

Target Components Discrimination using Adaptive Time-Delay Neural Network*

DAW-TUNG LIN

*Department of Computer Science and Information Engineering
Chung Hua University
Hsinchu, 300 Taiwan*

The feasibility of distinguishing multiple type components of exo-atmospheric targets is demonstrated by applying the Time Delay Neural Network (*TDNN*) and the Adaptive Time-Delay Neural Network (*ATNN*). Exo-atmospheric targets are especially difficult to distinguish using currently available techniques because all target parts follow the same spatial trajectory. Thus, classification must be based on light sensors that record signal over time. Results have demonstrated that the trained neural networks are able to successfully identify warheads from other missile parts on a variety of simulated scenarios, including differing angles, fragmented pieces and tumbling. The network with adaptive time delays (the *ATNN*) performs highly complex mapping on a limited set of training data and achieves better generalization to overall trends of situations compared to the *TDNN*. We also apply the theorem of Funahashi, Hornik et al and Stone-Weierstrass to state the general function approximation ability of the *ATNN*. The network is trained on additive noisy data and shows that it possesses robustness to environment variations.

Keywords: automatic target recognition, target trajectories discrimination, time delay neural network, adaptive time-delay neural network, noise resilience

1. INTRODUCTION

Automatic Target Recognition (*ATR*) is a highly challenging problem due to target signature variability, environmental changes, and a limited database [22]. It involves extraction and discrimination of critical information from complex and uncertain data. We report here the application of the Adaptive Time-Delay Neural Network (*ATNN*) to exo-atmospheric target discrimination and reject false target-like trajectories. The scenario of this problem is schematically illustrated in Fig. 1. The components of a vehicle (a threat) separate and will be and detected by a visual sensor after launch. Among these components, we are interested in only the warhead component as the other components are not warheads. The objective is to discriminate between these components regardless of different aspect angles, tumbling or broken pieces and find the target component. Exo-atmospheric targets are especially difficult to distinguish using currently available techniques because all target parts follow the same trajectory during the exo-atmospheric portion of the flight.

Received October 22, 2002; revised February 6, 2003; accepted June 16, 2003.

Communicated by Chung-Yu Wu.

* This work was supported in part by the National Science Council, (Grand NSC-85-2213-E216-079), Taiwan, R.O.C., and the Applied Physics Laboratory of Johns Hopkins University. Acknowledgment also due to Dr. Judith E. Dayhoff for her invaluable discussions and comments.

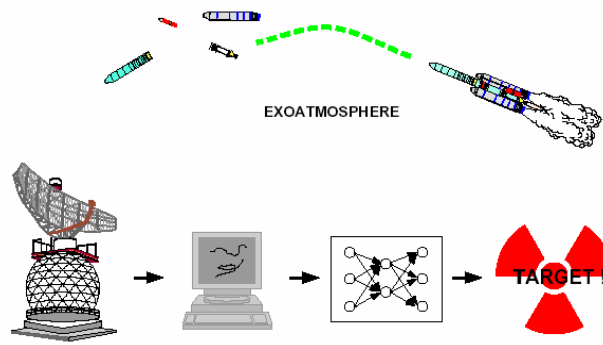


Fig. 1. Scenario of the exo-atmospheric target components discrimination.

The ATR problem involves multidisciplinary issues such as target signature configuration, sensor devices development, software processing technology, and hardware implementation considerations. Generally, methodologies currently being developed includes: statistical approaches, model-based pattern recognition, knowledge-based analysis, adaptive and learning models. However, no single method is likely to be applied as a total solution to ATR field. Morphology theory and gabor wavelet transform have been employed to forward looking infrared images (FLIR) [3]. The emphasis is given to obscured target detection but system performance varies due to the trade off of false alarm setting. To recognize the shape of aircraft in FLIR images, a deformation operator for thresholding to model the types of objects proposed [11]. The Bayesian approach has been utilized to match the target by rigid templates under the condition of infinite poses for FLIR and delay-Doppler radar images [16]. However, the above methods need *a priori* knowledge of combining *a priori* knowledge. Furthermore, the success of approach also depends on whether the *a priori* statements are accurate and thorough. Neural net technology offers a number of tools such as learning and adaptation, generalization and robustness, feature extraction, which could form the basis of a fruitful approach to the ATR problem [1, 11, 17, 19, 25, 26]. Neural net architectures with dynamic and temporal capabilities are more promising for signal analysis. The time-delay neural network (TDNN) proposed by Waibel et al, employs time-delays on connections and has been successfully applied to phoneme recognition [23, 24]. The result is a dynamic learning technique for spatiotemporal classification. The Adaptive Time Delay Neural Network (ATNN), which adapts time delays as well as weights during training, is a more advanced version of the TDNN. We address this problem by using both the TDNN and ATNN, and conclude that the ATNN can make further improvements in target components discrimination.

2. ATNN APPROACH

2.1 Architecture Overview

The Adaptive Time Delay Neural Networks adapts its time delay values as well as its weights during training, to better accommodate to changing temporal patterns, and to

provide more flexibility for optimizing tasks [5, 6, 12, 14]. Simulation results from the ATNN on system modeling, trajectories production, and prediction have been shown to be promising [6, 12, 14].

For illustration, a three layered ATNN is shown in Fig. 2, where n_{ji} denotes the number of time-delays employed on the connections to node j from node i , index h indicates the interconnections layer. Each node j of layer h (e.g., $j \in \mathcal{N}_h$), receives n_{ji} inputs from each $i \in I_{j,h-1}$, where $I_{j,h-1}$ is the subset of nodes on layer $h-1$ ($I_{j,h-1} \subseteq \mathcal{N}_{h-1}$) that connects to unit j on layer h . **Time frame** \mathcal{T}_{ji} is a set of time delays ($\tau_{ji1}, \dots, \tau_{jini}$) employed on the connections to node j from node i . The time frame can have an additional index h ($\mathcal{T}_{ji,h}$) to signify layer h . Node i of layer h is connected to node j of the next layer, with the connection line having an independent and modifiable time-delay $\tau_{jik,h}$ and weight $w_{jik,h}$. Each node sums the net inputs from the activation values of the previous neurons, through the corresponding time-delays on each connection line. Then the output of node j is governed by a symmetric sigmoid function f of the net input. The adaptation of both time delays and weight parameters is derived based on the gradient descent method to minimize the cost function E during training based on error back-propagation [15]. The cost function is defined as the instantaneous mean square error measure of the desired output d_j and network output $a_{j,L}$ at time t_n as $E(t_n) = \frac{1}{2} \sum_{j \in \mathcal{N}_L} (d_j(t_n) - a_{j,L}(t_n))^2$, where L denotes the output layer. The weight and time-delay modifications are

$$\Delta w_{jik,h} \equiv -\eta_1 \frac{\partial E(t_n)}{\partial w_{jik,h}} \quad \Delta \tau_{jik,h} \equiv -\eta_2 \frac{\partial E(t_n)}{\partial \tau_{jik,h}},$$

where η_1 and η_2 are the learning rates. Detailed derivations of this learning algorithm are given in [5, 13].

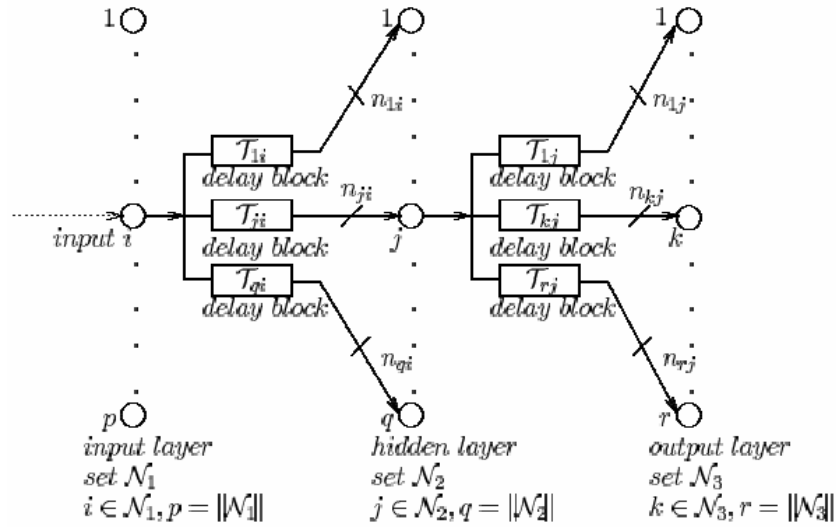


Fig. 2. Three layered ATNN.

2.2 ATNN: A Universal Spatiotemporal Function Approximator

The capability of multi-layer feed-forward networks has been studied theoretically. In previous work, Funahashi, Hornik, Stinchcombe and White have concluded that standard multi-layer feed-forward networks with as few as one hidden layer using arbitrary squashing functions are universal function approximators [10]. Similar theorems can also be seen in [4, 7, 18]. The superpositions of a sigmoidal function in achieving universal approximation is also discussed [2]. In this section, we rephrase this mathematical statement and then extend these results to networks with time-delays.

Let $r \in \mathcal{N}$ and A^r be the set of all affine functions from \mathcal{R}^r to \mathcal{R} : $A(x) = w \cdot x + b$ where w and x are vectors in \mathcal{R}^r , and $b \in \mathcal{R}$ is a scalar. Let $\phi(\cdot)$ be a measurable function from \mathcal{R} to \mathcal{R} and let $\Sigma^r(\phi)$ denote the class of functions $\{f(x) = \sum_{i=1}^q c_i \phi(A_i(x))\}$ from \mathcal{R}^r to \mathcal{R} where $x \in \mathcal{R}^r$, $c_i \in \mathcal{R}$, $A_i \in A^r$, $q \in \mathcal{N}$. We adopt Funahashi's main results as below:

Theorem 1 (adopted and restated from Funahashi [7]) Let $\{x_1, \dots, x_n\}$ be a set of distinct points of compact set K in Euclidean space \mathcal{R}^n and let $f: \mathcal{R}^n \rightarrow \mathcal{R}$ be an arbitrary real valued function on K . If ϕ is a bounded and monotone increasing differentiable function, then for any $\varepsilon > 0$, there exists an integer N and real constants c_i , θ_i , and $w_{i,j}$ ($i \in \{1, \dots, N\}$, $j \in \{1, \dots, n\}$) such that $\hat{f}(x_1, \dots, x_n) = \sum_{i=1}^N c_i \phi(\sum_{j=1}^n w_{i,j} x_j - \theta_i)$ of $\Sigma^N(\phi)$ satisfies $\forall x \in K$, $|f(x) - \hat{f}(x)| < \varepsilon$. That is, with N hidden units, a three layered network of class $\Sigma(\phi)$ can approximate any function to a desired accuracy.

Proofs of Theorem 1 are given in Funahashi [7] and Hornik, Stinchcombe and White [10].

Similar theorems can be applied to the TDNN and ATNN to approximate a spatio-temporal function with the extra degrees of freedom in the time-delay domain, considering the network is unfolded. Because the Stone-Weierstrass theorem played a central role in the proof of Theorem 1, we state it here for reference and later use. First we give a few necessary definitions before we introduce the Stone-Weierstrass theorem.

Definition 1 A family \mathcal{A} of real functions defined on a compact set K is an algebra if \mathcal{A} is closed under addition, multiplication and scalar multiplication.

Definition 2 A set \mathcal{A} separates points on compact set K if for every x, y , $x \neq y$ in K , there exists a function f in \mathcal{A} such that $f(x) \neq f(y)$.

Definition 3 A set \mathcal{A} vanishes at no point of K if for each x in K there exists f in \mathcal{A} such that $f(x) \neq 0$.

Theorem 2 (Stone-Weierstrass [10]) Let \mathcal{A} be an algebra of real continuous functions on a compact set K . If \mathcal{A} separates points on K and if \mathcal{A} vanishes at no point of K , then \mathcal{A} consists of all real continuous functions on K .

Let $x_1[t]$, $x_2[t]$, ..., $x_n[t]$ be n input signal channels measured/observed in time interval $[t_0, T]$ in a discrete manner. For fixed $\tau_{j,k} \geq 0$, $j, k \in \mathcal{N}$ and $t - \tau_{j,k}$ in the interval $[t_0, T]$,

we let $\{\langle x_1(t - \tau_{1,1}), \dots, x_1(t - \tau_{1,d}) \rangle, \dots, \langle x_n(t - \tau_{n,1}), \dots, x_n(t - \tau_{n,d}) \rangle\}$ be a set of distinct points of compact set K^d in Euclidean space \mathcal{R}^{nd} . We extend Theorem 1 and obtain the following corollary:

Corollary 1 Let $f: \mathcal{R}^{nd} \rightarrow \mathcal{R}$ be an arbitrary real valued function on K^d . If ϕ is a bounded and monotone increasing differentiable function, then for any $\varepsilon > 0$, there exists integer N , and real constants c_i , θ_i , and $w_{i,j,k}$ ($i \in \{1, \dots, N\}$, $j \in \{1, \dots, n\}$), such that

$$\hat{f}(x) = \sum_{i=1}^N c_i \phi \left(\sum_{j=1}^n \sum_{k=1}^d w_{i,j,k} x_j(t - \tau_{i,j,k}) - \theta_i \right)$$

satisfies $|f(x) - \hat{f}(x)| < \varepsilon, \forall x \in K^d$. In other words, with at least one hidden layer of N hidden units and d time-delays elements in each input-hidden connections pair, a three layered *TDNN* network (with delays employed on the first layer) can approximate any spatiotemporal function to a desired accuracy.

Proof: We apply the Stone-Weierstrass Theorem and use the notations above. Given $\phi(\cdot): \mathcal{R}^{nd} \rightarrow \mathcal{R}$ and let \mathcal{A} is the set of all affine functions from \mathcal{R}^{nd} to \mathcal{R} such that $\mathcal{A}(x) = w \cdot x + b$, $x \in K^d$, i.e. $x = \{\langle x_1(t - \tau_{1,1}), \dots, x_1(t - \tau_{1,d}) \rangle, \dots, \langle x_n(t - \tau_{n,1}), \dots, x_n(t - \tau_{n,d}) \rangle\}$. We denote $\Sigma^{nd}(\phi)$ as the class of functions $\hat{h}(x): \{\sum c_i \phi(\mathcal{A}(x))\}$. We need to show $\Sigma^{nd}(\phi)$ consists of all real functions on K^d . First we show $\phi(\mathcal{A})$ is separating on K^d that ensures $\Sigma^{nd}(\phi)$ is separating on K^d . Let $K^d \subset \mathcal{R}^{nd}$ be a compact set, so for ϕ from \mathcal{R}^{nd} to \mathcal{R} , $\Sigma(\phi)$ is an algebra on K^d . Pick $a, b \in \mathcal{R}$, $a \neq b$, such that $\phi(a) \neq \phi(b)$. Pick $\mathcal{A}(\cdot)$ such that $\mathcal{A}(x) = a$ and $\mathcal{A}(y) = b$, $x, y \in K^d$, then we obtain $\phi(\mathcal{A}(x)) \neq \phi(\mathcal{A}(y))$. Therefore ϕ is separating on K^d .

Secondly, we need to show $\phi(\mathcal{A})$ vanishes at no point on K^d . Pick $b \in \mathcal{R}$ such that $\phi(b) \neq 0$ and set $\mathcal{A}(x) = 0 \cdot x + b$. For all $x \in K^d$, $\phi(\mathcal{A}(x)) = \phi(b)$, the result follows. \square

The proof is mathematically equivalent to that of Hornik, Stinchcombe and White's theorem of universal approximation with multi-layer feed-forward networks. We extend from n dimensional input space to nd dimensional space. In other words, each distinct input point x_i in n dimensional input space is expanded into a distinct points set $\langle x_i(t - \tau_1), \dots, x_i(t - \tau_d) \rangle$ in nd dimensional input space.

3. AUTONOMOUS EXO-ATMOSPHERIC TARGET DISCRIMINATION

A real world application with *ATNN* is discussed in this section for exo-atmospheric target discrimination. As illustrated in Fig. 1, after the components of a launched vehicle separate, they will be detected by a visual sensor. The sensor records intensity and size over time for each component. Among these components, we are only interested in the target component; the other pieces can be classified as non-target components. The objective is to discriminate between these components while disregarding different aspect angles, tumbling or broken pieces and find the target component.

This is a very important but difficult task in target component discrimination, as stated in *APL* technical report by Resch [20], "The ability of a kinetic kill vehicle (KKV) to discriminate between warhead and booster parts or decoys is critical to theater ballistic

target defense (TBMD). Exo-atmospheric targets are especially difficult to distinguish using currently available techniques because all target parts follow the same trajectory during the exo-atmospheric portion of the flight.”

The sensors used in this analysis have dedicated specifications. The maximum radiance and size versus time are obtained from these sensors for all four components (denoted in this context as warhead, oxidizer tank, fuel tank, and fins, in which the warhead is the target we are interested in). Furthermore, these components may tumble with different aspect angles or break into several pieces. Examples are described in the Appendix (section 6) Table 7. One of the many things that makes *ATR* so hard is that the same target can vary wildly in appearance depending aspect angles, atmospheric effects, and other variables.

3.1 TDNN Technique and Simulation Setups

A three layered *TDNN* is used to perform the discrimination task. This network contains an input layer with six inputs, one hidden layer with three hidden units, and two output nodes to indicate true or false target. Three of the inputs are the ratios of a piece’s highest radiance to the highest radiance of the other pieces. The other three inputs are the ratios of a piece’s size to that of the other pieces. Data is input to the network sequentially starting at a certain number of seconds before intercept and ending very shortly before intercept. The target value is one or zero representing either true or false targets for each particular training set composed of the ratios described above. The number of time-delays is selected as 4 and 6 on the first and second layer of connections, respectively. Time-delays on each pair of connections between neurons are fixed and are consecutively and equally spaced as $0, \tau, 2\tau, \dots$, where τ is the data sampling interval. The layout of the TDNN network is depicted in Fig. 3.

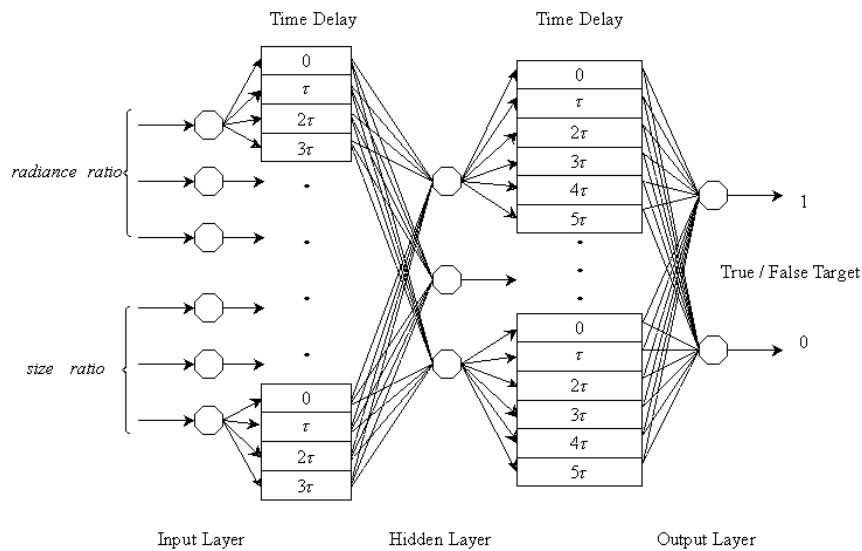


Fig. 3. TDNN used in the target components discrimination simulation.

A symmetric sigmoid function is employed on all processing units. This appears to be advantageous for convergence and recognition performance. The data is scaled and offset so as to be symmetric about 0 with range $[-0.5 : 0.5]$. Thirty-six sets of object data are employed in the simulation and constitute one training set and five test sets with various conditions of trajectories, including different aspect angles for different components, broken pieces and different time sampling. The data characteristics are described in detailed in Table 7 in Appendix (section 6). As we can see in Table 7, the training set contains 9 sets of time series trajectories, seven (objects #1 to #7) with the same aspect angles, the rest (#8 and #9) with different rumbling aspect angles, and one object (#7) with broken components. Test set #1, #2 and #4 include trajectories of components with various aspect angles intercepted at high or low altitudes, in which set #2 has smaller warhead aspect angles. Test set #3 contains possible conditions of broken pieces. Nine sets of trajectories are included in test set #5, all with the same aspect angles, but new angles are tested (45° and 75°) which are not shown in the training set.

3.2 Simulation Scheduling and Analysis

For every vehicle object, four series of n time-step data (wh, ot, ft, fins) are received simultaneously and fed into the network. The Neural network is trained to discriminate a warhead as a true target from the other pieces of components. For example, the training set includes 36 time-series data for 9 objects. The numbers of time-series data of test sets #1 to #5 are 20, 20, 16, 16, and 36, respectively. Each time-series data contains n time-step signals in terms of radiance ratio and size ratio as illustrated in section 3.1. The overall system performance of one data set is calculated as the correct identification of four time-series corresponding to whether a component is a true target or non-target for each object:

$$\text{overall identification rate} = \frac{\text{total \# of correct identifications for all components}}{4n \times \# \text{ of objects in a test (training) set}}$$

Two training schedules are used and tested on five different test sets. These schedules are described as below:

Schedule 1: This training schedule begins adapting weights after the first segment of data enters the network [20]. However, we do not allow data from a previous run to remain in the network at the same time that data from a new run enters the network.

Schedule 2: A new training schedule is used in which data fills the network before weight adaptation begins. Performance improved with this training schedule as shown in Table 1. Fewer iterations of training were required (only 5500 as compared with 14,300 shown in Table 2).

The learning rate parameters η_1 and η_2 of weight and time-delay adaptation for the above two schedules are arranged according to simulated annealing strategy [8] to enforce convergence as following:

Table 1. Successful identification rate of different test sets by using TDNN trained on 0.5 second increment data.

data set	Schedule 2	Schedule 1
training set	100%	89.61%
test set#1	100%	89.14%
test set#2	59.4%	53.18%
test set#3	59.4%	50.00%
test set#4	93.4%	81.82%
test set#5	97.9%	96.59%

Table 2. Over all performance of TDNN trained on 0.5 second increment data.

Overall	Schedule 2	Schedule 1
iterations	5565	14323
RMSE	0.044	0.105

$$\begin{cases} \eta_1 = 0.9, \eta_2 = 1 & \text{if training iteration} < 40000 \\ \eta_1 = 0.7, \eta_2 = 0.1 & \text{if } 40000 < \text{training iteration} < 80000 \\ \eta_1 = 0.5, \eta_2 = 0.01 & \text{otherwise} \end{cases}$$

We also used various numbers of hidden units to see if performance would improve. The TDNN was trained to perform target discrimination with the new scaling procedure (Schedule 2). Table 3 summarizes its performance for three hidden units ($3h'$) and five hidden units ($5h'$), as compared to our previous results labeled $3h$ (with three hidden units). Performance was better for the third, fourth, and fifth data sets.

Table 3. Comparison of discrimination performance with different data sets, various training methods and different network topologies.

data set	$3h'$	$5h'$	$3h$	$3h''$
training set	89.61%	89.61%	89.61%	85.61%
#1	89.14%	88.89%	89.14%	88.13%
#2	66.36%	65.00%	53.18%	64.09%
#3	72.72%	70.45%	50.00%	72.73%
#4	82.72%	79.55%	81.82%	83.64%
#5	96.59%	94.89%	96.59%	89.77%

Another experiment was performed to include additional data in the training set, with different aspect angles for different components. Previous training experiments were limited to data with the same aspect angle for different components. Performance improved on the third, fourth, and fifth data sets compared to the benchmark $3h$ run, but

only the fifth data set showed an improvement over other runs with the revised scaling procedure. Performance for this experiment is labeled $3h''$ in Table 3. As a whole, the network figured out the target in all cases. When we observe time-step by time-step for each set of trajectories (e.g. Figs. 4 and 5) and the overall performance (test set #2 66.36% compared to that of sets #1 and #4), the results show that the smaller the aspect angle of the warhead, the more difficulty the network had in distinguishing the warhead as the target.

Now, we consider different time-sampling data: every half second and every tenth second. When using tenth second data, the time delays in first layer were (0.0, 0.1, 0.2, 0.3) seconds, and with half second data they were (0.0, 0.5, 1.0, 1.5). We obtained 100 % performance on the faster sampling data. Identification occurred much faster than on the previous data set with half second timing. In all cases, with the new tenth second sample data, the activation rapidly becomes high for the target and rapidly low for the non-target runs.

For comparison, Figs. 4 to 11 show some of the simulation results on the same scenarios, but with two different sampling intervals. On the left side are the results from the 0.1 second sampling interval (Figs. 4, 6, 8, 10). On the right side are the results of the 0.5 second interval data (Figs. 5, 7, 9, 11).

In these figures, red lines represent the network output for war head components; green, blue and brown lines denotes the output of networks for oxidizer tank, fuel tank and fins, respectively. The Y axis indicates the network output values where 1 denotes a true target and 0 denotes a negative target. As we can observe from Figs. 4, 6, 8 and 10, the simulation results show quick and correct identification of the war head, (plotted in red line) which maintains a high output and while the outputs for the remaining components quickly go to zero. Several of the scenarios with half second interval data have some confusion in the identity of the target part (red line) and oxidizer tank (green line); examples can be seen in Figs. 5, 7 and 9. One result (Fig. 11) even shows a wrong response war head component where the red line is always at the bottom. But, with the 0.1 second data, this confusion does not occur. These cases are shown by comparing Figs. 6 to 7, Figs. 8 to 9, and Figs. 10 to 11. We conclude that in the application of the time-delay neural network, time sampling of every tenth second significantly improves performance in terms of speed of identification and percentage of correct recognition. In the above simulations, time sampling rates are restricted to 0.1 and 0.5 second due to special purpose and limitation of sensor devices. We have analyzed the effects of sampling rates on the training speed and signal generation and identification during production of trajectories of a chaotic series on the proposed network architecture [6]. Generally speaking, higher sampling rate and higher resolution will have potential advantages and should result in better preference. However, the selection of an optimal sampling time interval depends on various circumstances, such as cost consideration, device precision constraint, etc.. In this research, one of the objectives is to evaluate whether the proposed system can achieve better identification performance at a tenth second sampling intervals. Another goal is to evaluate the time complexity of the algorithm to meet the requirements of real-time recognition and early identification. The simulation results of the proposed system are very promising. It is also interesting to evaluate whether the performance is further improved when higher sampling rate data are available in the future.

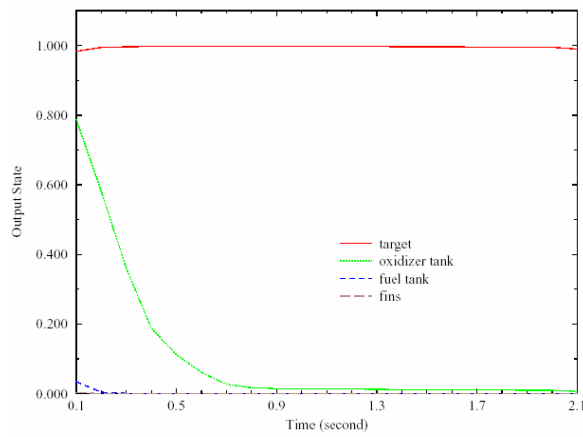


Fig. 4. Recognizing target with aspect angle 30°, sampling time interval 0.1 sec.

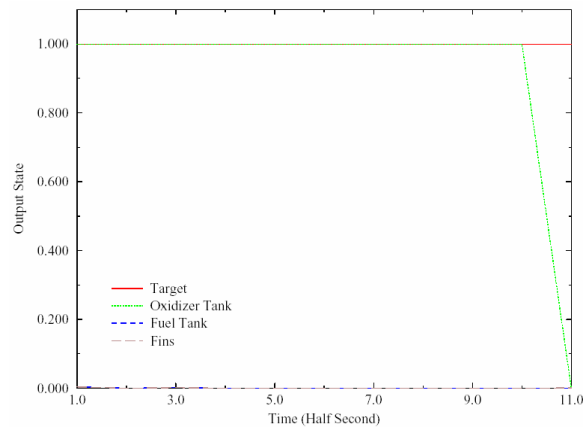


Fig. 5. Recognizing target with aspect angle 30°, sampling time interval 0.5 sec.

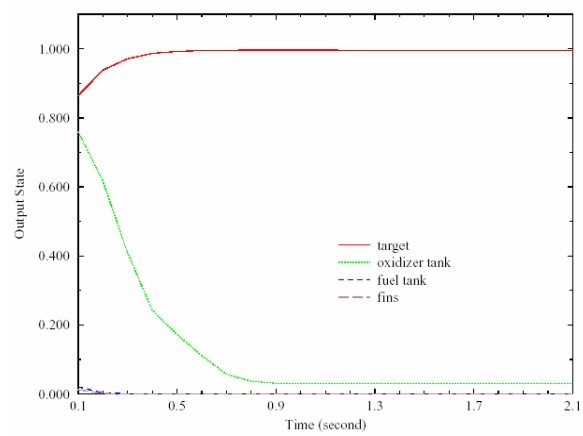


Fig. 6. Recognizing target with aspect angle 60°, sampling time interval 0.1 sec.

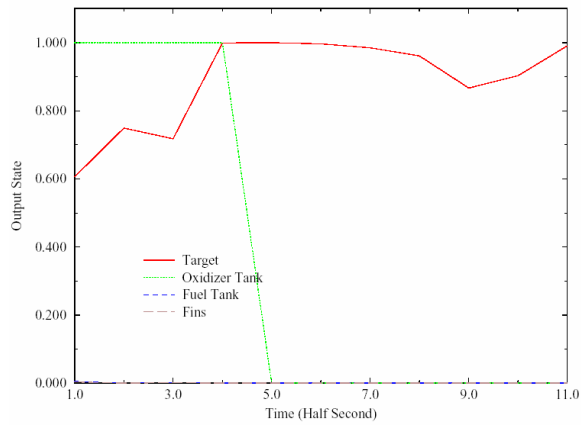


Fig. 7. Recognizing target with aspect angle 60°, sampling time interval 0.5 sec.

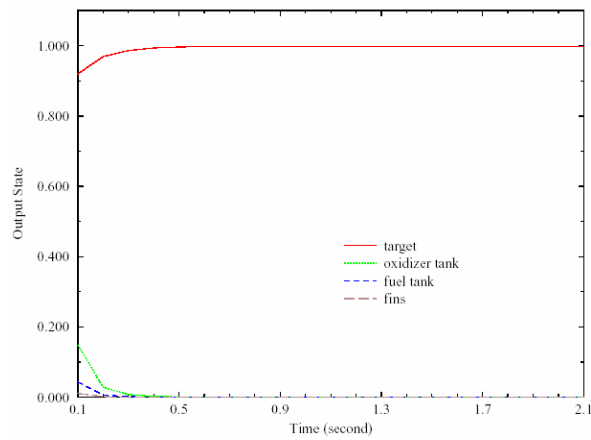


Fig. 8. Recognizing performance: tumbling aspect angles wh 90°, ot 60°, ft 60°, and fins 30°, sampling time interval 0.1 sec.

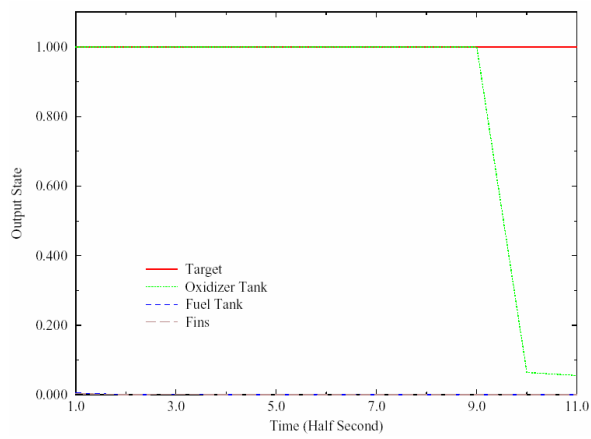


Fig. 9. Recognizing performance: tumbling aspect angles wh 90°, ot 60°, ft 60°, and fins 30°, sampling time interval 0.5 sec.

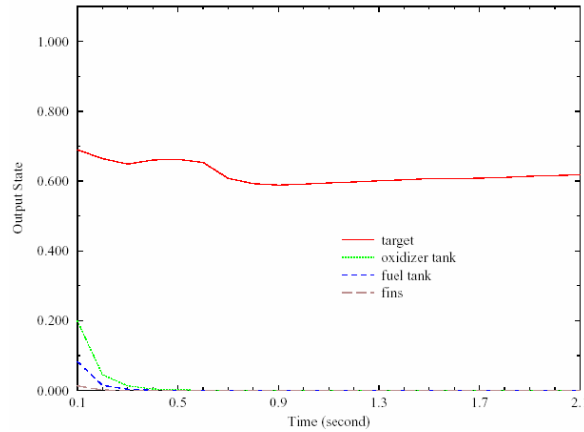


Fig. 10. Recognizing performance: tumbling aspect angles wh 30° , ot 60° , ft 90° , and fins 30° , sampling time interval 0.1 sec.

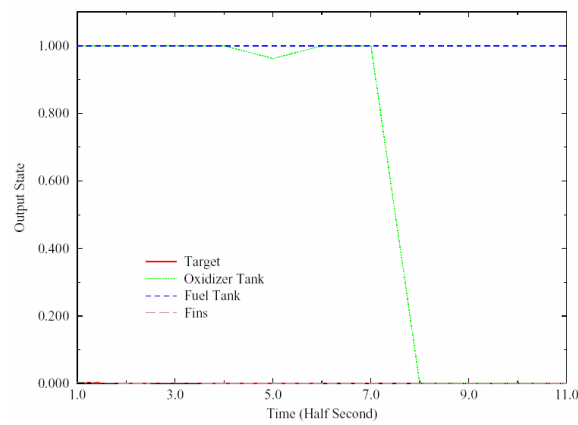


Fig. 11. Recognizing performance: tumbling aspect angles wh 30° , ot 60° , ft 90° , and fins 30° , sampling time interval 0.5 sec.

4. ATNN: A SUPERIOR APPROACH

4.1 Earlier Discrimination on More Realistic Sensor Data

In section 3.2, we have shown that data sampled at tenth second intervals gave superior performance compared to data sampled at half second intervals with the *TDNN*. We applied the *ATNN* to similar data, sampled at tenth second intervals. Furthermore, we needed a longer data scenario to allow more possibilities for time-delays within the network. We applied both the *ATNN* and *TDNN* to this long data. The performance of the *ATNN* was superior to that of the *TDNN*. The reason for this was that the *ATNN* has the ability to adapt time-delays in addition to weights, whereas the *TDNN* adapts only

weights (and leaves time-delays fixed). Fig. 12 shows the topology of the ATNN we employed in the simulation. Compared to the schematic diagram of TDNN in Fig. 3, the time-delays have been replaced by adaptable variables $\tau_1, \tau_2, \dots, \tau_n$. As a result, performance overall was very promising.

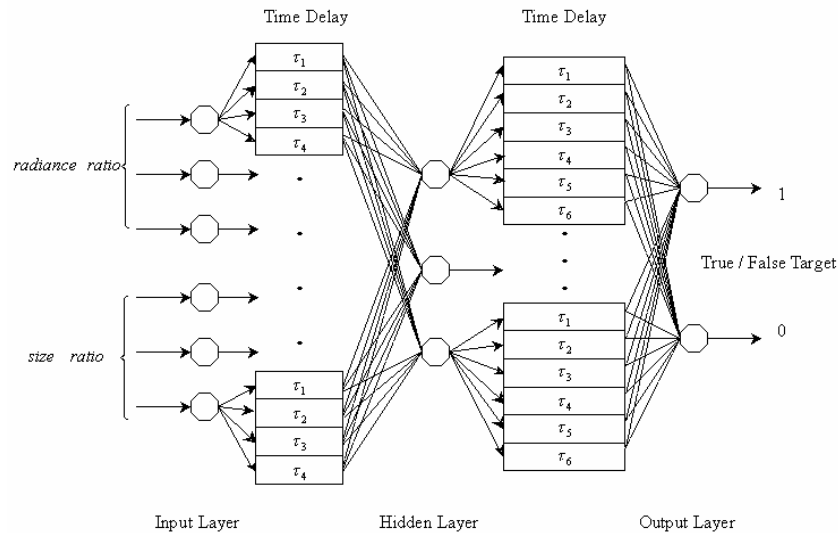


Fig. 12. ATNN used in the target components discrimination simulation.

The data consisted of three target break-up scenarios, taken from 30° , 60° , and 90° aspect angles. The new data was for the case where the piece can move around on the sensor, so the resulting measurements are noisy. The piece can either be located on one pixel or located over two pixels (half of the piece lies in one pixel and half lies in an adjacent pixel). This is more realistic than assuming that the sensor can lock the piece onto one pixel at all times [21]. Thus, the data the neural network is trained and tested on jumps up and down randomly from one time step to the next. This is more realistic than the smooth data we worked with previously. The noise made the recognition problem more challenging for the network.

The *TDNN* was initially used on this bumpy data and the result was not as good as before (with the smooth data). The *TDNN* results for the new data are shown in Figs. 14, 16, and 18. *TDNN* is not capable of resolving two pixel data with slopes changes. As we can see in Figs. 14 and 16, the output of the war head component plotted in red lines are very unstable.

The *ATNN* was then trained on the new data and tested. The network provides more flexibility as time-delay elements are adapted according to the characteristics of the data. The *ATNN* achieved 100% correct recognition for all three aspect angles. When the networks were given the entire time to recognize the parts, and the recognition was performed at the end of the data stream, then the *ATNN* performed with 100% correct recognition, whereas the *TDNN* had only 33% recognition at end of the data stream.

It is important to consider a situation where the network is required to identify the parts before the end of the data stream. Thus we computed a stepwise performance percentage, which consists of the percent correct recognition averaged over all possible time steps in the data. On this basis, the *ATNN* achieved a 99.46% correct recognition rate. This means that if the *ATNN* were requested to make a decision at any time during the incoming data stream, it would have been correct 99% of the time. Early identification is made possible with *ATNN*. The *ATNN* performance was far superior to that of the *TDNN*, and was far more robust to noise in the data. For comparison, the *ATNN* results (Figs. 13, 15, and 17) are shown to the left of *TDNN*'s results (Figs. 14, 16, and 18) on the same observation aspect angles. A summary is shown in Table 4.

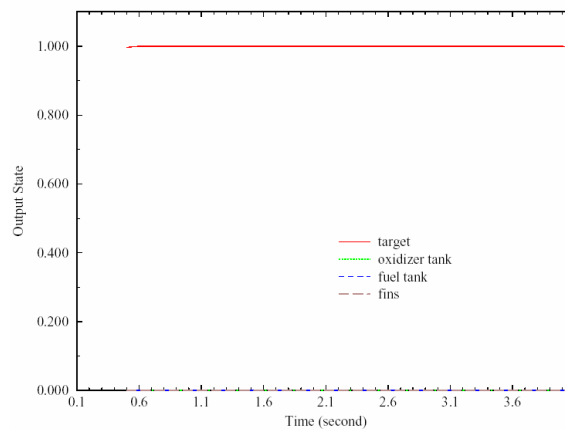


Fig. 13. ATNN performance on bumpy data with aspect angle 90° , sampling time interval 0.1 sec.

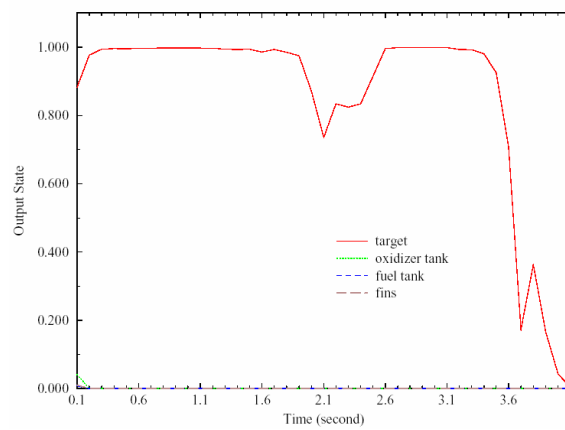


Fig. 14. TDNN performance on bumpy data with aspect angle 90° , sampling time interval 0.1 sec.

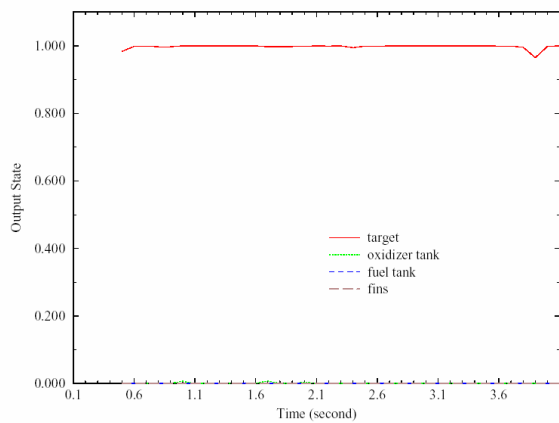


Fig. 15. ATNN performance on bumpy data with aspect angle 60°, sampling time interval 0.1 sec.

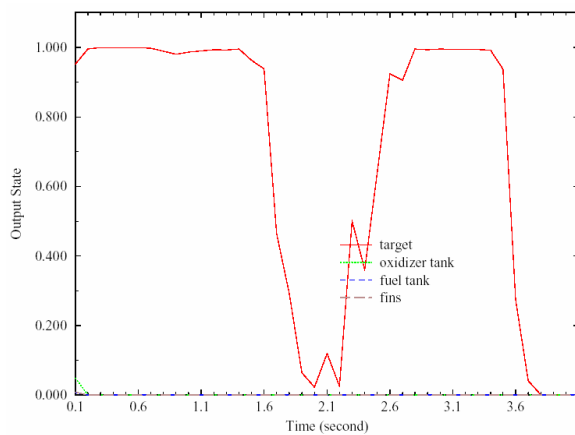


Fig. 16. TDNN performance on bumpy data with aspect angle 60°, sampling time interval 0.1 sec.

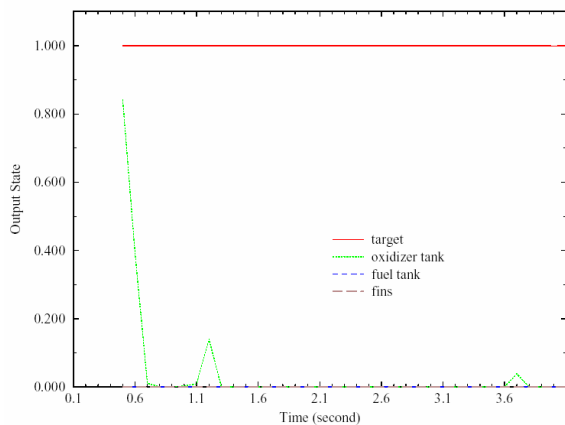


Fig. 17. ATNN performance on bumpy data with aspect angle 30°, sampling time interval 0.1 sec.

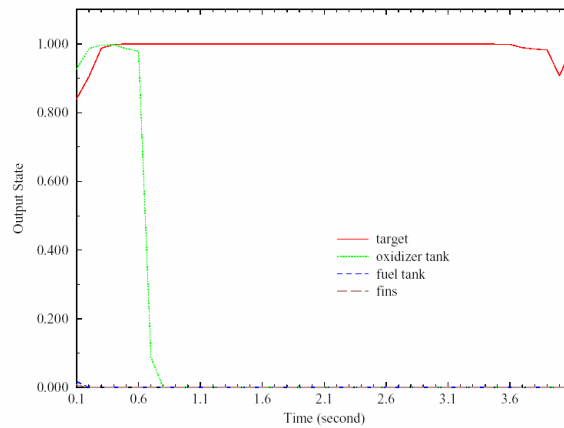


Fig. 18. TDNN performance on bumpy data with aspect angle 30° , sampling time interval 0.1 sec.

Table 4. Comparison of end-point recognition and each time-step performance of *ATNN* vs. *TDNN* with more realistic and bumpy data.

	ATNN	TDNN
End Point Performance	100%	33.33%
Time Stepwise Performance	99.46%	80%

4.2 Environment Variations and Robustness

One of the events which degrades the performance of automatic target recognition is environmental change. Sources of changes and noise include blurred images, camera vibration, heat, atmospheric effects, etc. The effects of noise and how to perform recognition in spite of noise will be very important for eventual deployment of this technology. Current target recognition systems are unable to modify their behavior based on the dynamic environmental changes occurring around them. In order to achieve robustness, the system must be able to adapt its representation of this dynamic environment while maintaining acceptable performance [22]. We evaluated our neural network's performance in the presence of noise, and improved the neural network robustness by training on noisy data.

To study the effects of noise on performance, we used different scenarios in which noise was added during training or recall or both. Simulation results show that the *ATNN* is robust to moderate amounts of noise, and that its robustness improves when training is done with noisy data. This approach is important for eventual implementation in a real world environment where substantial noise is expected. The data set is based on a simulation of a more realistic scenario than previous data sets, in which the scenario includes tenth second samples with some noise added during simulation by the target simulator. These initial results showed that the *ATNN* recognized the target parts in spite of noise during the simulation. We are concerned, however, that the real-world environment will have additional noise; the initial runs did not consider the effects of such additional noise. We address these effects here.

Our evaluation of noisy scenarios first required simulated data consisting of a simulated target scenario. We chose to start with noisy simulated data and to add varying amounts of additional noise. Sources of noise include (1) the simulator or (2) addition of extra noise to simulated data.

Two different types of training scenario were tested:

1. the *ATNN* was trained with data directly from the target simulator, and
2. the *ATNN* was trained with noisy data consisting of simulation data with extra noise added.

The amount of additive noise was varied in each scenario. The simulated noisy data is transformed as: s_n : size ratio + $k \times 0.01 \times n_r$, r_n : radiance ratio + $k \times 0.001 \times n_r$, where k varies from 1 to 10, and n_r denotes a normal distribution random number with zero mean ($\mu = 0$), and variance (σ) is equal to 1, s_n is an instance of size data with additional noise added, and r_n is an instance of radiance data with extra noise added.

First, the network was trained on simulated data and tested on data with additional noise added. Table 5 shows the results. For each value of k (e.g., each level of additive noise), ten runs were performed. For each value of k , the ten runs are shown in a column in Table 5. In general, the performance degraded as the amount (k) of noise increased. The data is plotted in Fig. 19 with the solid line trace.

Fig. 19 shows graceful degradation, as performance is quite high (98-100%) until $k > 2$.

Next the network was trained with data that included additional noise. This data set contained the original simulated data in addition to the data with additive noise. The variance values of the additive noise were 0.02 and 0.002 for size and radiance, respectively.

Table 5. Performance of *ATNN* trained on pure data and tested with varying amounts of noise added.

runs	k										
	0	1	2	3	4	5	6	7	8	9	10
#1	99.77	100.00	98.64	87.16	75.00	25.00	31.30	25.00	75.00	25.00	25.22
#2	99.77	99.09	95.27	25.00	25.00	75.00	75.00	25.22	75.00	75.22	74.77
#3	99.77	99.77	99.09	91.21	75.00	25.00	75.00	25.00	74.77	25.00	74.32
#4	99.77	99.32	97.52	96.39	76.12	75.00	75.00	25.00	74.77	75.00	36.93
#5	99.77	100.00	99.54	89.86	75.00	75.00	75.00	25.45	80.63	75.00	25.00
#6	99.77	99.32	96.62	85.36	25.00	75.00	84.00	75.00	25.22	75.00	25.00
#7	99.77	99.54	96.62	80.85	83.78	75.00	75.00	75.00	25.22	25.00	75.00
#8	99.77	99.77	98.42	97.74	75.00	68.69	25.00	25.00	75.00	25.00	25.00
#9	99.77	100.00	98.87	92.34	25.00	75.00	25.00	75.00	75.00	26.57	75.00
#10	99.77	99.54	100.00	75.22	75.00	25.00	75.00	25.00	75.00	75.00	76.57
average	99.77	99.63	98.06	82.11	60.99	59.36	61.53	40.06	65.56	50.18	51.28

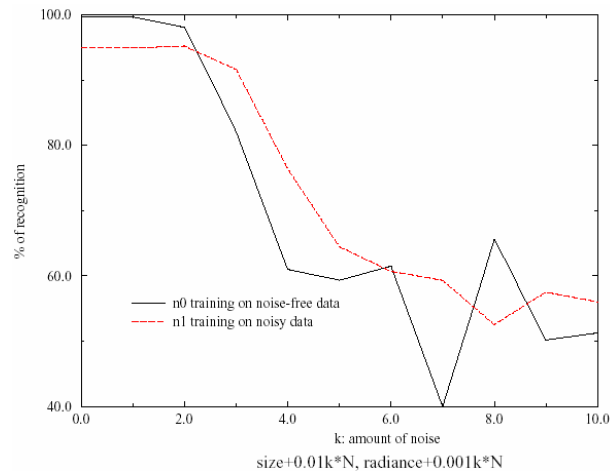


Fig. 19. Performance of the *ATNN* as a function of the amount of noise in the data. For noisy environments, performance is superior when training is done on noisy data.

Table 6. Performance of *ATNN* trained on pure and noisy data and tested on varying amounts of noise added.

runs	k										
	0	1	2	3	4	5	6	7	8	9	10
#1	94.94	93.18	96.71	96.21	87.62	72.47	77.02	73.48	30.30	69.44	66.66
#2	94.94	94.69	92.17	79.79	72.22	54.79	30.51	31.31	50.25	70.95	26.76
#3	94.94	95.95	95.95	96.21	64.39	72.72	29.79	73.48	72.22	77.02	59.34
#4	94.94	95.45	95.45	94.19	87.12	66.66	78.28	28.28	71.46	30.80	66.16
#5	94.94	96.96	96.96	91.41	80.30	83.83	77.77	75.25	73.48	39.14	61.61
#6	94.94	94.19	94.69	90.65	86.86	32.82	80.80	71.46	65.90	31.56	68.93
#7	94.94	94.44	93.18	92.17	90.40	41.91	57.82	76.01	72.72	69.19	28.53
#8	94.94	94.44	95.45	95.20	28.53	84.09	73.98	71.21	27.27	67.42	70.20
#9	94.94	95.45	95.20	94.69	83.58	63.38	71.21	26.26	30.30	65.65	35.85
#10	94.94	94.44	96.46	85.60	83.33	71.96	30.30	66.41	31.56	53.78	76.01
average	94.94	94.92	95.22	91.61	76.43	64.46	60.70	59.31	52.55	57.50	56.01

The trained *ATNN* was again tested on varying amounts of noisy data as described above. The results of ten runs are shown in Table 6. The average performance is provided in the last row. Compared with the average performance in Table 5 (the last row), network trained with additive noise improved the identification capability. The comparison is shown in Fig. 19.

For low noise ($k \leq 2$), there was slightly lower performance when the network was trained on noisy data. For higher noise ($2 < k < 8$), better performance was attained when the network was trained with additive noise. For even larger amounts of noise ($k \geq 8$), there was less jitter in performance when the network was trained with additive noise – more stable performance. We can conclude that the *ATNN* is more robust in noise situa-

tions when it is trained with additive noise. Similar studies and conclusions can be found in [9]. Generalization is one of the characteristics of neural networks. Noise removal capability of ATNN has been reported in our previous paper [12]. An important property of neural nets is noise tolerance. In ATNN, features can be captured in multiple time delays to resolve the embedding dynamics. The proposed network is not affected too seriously by additional noise.

5. CONCLUSIONS

In this paper, we have applied the Adaptive Time-Delay Neural Network to discriminate between similar time varying signals on the same exo-atmospheric trajectory and recognize the target. One of the many things that makes *ATR* so difficult is that the same target can vary wildly in appearance depending on aspect angles, atmospheric effects, and other variables. Promising results have been demonstrated and show the ATNN can detect the target object among broken pieces observed at various aspect angles, in spite of noise, tumbling, or fragmented pieces. The ATNN provides better dynamics and flexibility for the network itself to approach an efficient performance level and to optimize its configuration. Thus, ATNN can identify relationships in multi-channel data. Extended corollary is proposed such that the ATNN inherits the properties and capabilities of feed-forward network as a universal function approximator. The ATNN guides us to choose the appropriate interconnections in general multidimensional analysis.

6. APPENDIX: DATA CHARACTERISTICS

A detailed description of object trajectory data for the training set and test sets are listed in Table 7. As we can see, object components may tumble with different aspect angles or break into several pieces, where wh, ot, ft and fins represent warhead, oxidizer tank, fuel tank and tail fins, respectively. There are total of 36 different vehicles intercepted by the sensor device at high or low altitude (200km or 80km). Each object contains four component trajectories (wh, ot, ft, fins) and may have different aspect angles with respect to sensor devices. Furthermore, these components may break into several pieces and retains the major bulk. For instance, the object #1 in Table 7 is intercepted at lower altitude and all components are at the same aspect angle of 30° with no broken pieces. Object #7 is at a higher altitude with the same aspect angle of 90° , but the oil tank is broken and remains only one-third the size of the original oil tank. While components of object #9 turn out to be at different aspect angles: war head 60° , oil tank 90° , fuel tank 60° , and fins 30° with no broken pieces, etc..

For a specific vehicle object, four time series data of n time-steps (wh, ot, ft, fins) are received simultaneously and fed to the network. The neural networks (TDNN and ATNN) are trained to discriminate a warhead as the true target from the other pieces of components. For example, the training set includes 36 time-series data for these 9 objects (4×9). Two sampling rates are available: 0.5 second and 0.1 second.

Table 7. Detail description of the trajectories data in the training set and test sets. Object components may tumble with different aspect angles or break into several pieces, where wh, ot and ft represent warhead, oxidizer tank, and fuel tank, respectively.

Data Set	Object	Aspect Angle				Broken Pieces	Intercept Altitude
		wh	ot	ft	fins		
						N/A	Low
training set	#1	30°	30°	30°	30°	N/A	low
	#2	60°	60°	60°	60°	N/A	low
	#3	90°	90°	90°	90°	N/A	high
	#4	30°	30°	30°	30°	N/A	high
	#5	60°	60°	60°	60°	N/A	high
	#6	90°	90°	90°	90°	N/A	high
	#7	90°	90°	90°	90°	front 1/3 ot	high
	#8	30°	60°	30°	30°	N/A	high
	#9	60°	90°	60°	30°	N/A	high
test set #1	#10	90°	60°	60°	90°	N/A	high
	#11	90°	60°	60°	30°	N/A	high
	#12	60°	90°	60°	60°	N/A	high
	#13	60°	90°	30°	60°	N/A	high
	#14	60°	90°	90°	60°	N/A	high
test set #2	#15	60°	30°	60°	60°	N/A	high
	#16	60°	30°	60°	90°	N/A	high
	#17	30°	60°	30°	30°	N/A	high
	#18	30°	60°	90°	30°	N/A	high
	#19	30°	60°	90°	60°	N/A	high
test set #3	#20	90°	90°	90°	90°	front 1/3 ot	high
	#21	90°	90°	90°	90°	back 1/3 ot	high
	#22	90°	90°	90°	90°	front 1/2 ft	high
	#23	90°	90°	90°	90°	front 1/3 fins	high
test set #4	#24	60°	30°	90°	90°	N/A	high
	#25	60°	30°	90°	90°	N/A	low
	#26	60°	30°	90°	90°	N/A	high
	#27	60°	30°	90°	90°	N/A	high
test set #5	#28	30°	30°	30°	30°	N/A	low
	#29	60°	60°	60°	60°	N/A	low
	#30	90°	90°	90°	90°	N/A	low
	#31	30°	30°	30°	30°	N/A	high
	#32	45°	45°	45°	45°	N/A	high
	#33	60°	60°	60°	60°	N/A	high
	#34	75°	75°	75°	75°	N/A	high
	#35	90°	90°	90°	90°	N/A	high
	#36	90°	60°	90°	90°	N/A	high

REFERENCES

1. B. Bai and N. H. Farhat, "Learning networks for extrapolation and radar target identification," *Neural Networks*, Vol. 5, 1992, pp. 507-529.
2. A. R. Barron, "Universal approximation bounds for superpositions of a sigmoidal function," *IEEE Transactions on Information Theory*, Vol. 39, 1993, pp. 930-945.
3. D. Casasent and A. Ye, "Detection filters and algorithm fusion for ATR," *IEEE Transactions on Image Processing*, Vol. 6, 1997, pp. 114-125.
4. D. S. Chen and R. C. Jain, "A robust back propagation learning algorithm for function approximation," *IEEE Transactions on Neural Networks*, Vol. 5, 1994, pp. 467-479.
5. S. P. Day and M. R. Davenport, "Continuous-time temporal back-propagation with adaptive time delays," *IEEE Transactions on Neural Networks*, Vol. 4, 1993, pp. 348-354.
6. J. E. Dayhoff, P. J. Palmadesso, F. Richards, and D. T. Lin, "Patterns of dynamic activity and timing in neural network processing," O. Omidvar and J. Dayhoff, eds., *Neural Networks and Pattern Recognition*, 1998, pp. 105-141.
7. K. I. Funahashi, "On the approximate realization of continuous mappings by neural network," *Neural Network*, Vol. 2, 1989, pp. 183-192.
8. J. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the Theory of Neural Computation*, Addison Wesley, Redwood City, 1991.
9. L. Holmström and P. Koistinen, "Using additive noise in back-propagation training," *IEEE Transactions on neural Networks*, Vol. 3, 1992, pp. 24-38.
10. K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, Vol. 2, 1989, pp. 359-366.
11. B. Kamgar-Parsi, A. K. Jain, and J. E. Dayhoff, "Aircraft detection: a case study in using human similarity measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 23, 2001, pp. 1404-1414.
12. D. T. Lin, "Sampling effects on trajectory learning and production," *Journal of Information Science and Engineering*, Vol. 13, 1997, pp. 293-310.
13. D. T. Lin, J. E. Dayhoff, and P. A. Ligomenides, "A learning algorithm for adaptive time-delays in a temporal neural network," Technical Report SRC-TR-92-59, Systems Research Center, University of Maryland, 1992.
14. D. T. Lin, J. E. Dayhoff, and P. A. Ligomenides, "Trajectory production with the adaptive time-delay neural network," *Neural Networks*, Vol. 8, 1995, pp. 447-461.
15. J. L. McClelland, D. E. Rumelhart, and the PDP Research Group, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Vol. 2, MIT Press, Cambridge, 1986.
16. L. I. Perlovsky, W. H. Schoendorf, B. J. Burdick, and D. M. Tye, "Model-based neural network for target detection in sar images," *IEEE Transactions on Image Processing*, Vol. 6, 1997, pp. 203-216.
17. T. Poggio and F. Girosi, "Networks for approximation and learning," in *Proceedings of IEEE*, Vol. 78, 1990, pp. 1481-1497.
18. J. C. Principe, M. Kim, and J. W. Fisher, "Aided and automatic target recognition based on sensory inputs from image forming systems," *IEEE Transactions on Image Processing*, Vol. 7, 1998, pp. 1136-1149.

19. C. L. Resch, "New time delay neural network to distinguish exo-atmospheric warheads," Technical Report AM-93-E141, Applied Physics Laboratory, The Johns Hopkins University, 1993.
20. C. L. Resch, "Effects of jitter on the ability of a time delay neural network to distinguish exo-atmospheric warheads," Technical Report AM-94-E010, Applied Physics Laboratory, The Johns Hopkins University, 1994.
21. M. W. Roth, "Survey of neural network technology for automatic target recognition," *IEEE Transactions on Neural Networks*, Vol. 1, 1990, pp. 28-43.
22. K. P. Unnikrishnan, J. J. Hopfield, and D. W. Tank, "Connected-digit speaker-dependent speech recognition using a neural network with time-delayed connections," *IEEE Transactions on Signal Processing*, Vol. 39, 1991, pp. 698-713.
23. A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Transactions on Acoustics, Speech, Signal Processing*, Vol. 37, 1989, pp. 328-339.
24. A. Waibel, K. J. Lang, and G. E. Hinton, "A time-delay neural network architecture for isolated word recognition," *Neural Networks*, Vol. 3, 1990, pp. 23-43.
25. L. C. Wang, S. Z. Der, and N. M. Nasrabadi, "Automatic target recognition using a feature-decomposition and data-decomposition modular neural network," *IEEE Transactions on Image Processing*, Vol. 7, 1998, pp. 1113-1121.
26. S. S. Young, P. D. Scott, and C. Bandera, "Foveal automatic target recognition using a multiresolution neural network," *IEEE Transactions on Image Processing*, Vol. 7, 1998, pp. 1122-1135.



Daw-Tung Lin (林道通) received the B.S. degree in Control Engineering from National Chiao Tung University, Hsinchu, Taiwan, in 1985, and the M.S. and Ph.D. degrees in Electrical Engineering from the University of Maryland at College Park, MD, U.S.A., in 1990 and 1994, respectively. Since 1995, he has been on the faculty of the Department of Computer Science and Information Engineering at the Chung Hua University, Hsinchu, Taiwan, where he is currently an Associate Professor. His current research interests are neural networks and fuzzy system, medical image processing, and pattern recognition.