate positions and not used in tracking process.

(iii) Red. Trajectory for the pursuer plane considering white noise added to the previous trajectory. It simulates the actual radar detection. This trajectory, as the previous one, is conditioned by a pursuit curve determined by the pursued plane (blue trajectory). This data together with pursued data (affected by noise, blue line plus noise) are used to calculate contextual information to consider in tracking process.

(iv) Magenta. Estimated trajectory obtained by tracking process from red trajectory according only to the predicted increases by the prediction NN.

(v) Green. *Final estimated trajectory* by *the smoothing* NN, obtained by tracking process from *red* trajectory incorporating context information considered in the estimation. This is the *final estimation* of the proposed NN tracking algorithm.

(vi) Yellow. Estimated trajectory by α-β filter (g-h filter, $\alpha = 0.5$) [8, 9], classical filter. It is used as comparative data.

Samples shown in Figure 7 have been obtained from a unique trained neural network tracking entity (prediction NN and smoothing NN for X and Y component, four trained neural networks).

Axis incorporates a scale to be able to compare the different results obtained in the tracking process. Measurement units are not relevant, but considering a typical fighter-to-fighter combat each unit could be considered as one tenyh of nautical miles. Nevertheless, the same curves for helicopters combat would imply different scale and therefore different units.

Once the tracking NNs (prediction NN and smoothing NN) has been trained, subjecting the NN architecture to a set of validation data the following results are observed.

As additional information, like a magenta trajectory, it is shown in the chart the first NN stage output, the *prediction NN*, where no contextual information is considered. This incremental values are the input for the second NN stage, the *smoothing NN*, which provide us the final result (green trajectory).

As training data for both neural networks a set of trajectories, always with random noise generated for each measurement, has been used. Each trajectory is compound by 100 data. Three different training data sets have been considered:

(a) a set of 10 trajectories (100 data per trajectory),

(b) a set of 20 trajectories,

(c) a set of 50 trajectories.

Once both neural networks were trained (a, b, or c), a validation data set of 20 trajectories (again 100 data per trajectory) was considered. The tracking process result for the neural tracking algorithm proposed in this paper is compared with classical *α-β filter* (g-h filter, $\alpha = 0.5$) [8, 9].

To compare the results between neural filter and classical filter the following metrics are calculated for each tracked trajectory.

Table 1: Numerical results for different training cases (with 20 trajectories for validation).

	NT MSE-X	CT MSE-X	NT MSE-Y	CT MSE-Y	CV X	CV Y
			TS = 50			
Average	0,00434596	0,011301885	0,004514052	0,0115171	0,854828377	0,83420158
Minimum	0,003265019	0,009954887	0,003731276	0,00888355	0,746030615	0,64807045
Maximum	0,005497492	0,01299676	0,005526362	0,01324728	0,998696361	1,0484481
			TS = 35			
Average	0,00489405	0,01234154	0,00475068	0,01115477	0,85303732	0,85359688
Minimum	0,00397036	0,01076685	0,00359512	0,00806498	0,75620495	0,74105283
Maximum	0,00588607	0,0142457	0,00568751	0,01382003	0,96719392	1,06696146
			TS = 20			
Average	0,00479757	0,01202544	0,004705733	0,01126898	0,87205893	0,83188561
Minimum	0,00398021	0,01077879	0,003563132	0,00927324	0,69621001	0,67934719
Maximum	0,00574476	0,01397357	0,005927035	0,01349507	1,06844267	1,00220355
			TS = 15			
Average	0,00549193	0,01214101	0,00544172	0,01199342	0,87569316	0,83879903
Minimum	0,00380993	0,01011885	0,0041497	0,00983185	0,76183417	0,68802747
Maximum	0,00683425	0,01452005	0,00631471	0,01356212	1,00188802	0,9665112
			TS = 10			
Average	0,005817	0,01196749	0,005719515	0,0112769	0,89183096	0,8069522
Minimum	0,00470605	0,01021932	0,005033928	0,00887119	0,784827	0,64173938
Maximum	0,00753874	0,01393356	0,006757601	0,01438699	1,04596164	0,95789102

Table 2: Average results for different training cases (with 20 trajectories for validation).

TS	NT MSE-X	NT MSE-Y	CT MSE-X	CT MSE-Y	CV X	CV Y
10	0,005817	0,005719515	0,01196749	0,0112769	0,89183096	0,8069522
15	0,00549193	0,00544172	0,01214101	0,01199342	0,87569316	0,83879903
20	0,00479757	0,004705733	0,01202544	0,01126898	0,87205893	0,83188561
35	0,00489405	0,004750675	0,01234154	0,01115477	0,85303732	0,85359688
50	0,00434596	0,004514052	0,01130189	0,0115171	0,85482838	0,83420158

TS: training set size (number of trajectories); NT MSE-X: MSE component X neural tracking; NT MSE-Y: MSE component Y neural tracking; CT MSE-X: MSE component X classical tracking; CT MSE-Y: MSE component Y classical tracking; CV X: CV component X for Neural Tracking; CV Y: CV component Y for neural tracking.

(i) *Mean square error (MSE)* for each Cartesian component (X, Y), to quantify the difference between values implied by a filter (neural or classical) and the true values (ideal optimal estimate). For any different [training set (a, b, c), validation set (20 trajectories), and filter (neural, classical)] simulation minimum, maximum, and average values are calculated.

(ii) *Coefficient of variation (CV)*, again for each Cartesian component (X, Y), as a normalized measure of dispersion, independent of the unit in which the measurement has been taken, applied to the difference between estimation error in neural tracking and classical tracking (in each cartesian component). Distributions with CV < 1 will be considered with low variant. For any different [training set (a, b, c) or validation set (20 trajectories)] simulation minimum, maximum, and average values are calculated for neural filter.

Numerical results for these metrics are shown in Table 1 for the different considered training set size cases (validation with 20 trajectories).

In each table, *average*, *maximum* and *minimum, MSE values* are obtained from MSE calculated from the set of 20 estimated curves, comparing each estimated curve with the ideal optimal solution, the real trajectory, without error.

The values shown in Table 2 are obtained as the average results for the 20 validation trajectories considered in each training case.

In Figure 8 it can be observed that MSE for neural tracking decreases slowly as training set size is increased.

In the results introduced in Figure 8 it can be observed that the trajectory estimated by the NN architecture, where

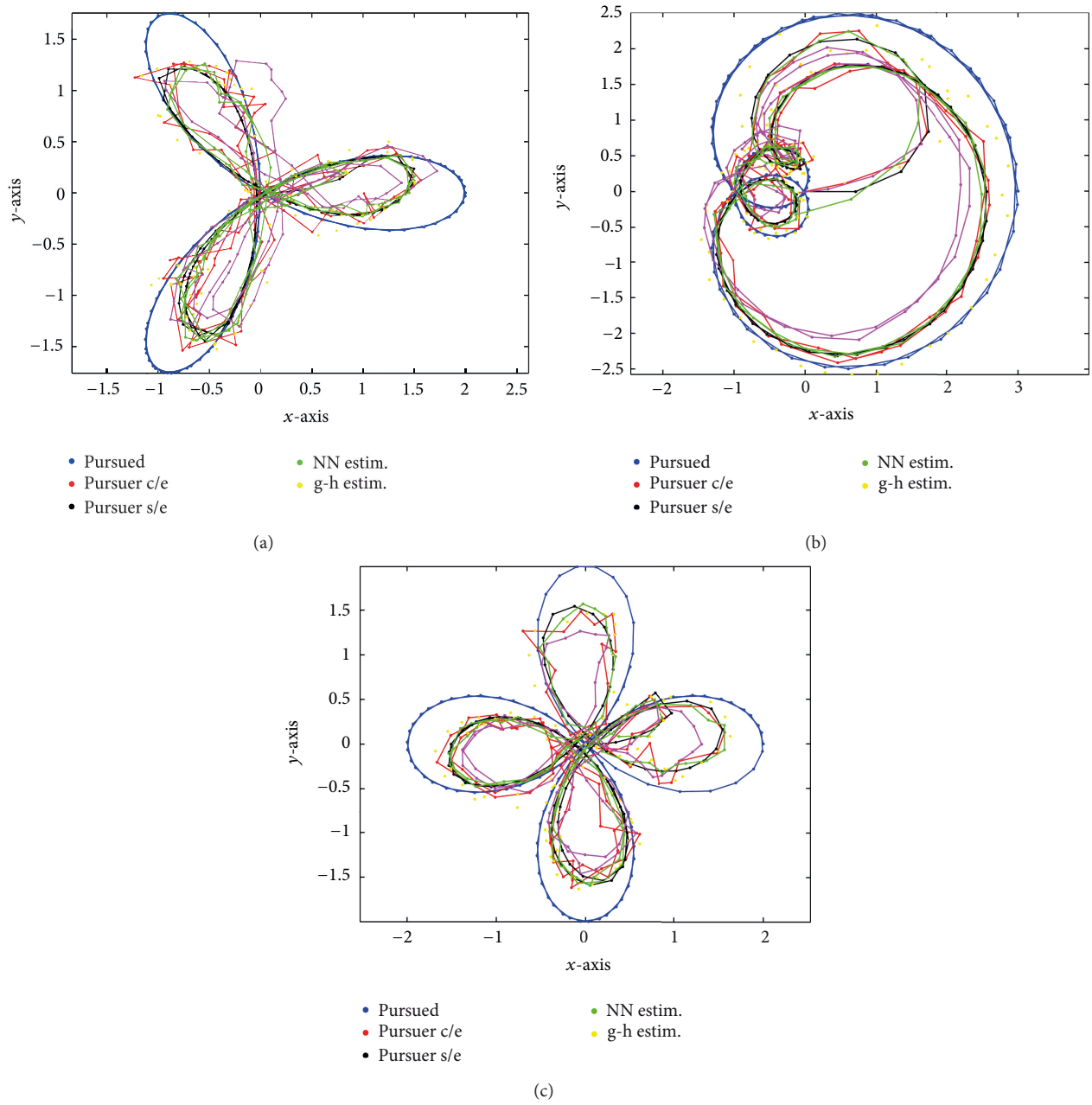

Figure 7: Examples of estimated curves with one trained neural network tracking architecture.

contextual information is considered, shown in Figure 7 like green trajectory, greatly improves the tracking performance obtained with a classical tracking algorithm (α-β, $\alpha = 0.5$) (Figure 9) (yellow trajectory in Figure 7). It is clearly reflected in MSE values obtained.

Checking the CV for neural tracking (Figure 10) it can be said that our neural filter provides estimation with low variance (CV < 1) in both components X and Y, or what it is the same, the error in the provided estimation remains steady. This data can be verified checking *maximum* and *minimum* MSE values shown in Table 1 and comparing these values with the obtained for the classical algorithm. So, clearly the dispersion is improved comparing these data with the data obtained for classical tracking algorithm.

In Figure 10, it can be seen that as the training set size increase the CV both component equal, decreasing the difference in variance between them.

The estimated trajectory fairly accurately fits the trajectory free of error (black trajectory), removing much of the noise that might contain the measurements provided as input data.

The results presented in this paper must be taken as purely experimental results, which have allowed us to study the feasibility of the proposal outlined here, considering the

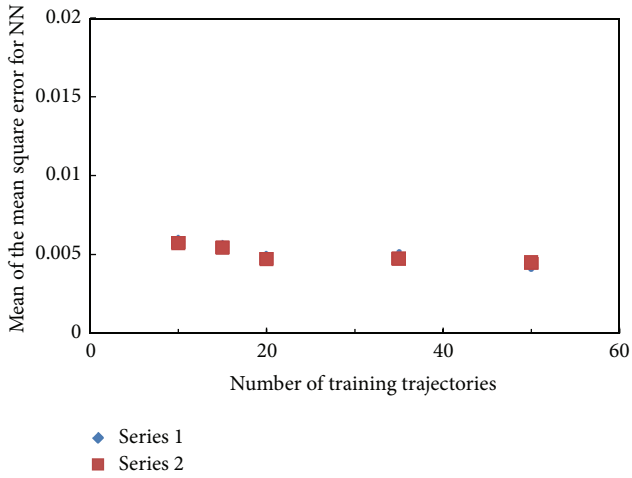

Figure 8: MSE for neural tracking (Series 1 = X, series 2 = Y).

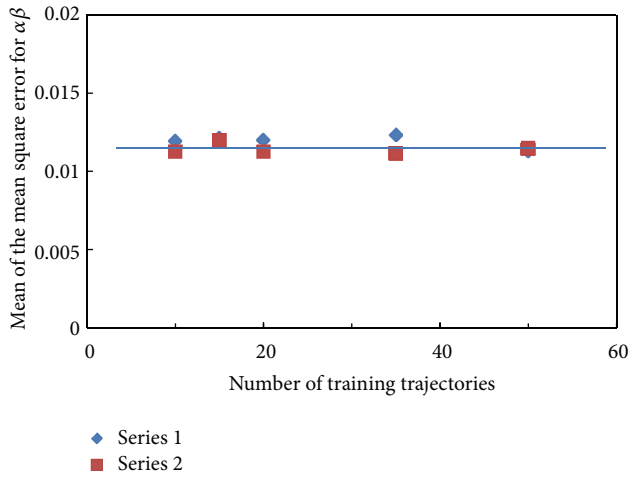

Figure 9: MSE for classical tracking (Series 1 = X, series 2 = Y).

case previously exposed as a concrete case study. For each considered case, a verification and validation scheme for the NN implemented must be developed [35].

4. Conclusions

In this case study the objective is to document a proposal to incorporating contextual information, which can be found in certain aircraft maneuvers, to a typical estimation and tracking process running on any air surveillance radar system.

The aim of incorporate this contextual information is to decrease the uncertainty and imprecision that accompany any radar measurement which is used to feed this kind of systems.

This contextual information must be considered for the maneuvers to be estimated: it is completely different to estimate a fighter-to-fighter combat curve than to estimate a fighter air-to-ground attack. In our case, the study is adapted to a specific fighter-to-fighter combat maneuvering.

The idea of incorporating contextual information to an air target estimation/tracking algorithm in the scope of

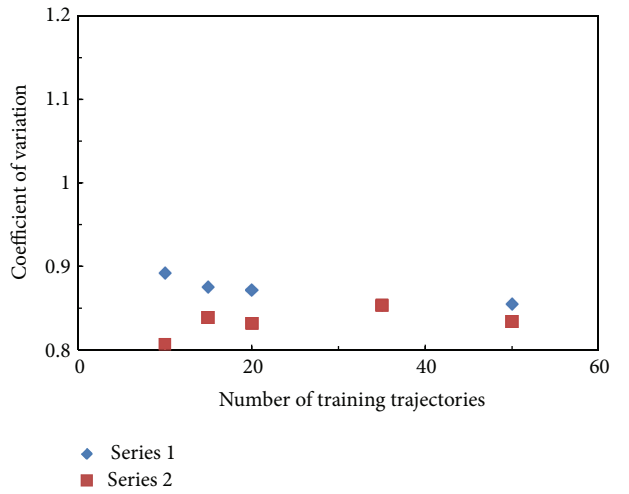

Figure 10: CV for neural tracking (Series 1 = X, series 2 = Y).

radar surveillance, taking into account the own requirements of this real time systems, implementing this on a neural network architecture, which will allow us to ignore the complexity of formulating the rules for the integration of this information with the information directly contained into the radar measurements provided as inputs to the system, allows us to use this solution as an alternative to the traditional tracking algorithms, with a purely statistical approach.

The implemented NN architecture, with a set of NNs, gives the system a more robust behavior against a possible implementation on a single network, where all information used in this estimation process could be incorporated.

The results presented for the simulations are likely to preclude the use of this proposal which would be accepted as a possible tracking algorithm to be used under certain conditions, improving the results obtained in these specific cases.

The dependence of the used NNs to the conditions presented, as well as the need to achieve convergence in the estimation results against the expected results, makes advisable to use a set of NNs, connected with a certain architecture, where the contextual information applied to each case would be dependent on the own curve definition to be estimated.

Acknowledgments

This work was supported in part by Projects MEyC TEC2012-37832-C02-01, MEyC TEC2011-28626-C02-02, and CAM CONTEXTS (S2009/TIC-1485).

References

[1] Y. Bar-Shalom and X. Li, "Multitarget-multisensor tracking: principles and techniques," *IEEE Aerospace and Electronics Systems Magazine*, vol. 11, no. 2, 1996.

[2] R. W. Sitler, "An optimal data association problem in surveillance theory," *IEEE Transactions on Military Electronics*, vol. 8, pp. 125–139, 1964.

[3] A. Farina and F. A. Studer, *Radar Data Processing, Vol. 2. Advanced Topics and Applications*, Electronic and Electrical Engineering Research Studies: Electronic Circuits and Systems Series, Wiley and Research Studies Press, Hertfordshire, UK, 1986.

[4] S. S. Blackman, *Multiple Target Tracking with Radar Applications*, Artech House, Dedhan, Mass, USA, 1986.

[5] Y. Bar-Shalom and T. E. Fortmann, *Tracking and Data Association*, Academic Press, San Diego, Calif, USA, 1988.

[6] P. L. Bogler, "Tracking a maneuvering target using input estimation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 23, no. 3, pp. 298–310, 1987.

[7] X. R. Li and V. P. Jilkov, "Survey of maneuvering target tracking. Part V: multiple-model methods," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 4, pp. 1255–1321, 2005.

[8] A. Farina and F. A. Studer, *Radar Data Processing, Vol. 1. Introduction and Tracking*, Electronic and Electrical Engineering Research Studies: Electronic Circuits and Systems Series, Wiley and Research Studies Press, Hertfordshire, UK, 1985.

[9] E. Brookner, *Tracking and Kalman Filtering Made Easy*, Wiley-Interscience, New York, NY, USA, 1998.

[10] R. G. Brown and Y. C. Hwang, *Introduction to Random Signals and Applied Kalman Filtering*, John Wiley & Sons, New York, NY, USA, 1992.

[11] N. Audsley and A. Burns, "ESPRIT BRA project (3092), predicatably dependable computer systems," in *Real Time System Scheduling*, vol. 2, chapter 2, Department of Computer Science, University of York, North Yorkshire, UK, 1990.

[12] M. K. Sundareshan, "Neural network schemes for data fusion and tracking of maneuvering targets," Performance Report 1 (ONR grant # N00014-95-1-1224)., 1995.

[13] W. Juszkiewicz and A. Stateczny, *GRNN Cascade Neural Filter for Tracked Target Maneuver Estimation*, Neural Networks and Soft Computing, Zakopane, Poland, 2000.

[14] A. K. Sarkar, S. Vathsal, S. Sundaram, and S. Mukhopadhay, "Target acceleration estimation from radar position data using neural network," *Defence Science Journal*, vol. 55, no. 3, pp. 313–328, 2005.

[15] R. L. Shaw, *Fighter Combat. Tactics and Maneuvering*, Naval Institute Press, Annapolis, Md, USA, 1985.

[16] S. S. Blackman, *Multiple Target Tracking with Radar Applications*, Artech House, Norwood, Mass, USA, 1986.

[17] M. Lloyd, *Pursuit Curves*, Academic Forum, Atlanta, Ga, USA, 2007.

[18] N. Audsley, *Survey: Scheduling Hard Real Time Systems*, Department of Computer Science, University of York, North Yorkshire, UK, 1990.

[19] E. Vivancos, L. Hernández, and V. Botti, "Distributed artificial intelligence in real time environments," *Iberoamerican Journal of Artificial Intelligence*, vol. 2, no. 6, 1998.

[20] J. A. Stankovic and K. Ramamritham, "What is predictability for real-time systems?" *Real-Time Systems*, vol. 2, no. 4, pp. 247–254, 1990.

[21] M. R. Ananthasayanam, A. K. Sarkar, and S. Vathsal, "Estimation of rapid target acceleration in real time from position alone measurements of tracking radar," in *Proceedings of National Systems Conference (NSC '03)*, pp. 64–69, IIT Kharagpur, 2003.

[22] F. Amoozegar, "Neural-network-based target tracking state-of-the-art survey," *Optical Engineering*, vol. 37, no. 3, pp. 836–846, 1998.

[23] L. Chin, "Application of neural networks in target tracking data fusion," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 30, no. 1, pp. 281–287, 1994.

[24] F.-B. Duh and C.-T. Lin, "Tracking a maneuvering target using neural fuzzy network," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 34, no. 1, pp. 16–33, 2004.

[25] J. Zhongliang, X. Hong, and Z. Xueqin, "Information fusion and tracking of maneuvering targets with artificial neural network," in *Proceedings of the IEEE International Conference on Neural Networks (ICNN '94)*, pp. 3403–3408, June 1994.

[26] C. Y. Kong, C. M. Hadzer, and M. Y. Mashor, "Radar tracking system using neural networks," *International Journal of the Computer, the Internet and Management*, vol. 6, no. 2, 1998.

[27] T. Teng, A. Tan, Y. Tan, and A. Yeo, "Self-organizing neural networks for learning air combat maneuvers," in *Proceedings of IEEE World Congress on Computational Intelligence*, 2012.

[28] R. Schvaneveldt, A. Benson, T. Goldsmith, and W. Waag, "Neural network models of air combat maneuvering," Tech. Rep. AD-A254 653, Human Resources Directorate, Aircrew Training Research Division, Mesa, Arizona, 1992.

[29] A. M. Sánchez, M. A. Patricio, J. García, and J. M. Molina, "Video tracking improvement using context-based information," in *Proceedings of the10th International Conference on Information Fusion (FUSION '07)*, Quebec, Canada, July 2007.

[30] A. Benavoli, L. Chisci, A. Farina, S. Immediata, L. Timmoneri, and G. Zappa, "Knowledge-based system for multi-target tracking in a littoral environment," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 42, no. 3, pp. 1100–1116, 2006.

[31] A. Gelb and Technical Staff of the Analytical Sciences Corporation, Eds., *Applied Optimal Estimation*, MIT Press, Cambridge, Mass, USA, 1974.

[32] N. Morrison, *Introduction to Sequential Smoothing and Prediction*, McGraw-Hill, New York, NY, USA, 1969.

[33] J. A. Freeman and D. M. Skapura, *Neural Networks Algorithms, Applications and Programming Techniques*, Addison-Wesley, Boston, Mass, USA, 1991.

[34] B. Randell, J. Laprie, H. Kopetz, and B. Littlewood, Eds., *Predictably Dependable Computing Systems*, Springer, Berlin, Germany, 1995.

[35] L. L. Pullum, B. J. Taylor, and M. A. Darrah, *Guidance for the Verification and Validation of Neural Networks*, John Wiley & sons, Hoboken, NJ, USA, 2007.