Intelligent Inertial Navigation System and Global Positioning System Navigator Based on Artificial Neural Network

*Mr. AHMED MUDHER HASSAN

Received on : 28 /11 2007
Accepted on :30 / 6 /2008

Abstract

The integration of Global positioning system (GPS) and Inertial Navigation System (INS) are continuously gaining interests in many positioning and navigation applications. Both systems have their unique features and shortcomings. Their integration offers systems that overcome each of their drawbacks and maximize each of their benefits.

An INS/GPS integration method based on Artificial Neural Networks (ANNs) to fuse INS measurements and GPS measurements has been suggested. It is also provide high performance INS/GPS integration with accurate prediction for position and velocity components during GPS signal absence. Thus the integration of the two systems presents a number of advantages and overcomes each systems inadequacy.

An ANN was adopted in this paper using position and velocity update architectures and utilizing the window based weight updating strategy to updates the navigation knowledge in the proposed INS/GPS integration system and improve the limitation of traditional weight updating strategy using two data test IMU systems.

Keywords: Inertial Navigation System, GPS, Neural Networks, real-time implementation.

الخلاصة:

التكامل بين منظومتي (GPS)و(INS) أكتسب أهتماما متواصلا في مختلف التطبيقات الملاحية والموقعية. كلا النظامين لهما خصائص ومساوئ. وتكاملهما يقدم نظام يتجاوز كل مساوئهما ويزيد من منافعهما.

أقترحت طريقة تكامل بين (INS/GPS) مبنية بأستخدام الشبكة العصبية (ANN) لتجميع قراءات (INS) و(GPS). كذلك فهي توفر أداء عالي لتكامل (INS/GPS) بتنبأ دقيق لمركبات الموقع والسرعة خلال توقف أشارة (GPS). هكذا فأن تكامل النظامين يوفر عدد من الفوائد و يتجاوز القصور في كل نظام.

الشبكه العصبيه تبنت في هذا البحث باستخدام شبكتين واحده للموقع واخرى للسرعه و بالاستفاده من الستراتيجيه المبنيه على تحديث الاوزان لتحديث المعلومات الملاحية في منظومة ال(INS/GPS) المقترحة وتحسين المحددات التقليدية لستراتيجيه تحديث الاوزان بأستخدام نوعين من القراءات ال(IMU) لاختبار المنظومة.

Abbreviations

ANNs	Artificial Neural Networks
GPS	Global Position System
INS	Inertial Navigation System
IMU	Inertial Measurement Unit
PUA	Position Updates Architecture
VUA	Velocity Updates Architecture

List of symbols

GPS (*i*)	*i*th GPS window
φ (·)	activation function
W (n)	synaptic weights
b	bias

Control and System Eng. Dept. University of Technology

1. Introduction

It is well established that Global Positioning System (GPS) can provide position and velocity information of moving platforms with consistent accuracy throughout the surveying mission [1].

However, as GPS-based navigation requires at least four satellites, a major drawback of GPS-dependence navigation systems is that their accuracy degrades significantly with poor satellite geometry, cycle slips, and satellite outages. Signal outage is more significant for land vehicle positioning in urban centers, which takes place when encountering highway overpasses or tunnels. On the other hand, an Inertial Navigation System (INS) measures the linear acceleration and angular rates of moving vehicles through its accelerometers and gyroscopes sensors, respectively. For short time intervals, the integration of acceleration and angular rate results in highly accurate velocity, position and attitude with almost no time lags. However, because INS position outputs are obtained by integration, they drift at low frequencies. To obtain very accurate outputs at all frequencies, the INS should be updated periodically using external measurements. For this purpose, INS measurements are integrated with GPS measurements to provide a navigation system that has superior performance in comparison with either a GPS or an INS stand-alone system.

This paper looks at way in high quality integration where low cost inertial sensors are used to obtain improved performance. Also, it is different from [2] which depend on strapdown inertial navigation system (SINS) as a stand alone navigation system. Where, the errors of a stand-alone INS grow with time because of residual bias errors in both accelerometers and gyroscopes and need to be modeled as done in this paper where the GPS is used to update the navigation solution provided by the INS using the weight updated strategy while the error growing through the time in

position and velocity which provided by the INS.

Artificial Neural Networks (ANNs) have shown to be very efficient in modeling complex dynamics in several engineering problems. Due to the nonlinear nature of ANNs, they are able to express much more complex phenomena than some linear modeling techniques. In fact, ANN is a powerful tool for solving nonlinear problems that map input data to output data without having any prior knowledge about the mathematical process involved. Multi-layer perceptron ANNs have been suggested for fusing data from an INS and the GPS and obtaining reliable navigation solutions. Two ANN-based INS/GPS integration architectures were investigated. These are the Position Update Architecture (PUA) and the Velocity Update Architecture (VUA). Both architectures are briefly discussed later in this paper.

2. Objectives

The objectives of this paper are to develop an intelligent navigator to fuse an INS data and GPS measurements utilizing the Artificial Neural Network (ANN) to update the synaptic weights as well as the GPS signal is available, also to evaluate the performance of the proposed ANN architecture using test data of two different INS systems (tactical and navigation grades IMU systems) by comparing the two grades IMU systems.

3. INS/GPS integration based on Artificial Neural Network

Artificial Neural Network (ANN) is made up of many computational processing elements called neurons or nodes. These nodes operate in parallel and are connected together in topologies that are loosely modeled after biological neural systems. The training of ANN is carried out to associate correct output responses to particular input pattern. Once trained properly, an ANN has

the ability to generalize when similar, but not identical patterns are introduced to the network [3, 4].

3.1 Artificial Neural Network

In this paper, a Multi-Layer Perceptron (MLP) network with the architecture shown in **figure (1)** is considered. The input layer of the network has two input neurons for the INS position or velocity components and the time, while the output layer has only one output neuron for the corresponding position in PUA or velocity in VUA. We have also considered only two hidden layers of six neurons as an optimum compromise between network complexity and performance. More complex network structures with more hidden layers and more neurons in each layer can be adopted. However, we have determined that the network architecture shown in **figure (1)** is appropriate enough for modeling the position and velocity components and achieving the desired accuracy.

The weights W and the biases b are the ANN parameters that are computed during the training procedure and they determine the input/output functionality of the network. The weights are multiplied by the inputs to each neuron while the biases are considered at each neuron to limit or lower the input to the activation function $\varphi(\cdot)$ [4]. A hyperbolic tangent function (tansigmoid) is employed inside the hidden neurons to model the non-linearity in the input/output relationship.

A linear activation function is considered at the neuron of the output layer to perform as a linear superposition of the outputs of the hidden neurons, **table (1) and (2)** contain more details of the network architecture and training parameters. It should be highlighted that individual ANN module of the form shown in **figure (1)** is designed for each position and velocity components. These types of networks are known as feedforward backpropagation neural network. The forward pass of the computation involves feeding the inputs to the network starting from the input layer [4]. The output is obtained and compared to the target (desired performance) to determine the estimation error. This error is propagated through the network in the backward direction (opposite to the flow of the input data) starting from the output layer and is utilized to update the computation of the network parameters. The forward and backward computations are repeated until the optimal values of the synaptic weights are achieved, which corresponding to certain objective mean square estimation errors. The network weights are updated according to certain learning rules to minimize the mean square value of the estimation error. In this paper, we have utilized the Levenberg-Marquadrt learning rule, which provides the fastest training algorithm among other learning rules.

4. Window Based Weights Updating Strategy

As the synaptic weights are the core components of the navigation knowledge, development a strategy to accumulate the acquired navigation knowledge by updating the synaptic weights whenever the GPS signal is available (no GPS outages) is most important.

In most of their applications, ANNs are trained using some known training data set (input/desired output) to obtain the optimal values of the synaptic weights via off-line training. For any other set of inputs, different from those used in training, the synaptic weights can then be applied to provide prediction of unknown network outputs. It is worth mentioning that ANNs weights are frozen after completing the training procedure and no further modification will be made during the prediction process [4].

In fact, off-line training mode can work well required to track direction changes and mimic the motion dynamics utilizing the latest available INS and GPS data. In other words, the synaptic weights should be

updated during the navigation process to adapt the network to the latest INS sensor errors and the latest dynamics condition whenever the GPS signal is available.

To implement such criterion, a window-based weights updating strategy, which utilizes the synaptic weights obtained during the conventional off-line training procedure (or probably from previous navigation missions) is stored in the database. This criterion utilizes the latest available navigation information provided by the GPS signal window to adapt the stored synaptic weights so that they can be applied to mimic the latest motion dynamic. The window-updated synaptic weights are stored after each training stage. They are then used as initial values for the weights to be estimated during the next training window or for prediction during GPS signal absence (outage). Prior to looking into the details of the window based weights updating strategy, several aspects of traditional weights updating strategies are given. Traditional methods can be classified as:

(1) sample-by-sample training, also known as online or sequential training, that modifies the synaptic weights for each input record after computing the weights updates;

(2) Batch training, which computes the synaptic weight updates for each sample and stores these values (without changing the weights). At the end of the whole training procedure, all the synaptic weight updates are added together and then the weights are modified with the accumulated synaptic weight updates [3]. One major difference between the sequential training approach and the batching training approach is that the sequential approach keeps the system weights constant while computing the error associated with each input sample. In contrast, the sequential training approach is constantly updating its weights. From an online operational point of view, the sequential mode of training is preferred over the batch mode since less local storage is required. In addition, the random presentation of available training patterns makes it less likely for the backpropagation algorithm to be trapped in a local minimum if the sequential mode of training is utilized. In contrast, using the batch mode of training provides a more accurate estimate of the gradient vector, thus giving more accurate estimation of the weights [3].

Another major advantage of the sequential training over the batch training arises if there is a high degree of redundancy in the data [5]. For example, suppose that a vehicle moving along a circular trajectory with a constant velocity under an ideal condition for ten runs. The whole training data set is ten times larger than a single run but contains a high level of redundancy. As a result, the batch training approach takes ten times longer than the sequential training approach to get the optimal values of network weights. On the other hand, the sequential training approach updates the weights after receiving each record of input samples. Therefore, it will not be affected by such highly redundant data. However, using the batch training approach, the network can learn more general relationships than using the sequential training approach as it usually utilizes most of the available training data at the same time instead of sample by sample. Both generalization and training efficiency are very critical for INS/GPS integration applications; therefore, it is very important to develop a weight updating method that can preserve the generalization ability without losing too much training efficiency. However in this paper a sequential training updates algorithm is used.

4.1. Development of Window Based Weight Updating Strategy

The window_based weight updating strategy considers the previously stored weights as the 'long term memory'. Although the stored weights might not be able to provide accurate prediction during a long GPS outage, they can be applied as the initial weights at the beginning of a new navigation mission. The GPS window signal concept is

then applied to introduce the new information, 'fresh memory', and modify the stored weights during navigation. In fact, this method combines the advantages of the sequential mode and the batch mode of training in order to make the training procedure suitable to real-time navigation applications. The proposed method attempts to enlarge the sample applied by the sequential training approach to fit a certain window length. In addition, the weights of each window are then updated using the batch training approach. In other words, the weights of each window are updated sequentially. As depicted in **figure (2),** the window_based weight updating strategy can be explained as follows:

(1) ***Weight initialization.*** The initial weights can be obtained using previously stored weights or a random initialization procedure. In this paper, the initial weights were obtained using random initialization at the first time when the ANN based INS/GPS architectures were set up. Consequently, the weights are stored after completing one navigation mission and are applied as the initial weights for the next mission. Accurate initial weights may significantly reduce training time.

(2) ***GPS signal reception.*** Within the first GPS window ($i = 1$), GPS(i), the weights are not updated, thus the stored weights are still the initial weights $W(i-1)$ (i.e., $W(0)$).

(3) ***GPS signal reception.*** At the next GPS window, GPS($i+1$), the stored weights, $W(i-1)$, are updated utilizing the presently available GPS information (GPS(i)). These weights are stored as $W(i)$ after training is completed. Steps 2 and 3 are repeated until a GPS signal blockage is detected.

(4) ***GPS outage.*** As depicted in **figure (3)**, in the case of a GPS outage (after GPS(i)), $W(i-1)$ is first applied for real time prediction and then $W(i)$ is then utilized to replace $W(i-1)$ and carry on real-time prediction during the remaining GPS outages.

Since the ANN training procedure takes time because of utilizing a backpropagation algorithm, updating the weights immediately at the latest available sample of GPS signal before outage is difficult. However, the utilization of the proposed strategy can still provide reasonable prediction accuracy during GPS outages since it provides the latest updated weights instead of real-time updated weights for real-time prediction. Therefore, failure in providing real-time updated weights does not mean the proposed strategy is not able to provide real time prediction. In contrast, it can utilize the latest updated and stored weights to provide real-time solutions. Combining the latest GPS window signals, the stored weights can be adaptively updated to follow the latest dynamics condition and INS errors, thus improving prediction accuracy during GPS outages.

5. PUA and VUA integration architectures utilizing of window based weight updating strategy

Both the PUA and VUA utilize multi-layer feed-forward NNs with a backpropagation training algorithm to integrate the data from an INS and GPS and mimic the dynamical model of the moving vehicle carrying both systems, After training the network, it can then operate in the prediction mode to provide the vehicle's position and velocity during GPS signal blockage (outage) or INS error.

Figure (3) shows the configuration of PUA and VUA. As long as the GPS signal is available, the position and velocity of GPS solutions are applied as the desired outputs for training the network and are also considered as the outputs to the PUA and VUN respectively. On the other hand, during GPS outages, the NN parameters are used in prediction mode for providing estimates of the position and velocity components utilizing the information derived from INS algorithm mechanization.

As long as the GPS signal is available, the GPS position and velocity and the INS outputs are used as the inputs to the PUA and VUA.

The training processes continue to improve estimation error in order to obtain the optimal values of NN parameters. During GPS outages, the NN parameters are, then, used in the prediction mode for providing estimates for the position and velocity components.

As indicated in **figure (3)**, no filtering algorithm is applied to process either the INS or GPS data. Both the PUA and VUA receive their inputs and the desired outputs (GPS position and velocity components) directly from the INS mechanization and GPS solutions, respectively. Not like the integration as in [6] were wavelet denoising algorithm was used to filter the GPS and INS outputs.

To evaluate the performance of the proposed window_based weight updating strategy, the NN work in prediction mode for this evaluation. In addition, a GPS window was implemented and continually slid over the GPS signal for window size duration, which equivalent to a number of GPS windows, as shown in **figure (4)**. The stored weight obtained from the first field test data were updated for each GPS window (during the availiality of GPS signal). In the case of a GPS outage, the latest updated weights stored in database were applied to provide real time prediction, as shown in **figure (4)**. Due to the nonlinear nature of neural networks, the PUA and VUA showed their capability of reducing the impact of uncompensated measurement errors. **Table (3)** lists the MSE in the prediction mode for the position and velocity in all directions. The results illustrated that the utilization of the proposed method to update the previously stored synaptic weights improved the prediction accuracy (for the GPS outage period).

Figure (5) shows the learning curves of the position and velocity components. From these figures it was found that the error in position is easier to predict than the velocity components.

Figure (6) shows a comparison between position and velocity difference for the two IMU grades the tactical and navigation grades. As shown in the **figures** that the position and velocity errors were in general didn't contain any long-term growth. In addition, the PUA and VUA for the navigation IMU grade provided the most stable and accurate solutions if compared to the tactical IMU grade which is needed a filtering process to denoising the GPS and INS outputs.

The results also demonstrated that using the PUA and VUA with a navigation grade IMU provided acceptable positioning and velocity accuracies. Than using it with a tactical grade IMU. In other words, the PUA and VUA are suitable for integrating GPS measurements with a navigation grade IMU. In general, the proposed ANN-based architectures have provided positioning and velocity accuracies far better than can be expected from a tactical grade IMU.

6. Conclusions

The conclusions drawn from the results presented in this paper are:

a. The parameters of the intelligent navigator are included in the navigation knowledge. Thus, they can be updated without a human expert during navigation whenever newly updated navigation knowledge is acquired.
b. A window_based weight updating strategy was introduced to use with ANN's for integrating GPS and INS systems in vehicular navigation application.
c. The stored synaptic weights can be adaptively updated to follow the latest motion dynamics and INS error characteristics.
d. The PUA and VUA architecture are recommended as the INS/GPS integration system using navigation grade IMU. It can achieve high level positioning accuracy requirement for real time prediction without INS computation components during GPS outages.

7. References

[1] Shin E-H and El-Sheimy N 2002 Accuracy improvement of low cost INS/GPS for land applications *the US Institute of Navigation, 2002 National Technical Meeting (San Diego, CA, 28-30 Jan. 2002)*.

[2] Salam I., "Intelligent SINS Navigator Based on ANN", IJCCCE, Univer

[3]

[4] sity of Technology, Vol. 6, No. 1, 2006.

[5] Saad, D. (1998): "On Line Learning in Neural Networks", Cambridge University Press.

[6]

[7] Ham, F. M. and Kostanic I., (2001): *Principles of Neurocomputing for Science and Engineering* (New York: McGraw-Hill).

[8] Fine T L (1999): *Feedforward Neural Network Methodology* (New York: Springer).

Ahmed M. Hassan, "A Neuro-Wavelet Techniques for GPS/INS System Integration" M.Sc. Thesis, Control and Systems Engineering Dept., University of Technology, 2005.

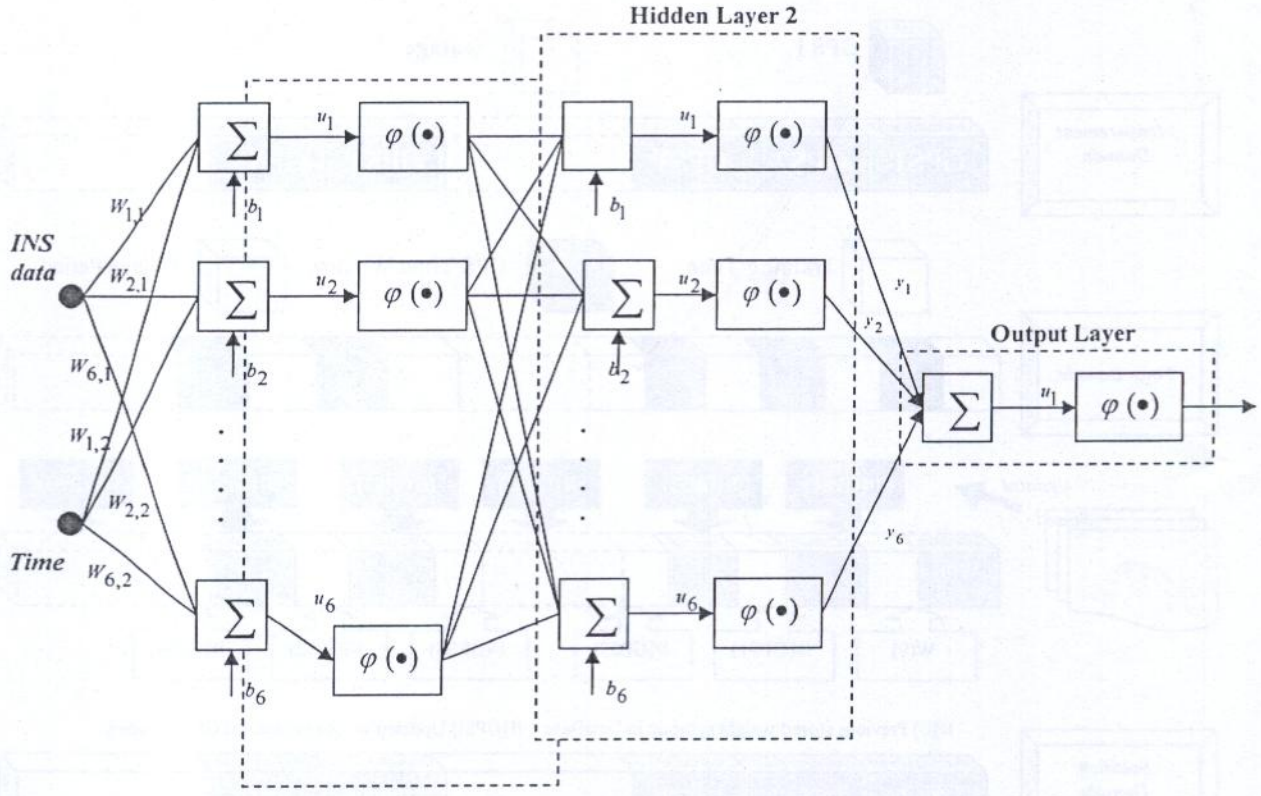

Figure (1): Architecture of a two hidden-layer MLP network for INS/GPS integration.

Table (1): Network Architecture specifies.

Layers	*Number of Neurons*	*Transfer Function*
1	2	Hyperbolic Tangent
2	6	Hyperbolic Tangent
3	6	Hyperbolic Tangent
4	1	Linear

Table (2): Network Training Parameters.

Network Architecture	Feed Forward
Performance Function	Mean Square Error
Training Function	Levenberg-Marquadrt Backpropagation
Epochs	500
Momentum Rate	0.001
Goal	0.0000000001

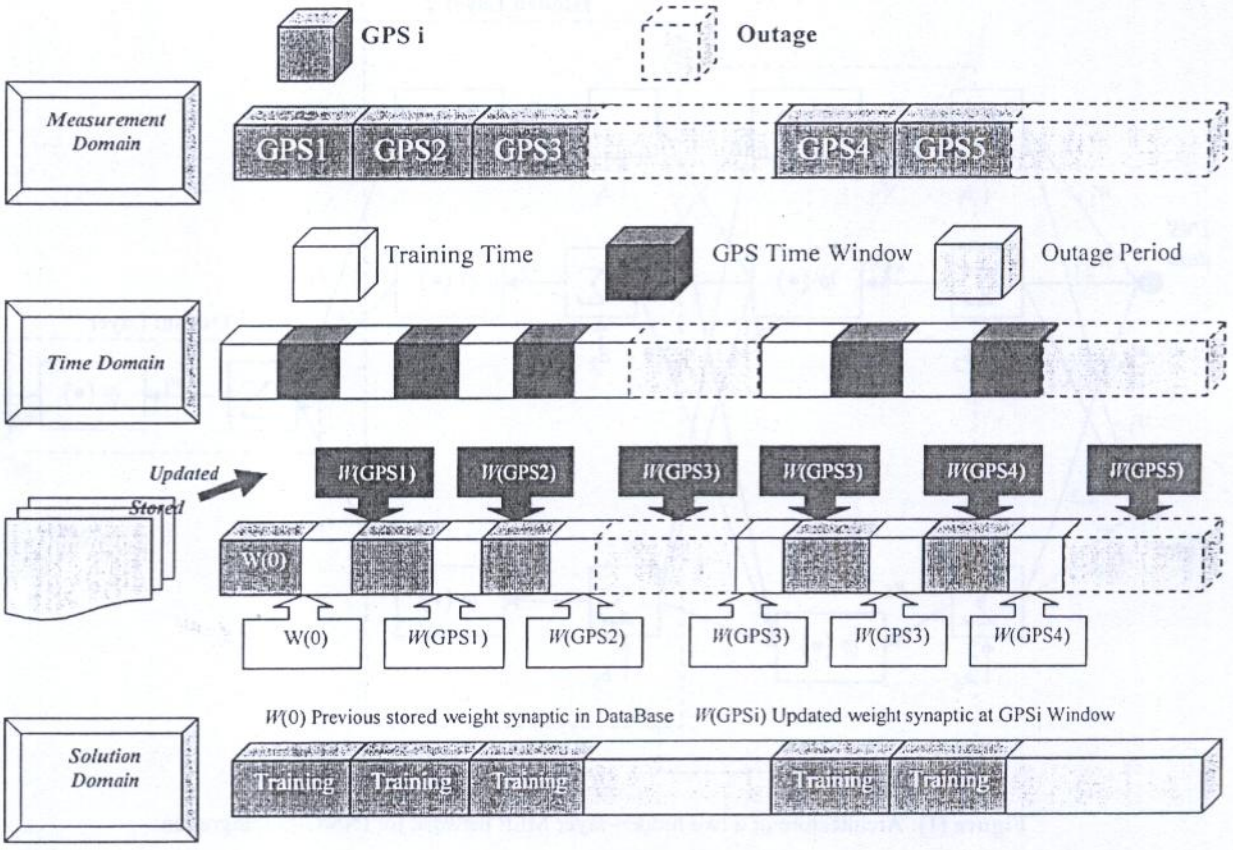

Figure (2): Window_Based Synaptic Weight Updating strategy.

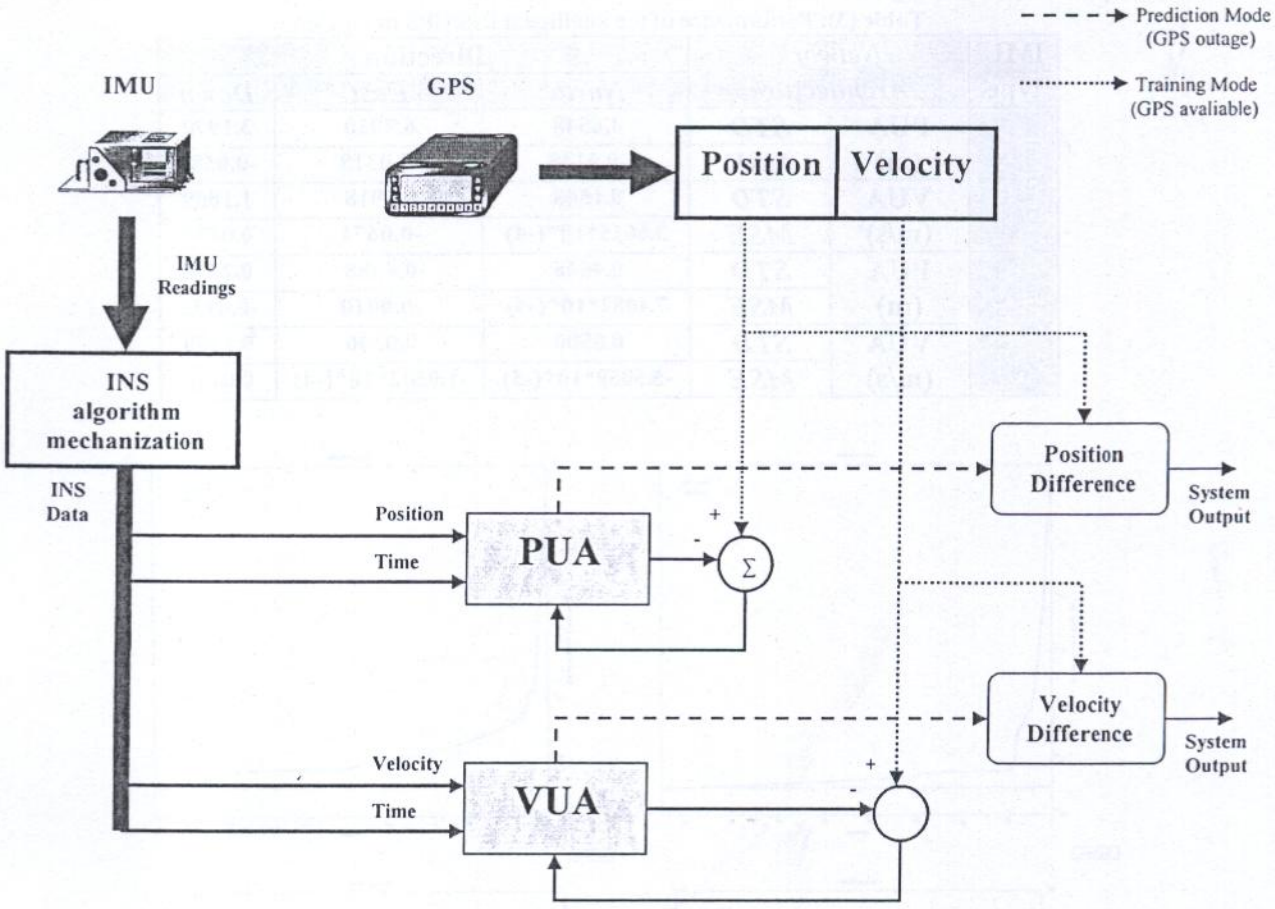

Figure (3): Configuration of the position and velocity update architecture.

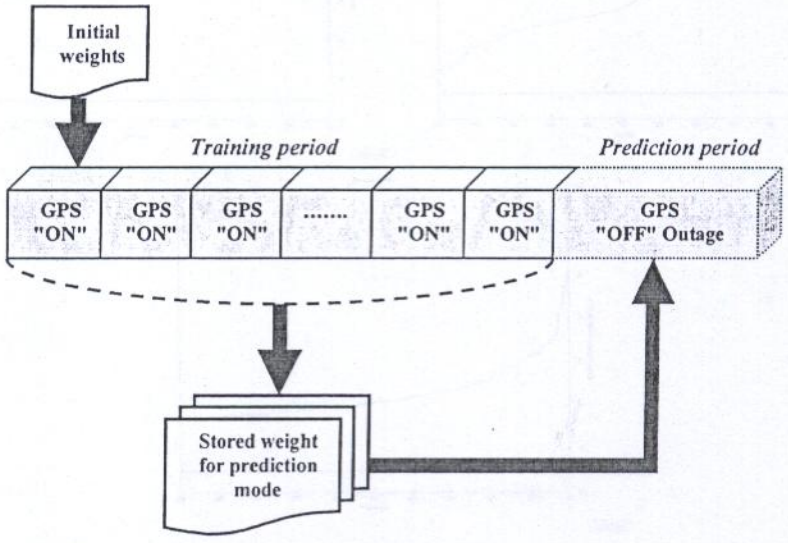

Figure (4): Window based-weight updating strategy.

Table (3): Performance of the intelligent INS/GPS navigator.

IMU type	Network Architecture		Direction		
			North	East	Down
Tactical grade	PUA (m)	STD	4.6548	6.9010	3.1970
		MSE	-0.0148	-0.0318	-0.0582
	VUA (m/s)	STD	0.1548	0.6918	1.1009
		MSE	3.6615*10^(-4)	-0.0671	0.0153
Navigation grade	PUA (m)	STD	0.4648	0.4388	0.2976
		MSE	7.4082*10^(-4)	-0.0010	-0.0132
	VUA (m/s)	STD	0.0500	0.0346	0.1350
		MSE	-5.5059*10^(-5)	-1.0502*10^(-4)	0.0122

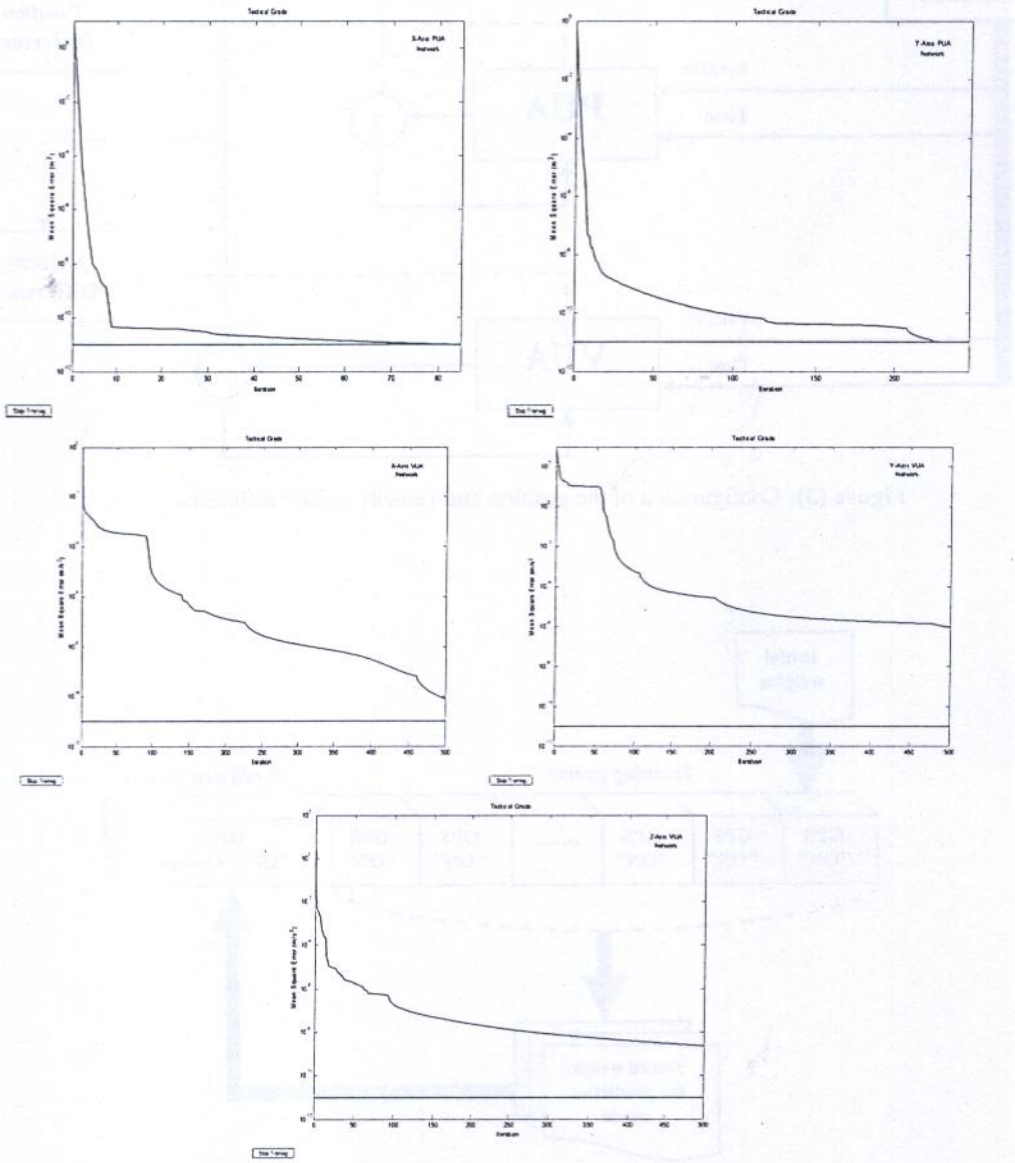

Figure (5): The learning curve of the networks in position and velocity for the tactical and navigation grades.

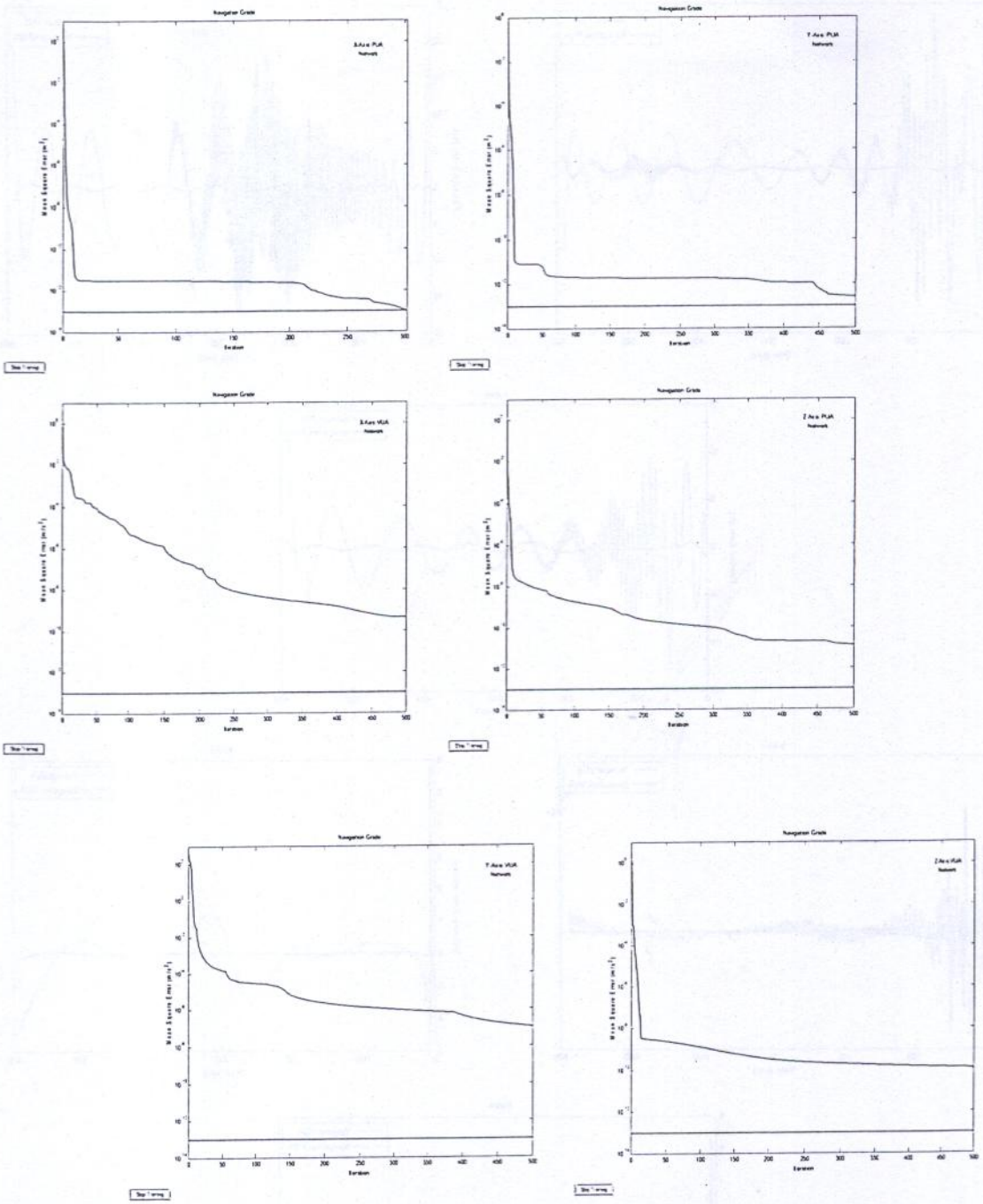

Figure (5): Continued.

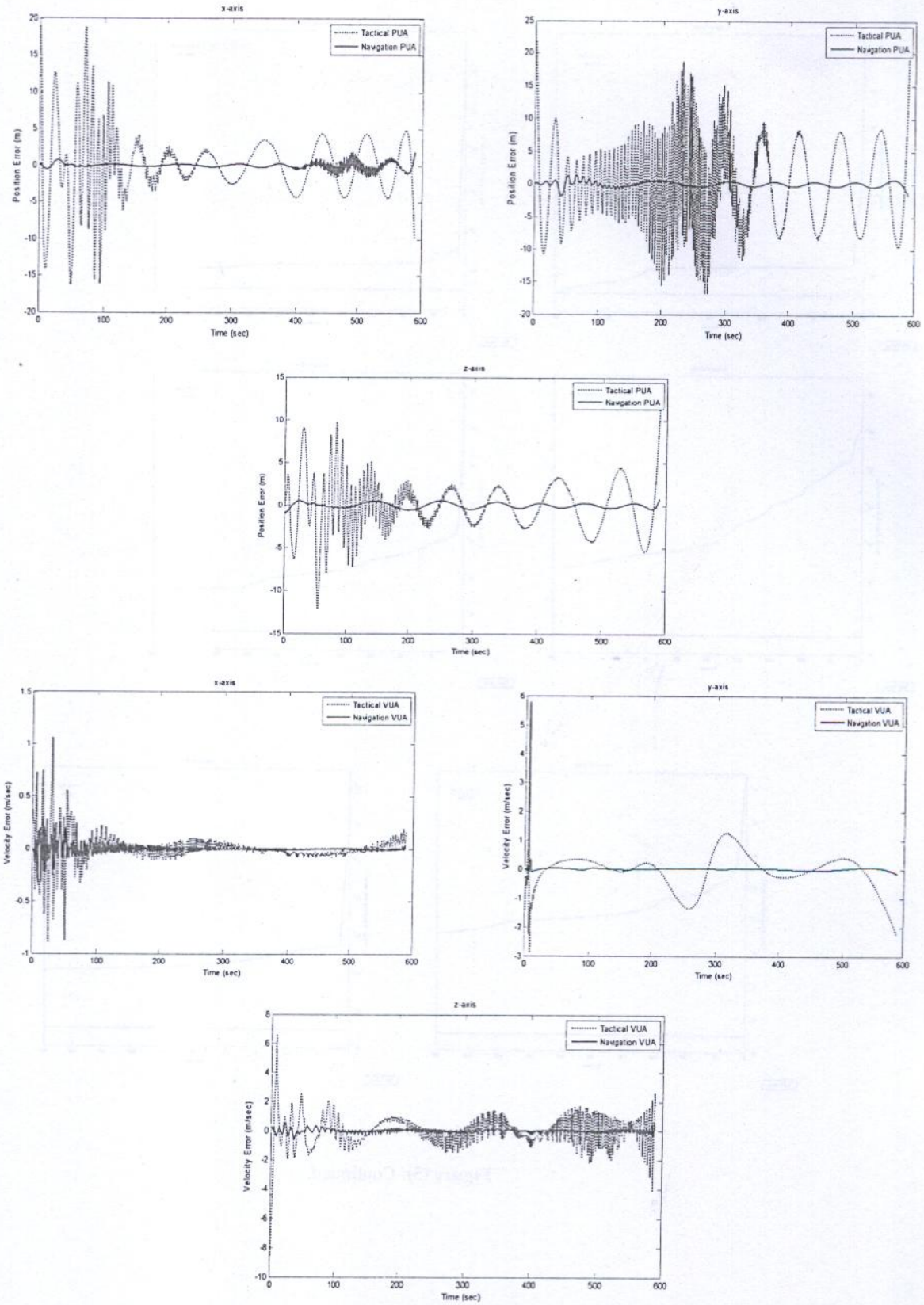

Figure (6): Comparison between Position and Velocity error difference for the two IMU grades.