ABSTRACT
An adaptive filter selection and tuning algorithm for a GNSS receiver is proposed wherein different pattern recognition approaches are employed to select proper adaptive multipath compensation and tracking techniques and corresponding tuning parameters based on the type of multipath environment and receiver motion mode (vehicular or pedestrian). Two different classification approaches, namely neural networks and support vector machines, are used for environment and motion state identification and the results are compared. Pattern recognition is performed based on a set of temporal and spectral features extracted from the correlation sequence of received signals. The multipath compensation tracking techniques are selected from a set of stochastic-gradient-based adaptive filters implemented in time and wavelet domains. The proposed algorithms are evaluated using real Galileo and GPS data collected in an urban environment via implementation in a software defined GNSS receiver.

INTRODUCTION
Multipath propagation poses significant challenges to satellite based navigation systems and remains a dominant source of accuracy degradation for high precision GNSS applications. Without accurate Line-Of-Sight (LOS) delay estimation in multipath environments, receivers cannot provide reliable position, velocity and time (PVT) estimates. Although there are many algorithms available that attempt to mitigate the effects of multipath, its mitigation remains an issue. The most common delay tracking algorithms include variants of the traditional delay-lock loop (DLL) method such as the double-delta correlator, strobe correlator and high resolution correlator techniques. Although these algorithms are effective when the receiver is subject to a few, weak multipath reflections, the performance of these techniques in severe multipath scenarios is still rather limited.

The other class of multipath mitigation techniques includes the advanced methods such as the Multipath Estimating Delay Locked Loop (MEDLL) [1], the Multipath Mitigation Technique, the Fast Iterative Maximum Likelihood Algorithm (FIMLA) [2], Sequential Maximum Likelihood [3], the Reduced Search Space Multipath Likelihood (RSSML) algorithm [4] and the deconvolution approaches [5]. This class of techniques is based on Maximum Likelihood (ML) estimation. These ML-based algorithms introduce large computational complexities to the receiver as a result of exploring large search spaces or performing matrix inversion procedures. At the cost of this complex multi-correlator structure, advanced estimation algorithms introduce multipath mitigation performance superior to that of correlation-based techniques. However, in some applications, this level of computational complexity may be expensive.

In order to mitigate the effect of multipath distortion on GNSS pseudorange measurements with a practically achievable computational complexity, this paper proposes an algorithm for adaptive compensation of the multipath channel through utilizing a tuned selection of stochastic-gradient-based approaches [6], including Least Mean Squares (LMS), Recursive Least Squares (RLS), Wavelet-based LMS (W-LMS) and Wavelet-based RLS (W-RLS) [12]. For example W-RLS and RLS are more appropriate than the other two algorithms in dealing with fast varying channels and W-LMS is more appropriate for slow varying channels and is less complex. Moreover, each of these algorithms has different tuning parameters that should be set according to the channel characteristics. In this paper, two pattern recognition algorithms, namely Neural Networks (NN) and Support Vector Machines (SVM) are used to automatically select the appropriate adaptive multipath-compensation-based tracking algorithm and the corresponding tuning parameters based on the characteristics of the multipath channel including the type of the environment and type of motion of the receiver. The pattern recognition of the channel samples is performed based on extracting a set of effective temporal and spectral features from the correlation sequence of the received GNSS signal. An optimum decision block is used in the structure of the adaptive multipath compensation methods to produce the control error signal in the decision feedback loop that is used to update the adaptive filters coefficients.

A set of simulations using GPS and Galileo signals is first used to produce a training sequence and test data
for different multipath environments including urban, suburban, indoor and open sky cases. The trained classifiers are then used in a separate simulation wherein a mixture of data produced under different multipath scenarios is used to evaluate the performance of the adaptive selection and tuning method. The performances of the adaptive selection method for different classifiers are compared. The proposed scheme is also tested using real Galileo and GPS signals in an urban environment (downtown Calgary), via implementation in a software defined GNSS receiver. In all of the real data tests, the position solutions computed using the proposed techniques are compared to reference trajectories obtained by a tightly coupled integrated GNSS-INS system and to the solutions computed by some of the conventional delay tracking techniques.

1. Multipath model

The received baseband signal in a multipath channel can be modelled as an $M$-path signal composed of a direct path and $(M-1)$ reflected rays plus an Additive White Gaussian Noise (AWGN) term $n(t)$ as

$$ r(t) = s(t) * h(t) = \sum_{k=1}^{M} A_k s(t - \tau_k) e^{j\phi_k} + n(t) $$

(1)

where $s(t)$ is the transmitted spread spectrum signal, $h(t)$ is the channel impulse response, $A_k, \phi_k$ and $\tau_k$ are the time-variant amplitude, instantaneous phase and delay parameters corresponding to the $k$th path. The received signal, after being down converted, filtered and sampled, is correlated with a replica of the Pseudo-Noise (PN) code. The output of the correlator can be expressed as

$$ y(\tau) = \sum_{k=1}^{M} a_k g(\tau - \tau_k) + w(\tau). $$

(2)

$$ \tau = 0, T, \ldots, (N-1)T $$

where $a_k = A_k e^{j\phi_k}$ is the complex path coefficient corresponding to the $k$th path, $g(\tau)$ is the ideal autocorrelation function of the PN code and $w(\tau)$ is the noise term at the output of the correlator. Equation (1) can be written in matrix form as

$$ y = aG + w, $$

(3)

where $y$ is the vector of the samples of $y(\tau)$ with a length of $N = \frac{T_p}{T_s}$, $T_p$ and $T_s$ being the search and sampling periods, and

$$ a = [a_1, a_2, \ldots, a_M]. $$

(4)

The vector $w$ with a length of $N$ is the vector of noise samples with a covariance matrix of $Q$, and $G$ is a $N \times M$ matrix that can be represented as

$$ G = [g_{a_1}, g_{a_2}, \ldots, g_{a_M}]^T $$

(5)

where $g_{a_k} = [g(0-\tau_k), g(T_1-\tau_k), \ldots, g((N-1)T_1-\tau_k)]^T$.

The statistical distribution of LOS and multipath parameters including the complex attenuation coefficients, number of paths and the delay parameters of different components as well as the temporal variations of all of these parameters are greatly affected by the type of multipath environment (size and shape of the reflectors as well as their spatial distribution) and the type of receiver motion. The statistical behaviour of multipath components determines the pattern of the autocorrelation function of the received GNSS signal and its temporal variations. In addition, different multipath patterns require different tracking strategies to achieve optimized tracking performance, and consequently optimized positioning accuracy. Therefore, identifying the type of multipath environment and the state of the receiver motion from the correlation pattern provides insightful knowledge about the statistics of the channel which can be used for adjusting the tracking strategy or the tracking parameters to obtain the best attainable tracking performance under various signal conditions. In this paper, the channel simulation software provided in [13] is used to generate received signal patterns that typically appear in different multipath environments. These generated patterns are used to train the classifiers.

For all of the simulation cases, in the pedestrian motion state the maximum speed of the receiver is 7 km/h and in the vehicular state the maximum speed is 50 km/h. In general, for the urban case the delay spread parameter of the channel takes larger values compared to the suburban case. Moreover, the urban and sub-urban channels can be better distinguished under pedestrian motion state rather than vehicular state.

2. Multipath pattern recognition

The task of a classification module is to learn patterns from the training data set that help to classify the observed data into different classes of interest. A classifier should be trained using a sufficient set of data in the training stage during which a set of coefficients is learned. These coefficients map input features to output classes. The training set should include all the varieties expected to be captured by the classifier. After completing the training stage, the classifier can be used to assign an output class to any new data that falls into one of the learned classes using the coefficients learned.
The most important stage in designing a classifier is selecting proper features that are able to effectively capture all the properties of data to distinguish between different classes. Having a very small number of features will result in a biased classifier that represents poor accuracy on both training and test data. However, learning with too many features in addition to increasing the computational complexity of the system, will result in over-fitting the data that cause a good performance on the training set but poor accuracy on the test data set. Obviously, both of the above cases should be avoided using well designed features. The following subsections explain the features and classifiers used herein.

**Feature Extraction**

The features for multipath classification are extracted from the correlation sequence of the received GNSS signal rather than the channel parameters themselves. This method of feature extraction is much more feasible than direct use of channel coefficient since accurate estimation of all the channel parameters requires very high sampling rates (hundreds of MHz). Two different sets of features, temporal and spectral, are extracted to represent each channel sample. The temporal features represent the type of channel and are used to classify the type of environment. The spectral features, however, characterize the type of receiver motion.

The correlation delay axis is divided into 16 bins. The first 15 bins are equispaced and cover the range of delays from zero to half of a chip. The last bin represents delays from half a chip to one chip. Therefore, there are 32 temporal features consisting of the magnitudes and relative phases of the correlation function at the centre of each bin. In order to extract the spectral features, a Fourier transform is performed on every 20 successive training channel samples. Therefore, for each correlation delay bin, 20 time-successive samples contribute in computing a Fourier-transformed sequence from which frequency-domain features are extracted. The index of the dominant spectral peak and its bandwidth are the two spectral features extracted for each bin. Therefore, there is a total of 32 frequency features for each channel sample.

![Figure 1: Feature extraction methodology for multipath pattern recognition](image)

After extracting the features matrix, \( \mathbf{x} \), each column of the matrix is normalized as

\[
x_j = \frac{x_j - \text{mean}(x_j)}{\text{max}(x_j) - \text{min}(x_j)}
\]

Therefore after normalization, all the features will be between 1 and -1.

**Classification Based on Neural Networks (NN)**

The neural network considered here consists of three layers as shown in Figure 2. The first layer is the input layer with 64 nodes which consists of the extracted features for the current input epoch. The second layer is a hidden layer with 15 nodes and the third layer is the output layer with six nodes where each node represents one of the output classes namely vehicular-urban, pedestrian-urban, vehicular suburban, pedestrian-suburban, indoor and open sky. A neural network provides a nonlinear mapping between input features and output labels. It is one of the most effective and less computationally complex classifiers.

In general, in a neural network that consists of \( L_N \) layers, \( L_N-1 \) matrices containing the required weight coefficients are trained. Each of these matrices transfers the node values of one layer to the corresponding one in the next layer. Before multiplying the node values of one layer to the corresponding weight matrix, one bias node with the value of unity is added to the current layer. In Figure 2 the weight matrices are shown by \( \theta^{(1)} \) and \( \theta^{(2)} \). After multiplying the activation values of one layer to the corresponding weight matrix, the resulting values are passed through a non-linear function, namely a sigmoid function, to obtain the activation values of the next layer. Therefore, if the activations of layers \( j \) and \( j+1 \) are referred to as \( a_j \) and \( a_{j+1} \) respectively, the relation between them can be represented as

\[
a_{j+1} = \text{sigmoid}\left( \theta^{(2)} \begin{bmatrix} 1 \\ a_j \end{bmatrix} \right) = \frac{1}{1 + e^{-\theta^{(2)} [a_j]}}
\]

This forward propagation continues until it reaches the last layer.

![Figure 2: The architecture of a neural network with one hidden layer](image)
Afterward, a back propagation algorithm is employed to update the weight matrices by flowing the gradient of the following cost function (that is to be minimized) computed at the output layer all the way back to the first layer as

$$J(\theta) = \frac{1}{m} \left[ \sum_{i=1}^{m} \sum_{j=1}^{s} y_{ij}^{(i)} \log \left( h_{\theta_s}(x_{ij}^{(i)}) \right) + \frac{\lambda'}{2m} \sum_{k=1}^{s} \sum_{l=1}^{s} \sum_{j=1}^{m} \left( \theta_{ij}^{(k)} \right)^2 \right]$$

(8)

where $K$ is the number of the output layers, $x_{ij}^{(i)}$ is the input layer (features) for the $k$th training epoch, $h_{\theta_s}(x_{ij}^{(i)})$ is the $k$th element of the computed output, $y_{ij}^{(i)}$ is the $k$th element of the true output vector for the $k$th epoch of data (it is a vector containing 1 in the index corresponding to the true class and zero elsewhere), $m$ is the total number of input frames and $\lambda'$ is the regularization parameter which is used to avoid an over fitting problem. The forward and backward propagation procedures are iteratively repeated until convergence is obtained (normally after a certain number of iterations).

**Support Vector Machines (SVM) Classification**

Support vector machines are among the best learning algorithms developed from statistical learning theory [14]. Similar to all other classification approaches, SVM learns a mapping from input features $x$ to an output class label $y$ ( $x\mapsto y$ ). In the simplest two-class classification case, the class labels are defined as either positive or negative one ( $y \in \{\pm 1\}$ ). The theory behind SVM is based on the idea of separating the positive and negative examples of the training set with a maximized geometric margin to the decision boundary (hyperplane) as shown in Figure 3. This is equivalent to optimizing the prediction confidence on the training data [15]. Since the input data may be linearly non-separable in the original feature space, it is projected into a much higher dimensional space $\mathbb{R}^l$ ($n$ may be infinite) using a non-linear mapping function $\phi(x)$, where $x$ is the matrix of original features in the $\mathbb{R}^l$ space ($l \ll n$), and $l$ is the original number of features for each training example. Therefore, $x$ is an $m$ by $l$ matrix whose $i$th row contains features of the $i$th training example.

![Figure 3: Maximizing the geometric margin in SVM](image)

After projecting the training data into the new space, the SVM trainer searches for a linear discriminant function $f(x) = w\phi(x) + b$ in the projected feature space (since training examples will be linearly separable after projection), and patterns are classified by the sign of $f(x)$. In this case, the geometric margin of the hyperplane parametrized by $(w, b)$ with respect to a training example $(x^{(i)}, y^{(i)})$ is defined as

$$y^{(i)} = y^{(i)} \left( \frac{w}{\|w\|} \phi(x^{(i)}) + b \right)$$

(9)

Given a training set $S = \{(x^{(i)}, y^{(i)}); i = 1, \ldots, m\}$, the geometric margin over the whole set is defined as the smallest of the geometric margins on the individual training examples. To maximize the geometric margin, SVM solves the following optimization problem [16]:

$$\max_{w, b} \frac{1}{2}\|w\|^2 + C \sum_{i=1}^{m} \left[ y^{(i)}H_i \left( \phi(x^{(i)}) + b \right) \right]$$

(10)

where $C$ is the regularization parameter, and $H_0$ and $H_1$ (Hinge loss functions), shown in Figure 4, can be represented as

$$H_i(z) = \max(1-z, 0), \quad H_{-i}(z) = \max(1+z, 0)$$

(11)

![Figure 4: Hinge loss functions](image)

Optimizing the cost function in (10) can be entirely written in terms of the inner products of $\phi(x^{(i)})$'s (Burges 1998). Thus, one does not need to know the high dimensional mapping function $\phi(x)$ to solve the optimization problem. Instead, the cost function can be represented and optimized only as a function of the dot
products of $\phi(x^{(i)})s$, which is referred to as the kernel. Specifically, the computation of $K(x,x') = \phi^T(x)x\phi(x')$ is much less expensive than that of $\phi(x^{(i)})$ and $\phi(x^{(j)}).$ The kernel function is a measure of similarity between the two examples. Several different non-linear kernels have been introduced and tested in the literature, including the polynomial kernel, radial basis function (RBF) and sigmoid kernel. Among those the RBF kernel, also known as the Gaussian kernel, is one of the most practical choices (Hsu et al 2003) and will be used here.

3. Test results for the trained classifier

The same channel models used in generating the training set and the cross-validation set examples (channel models in [13]) are also used in an independent simulation to generate the test set. The simulation combines the delayed and attenuated versions of a single path Galileo signal using the simulated channel parameters to generate correlation function patterns of the received signal based on (1). From each generated pattern, a total of 64 features are extracted based on the strategy explained in the previous section. For this test, the training set contains 10,000 patterns/class, and each of the cross-validation set and the test set contains 2000 patterns/classes. Six different classes are considered: Urban-Vehicular, Urban-Pedestrian, Suburban-Vehicular, Suburban-Pedestrian, Indoor and Open Sky. Table 1 shows the accuracy of classification for each class after evaluating the trained classifiers on the test data set. The results in the first column are based on the NN classification and the results in the second column correspond to the SVM classification.

<table>
<thead>
<tr>
<th>Class/Type</th>
<th>Accuracy of NN [%]</th>
<th>Accuracy of SVM [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Urban-Vehicular</td>
<td>68.9</td>
<td>71.2</td>
</tr>
<tr>
<td>Urban-Pedestrian</td>
<td>84.8</td>
<td>78.0</td>
</tr>
<tr>
<td>Suburban-Vehicular</td>
<td>88.3</td>
<td>84.1</td>
</tr>
<tr>
<td>Suburban-Pedestrian</td>
<td>92.5</td>
<td>87.3</td>
</tr>
<tr>
<td>Indoor</td>
<td>100</td>
<td>95.7</td>
</tr>
<tr>
<td>Open sky</td>
<td>100</td>
<td>100</td>
</tr>
</tbody>
</table>

The results in the table show that for most classes (except the Urban-Vehicular case where the two classifiers show almost similar performances), the NN classifier shows classification accuracy superior to that of the SVM classifier, specifically in distinguishing between urban and suburban classes. The table shows that urban and suburban patterns are more separable under the pedestrian motion state rather than in the car motion state. Open sky and indoor patterns are almost perfectly distinguished from urban and suburban cases. The number of iterations for NN was set to 50.

4. Multipath compensation using stochastic gradient method in time domain

Herein, the stochastic-gradient-based adaptive filtering concept is employed to compensate for the distortion produced by multipath on the autocorrelation sequence of the received signal. The input to the filter is the vector of the autocorrelation sequence of the received signal ($\hat{y}$) and its output is the estimate of the autocorrelation function of the LOS signal ($\hat{y}$). Therefore, the relation between the input and output vectors can be written as

$$\hat{y} = yC$$  \hfill (12)

where $C$ is an $N$ by $N$ matrix of the compensation coefficients. Thus, the problem of interest is to find an optimum value for the matrix $C$. In order to find this matrix, the following mean square error cost function is minimized at the output of the compensation filter:

$$J(C) = E[\|d - \hat{y}\|^2] = tr\left[ E\left( (d - yC)\Gamma_0 (d - yC)^T \right) \right]$$  \hfill (13)

where $d = a_{LOS} g_{LOS}^T$ is the true LOS correlation function, and $g_{LOS}^T$ is the vector of ideal correlation functions shifted by $\tau_{LOS}$. $\Gamma_0$ and $a_{LOS}$ are the complex path gain and delay parameters associated with the LOS signal. There are two recursive solutions for minimizing (13) that can be expressed as [7]

$$C_k = C_{k-1} + \mu\left[ \Gamma - RC_{k-1} \right]$$  \hfill (14)

$$C_k = C_{k-1} + \mu R^{-1}\left[ \Gamma - RC_{k-1} \right]$$  \hfill (15)

where $R = E[y^n y^T]$, $\Gamma = E[d^n y]$ and $0 < \mu \ll 1$ is a positive constant called step parameter. These recursive optimisation algorithms are referred to as gradient descent and Newton’s method and they require perfect knowledge of the autocorrelation and cross-correlation matrices. However, computing the true values of these matrices is not possible for time-varying channels. For this reason, stochastic gradient approaches are adopted to provide low-complex solutions capable of adaptive tracking of the estimated parameters by replacing $R$ and $\Gamma$ by their approximations. Herein, two stochastic
The LMS algorithm is derived by replacing \( \mathbf{R} \) and \( \Gamma \) in (15) by their instantaneous approximations, \( \hat{\mathbf{R}} = \mathbf{y} \mathbf{y}^T \) and \( \hat{\Gamma} = \mathbf{y} \mathbf{d} \), which results in the following recursive formula [7]:

\[
\mathbf{C}_k = \mathbf{C}_{k-1} + \mu \mathbf{y}_k \mathbf{d}_k - \mathbf{C}_{k-1} \mathbf{y}_k \mathbf{y}_k^T
\]

(16)

where \( k \) and \( k-1 \) are time indices. The RLS algorithm is obtained by replacing \( \mathbf{R} \) in (16) by its exponentially weighted time average and results in the following joint recursions [8]:

\[
\mathbf{P}_k = \lambda^{-1} \left[ \mathbf{P}_{k-1} - \lambda^{-1} \mathbf{P}_{k-1} \mathbf{y}_k \mathbf{y}_k^T \mathbf{P}_{k-1} + \frac{1}{1 + \lambda^{-1} \mathbf{y}_k \mathbf{y}_k^T} \right], \quad \mathbf{P}_{k-1} = \lambda^{-1} \mathbf{I}
\]

(17)

\[
\mathbf{C}_k = \mathbf{C}_{k-1} + \mu \mathbf{y}_k \mathbf{d}_k - \mathbf{C}_{k-1} \mathbf{y}_k
\]

(18)

where \( \mathbf{P}_k = \mathbf{R}_k^{-1} \). In (17), \( \lambda \) is a scalar that should be selected in the range of \( 0 < \lambda \leq 1 \). When a value smaller than unity is assigned to \( \lambda \), the recent samples are associated with larger weights than the previous ones. This strategy enables a tracking mechanism for the adaptive system.

The vector \( \mathbf{d}_k \) in (16) and (18) is unknown on the receiver side. Therefore, a feedback technique is developed to solve this problem by adding a LOS signal estimation block to the system. This block provides an estimate of the transmitted data \( \hat{\mathbf{y}}_k \) that is used as a substitute for \( \mathbf{d}_k \) to update the coefficients of the compensation matrix. Since the output of the adaptive compensation block is supposed to be multipath-free in the steady state, the best estimate of the LOS peak from this data is the ML estimate under the assumption of the presence of only a single path. When using this strategy, the output of the LOS estimation block still has the form of

\[
\hat{\mathbf{d}}_k = a_{\text{LOS},k} \mathbf{g}_{\text{LOS},k}
\]

in which \( \hat{\mathbf{y}}_{\text{LOS},k} \) and \( a_{\text{LOS},k} \) are represented by [12] and

\[
\hat{\mathbf{y}}_{\text{LOS},k} = \arg \max_r \left\{ \mathbf{g}_r^T \mathbf{G}^{-1} \hat{\mathbf{y}}_k \right\}
\]

(19)

\[
a_{\text{LOS},k} = \frac{\mathbf{g}_{\text{LOS},k}^T \mathbf{G}^{-1} \hat{\mathbf{y}}_k}{\mathbf{g}_{\text{LOS},k}^T \mathbf{G} \mathbf{g}_{\text{LOS},k}}
\]

(20)

Therefore, in (16) and (18), \( \mathbf{d}_k \) is replaced by \( \hat{\mathbf{d}}_k \).

5. Multipath compensation in wavelet domain

Wavelet Transform (WT) is a tool for simultaneous time and frequency analysis of non-stationary signals using a set of orthonormal waveform series. Discrete Wavelet Transform (DWT) can be used to estimate or equalize the channel impulse response through deconvolution in the wavelet domain [9]. In this section, channel estimation through deconvolution in the wavelet domain is first introduced and then a technique for DWT-based adaptive multipath mitigation is proposed. Implementing the adaptive algorithms discussed in the previous section in the wavelet domain decreases their computational complexity and provides signal denoising.

Both DWT (decomposition) and IDWT (reconstruction) procedures are implemented via a bank of linear low-pass and high-pass filters. When orthonormal wavelet bases are used for analysis and synthesis filters, there is no net effect of the successive DWT and IDWT operations on the convolved output. Therefore, the channel filter can be merged into the DWT portion of the filter bank as shown in Figure 5. For the sake of simplicity and without loss of generality, only one level of decomposition is considered here. In this figure, wavelet analysis and synthesis operations are shown for both transmitted and received signals. The dashed line in Figure 5 indicates the locations in the signal flow of both block diagrams where the DWT approximation coefficients and detail coefficients are the same for the two cases and can be expressed as

\[
\mathbf{y}_a = (\mathbf{F}) \mathbf{h} = \mathbf{F} (\mathbf{G} \mathbf{h}) = \mathbf{Fy}
\]

(21)

\[
\mathbf{y}_d = (\mathbf{F}) \mathbf{h} = \mathbf{F} (\mathbf{G} \mathbf{h}) = \mathbf{Fy}
\]

(22)

where \( \mathbf{y}_a \) and \( \mathbf{y}_d \) are the vectors of the approximation and detail coefficients of \( \mathbf{y} \), \( \mathbf{F} \) and \( \mathbf{F} \) are the matrices of low-pass and high-pass analysis filters, \( \mathbf{F}_a \) and \( \mathbf{F}_d \) are the matrices of the reconstruction (synthesis) filters.

Therefore, either equation (21) or (22) can be used to estimate \( \mathbf{h} \) using a deconvolution method. However, since the thermal noise mostly appears at the output of the high-pass filter and bias terms such as multipath appear at the output of the low-pass filter, the approximation equation \( (\mathbf{F}) \mathbf{h} = \mathbf{Fy} \) is used for channel estimation to provide denoising to the system. For this reason, in this section, the adaptive algorithms introduced previously are implemented in the wavelet domain in order to increase the robustness to noise and decrease the computation load of the system [12].

Considering \( \mathbf{G}_a = \mathbf{F} \mathbf{G} \) is the wavelet transform of \( \mathbf{G} \), the proposed wavelet-based LMS (W-LMS) and wavelet-based RLS (W-RLS) algorithms implement \( (\mathbf{F}) \mathbf{h} = \mathbf{Fy} \) by replacing \( \mathbf{y}_k \) and \( \mathbf{d}_k \) by their wavelet-domain transformed versions as
6. Real data results

In this section, a set of GPS L1 C/A and Galileo E1b/c results are demonstrated to further compare the performance of the proposed adaptive filter selection and tuning algorithms with those of the fixed strategy trackers with live signals. Moreover, the performance of the proposed algorithms is compared to the conventional Narrow Correlator (NC) technique. The test data set was collected in downtown Calgary. A tightly-coupled integrated GPS-INS (Inertial Navigation System) on the vehicle was used to obtain continuous reference position solutions with a 1-m accuracy for the purpose of performance assessment; a NovAtel antenna and SPAN system mounted on vehicle were used for this purpose. The sampling frequency of the front-end digitizer was 20 MHz. The proposed delay estimation techniques were implemented in a software receiver. The loop update rate was 20 ms and the PLL and DLL bandwidths were 15 and 2 Hz. In the implementation of narrow correlators, a correlator spacing of 0.1 chip was utilized. The parameters $\mu$ (for LMS and W-LMS) was set to 0.07 and $\lambda$ (for RLS and W-RLS) was set to 0.87. For wavelet transformation, Haar wavelet filters with one level of decomposition were used. This choice of wavelet was selected based on the deconvolution-based estimation results presented in [10]. The sky plot of the satellites and the data collection trajectory are shown in Figure 7.

The two adaptive filter selection and tuning algorithms (referred to as MixedNN and MixedSVM in the figures) detect the type of the multipath environment for each time snapshot using the trained NN and SVM classifiers and then select the proper tracking strategy. The W-RLS algorithm was employed for three of the scenarios, namely vehicular, pedestrian urban and vehicular suburban. For each of these cases, the value of $\lambda$ was selected based on the numbers provided in Table 2. For the suburban pedestrian scenario, the method uses W-LMS with the corresponding value of $\mu$ provided in Table 2 (since WLMS is less complex than WRLS [12]).
Table 2: Selected values of tuning parameters of adaptive algorithms under different multipath scenarios

<table>
<thead>
<tr>
<th>Scenario/Algorithm</th>
<th>Urban vehicular</th>
<th>Urban Pedestrian</th>
<th>Suburban vehicular</th>
<th>Suburban pedestrian</th>
</tr>
</thead>
<tbody>
<tr>
<td>WRLS/RLS</td>
<td>$\lambda = 0.75$</td>
<td>$\lambda = 0.8$</td>
<td>$\lambda = 0.95$</td>
<td>$\lambda = 0.95$</td>
</tr>
<tr>
<td>WLM/LMS</td>
<td>$\mu = 0.011$</td>
<td>$\mu = 0.01$</td>
<td>$\mu = 0.005$</td>
<td>$\mu = 0.005$</td>
</tr>
</tbody>
</table>

The time series of estimated position errors and associated RMS are shown in Figure 8 and Figure 9.

The most important observation from these figures is that both context-based algorithms outperform the best of the fixed-strategy algorithms which are based on the W-RLS technique. The NN-based strategy has resulted in an improvement of about 35% in RMSE of the up component and the SVM-based strategy has resulted in a similar improvement in the north component of the position solution. As expected from the results in Table 1, the NN-based strategy generally shows a slightly better performance compared to the SVM-based strategy (except for the North component).

Comparing the fixed strategy techniques together shows that, in general, the largest RMSE values correspond to the NC algorithm. However, the performance difference between NC and LMS is barely noticeable. RLS shows an improvement of about 14% to 33% compared to the LMS and NC. The W-LMS and W-RLS algorithms demonstrate important improvements compared to their time domain duals. For example for the up component, the RMSE value is 27 m for W-RLS and 51 m for RLS algorithms, which is equivalent to a 47% improvement. This improvement is smaller for the east and north components of this trajectory. W-LMS shows similar improvements compared to the time domain LMS. Another important observation is that the performance of LMS and RLS is similar in the wavelet domain.

7. CONCLUSIONS

An adaptive filter selection and tuning algorithm for GNSS receivers was proposed: it uses neural networks and SVM classification approaches to identify the type of multipath environment based on correlation samples of the received signals and selects and tunes the receiver’s tracking strategy accordingly. The accuracies of the trained classifiers were evaluated using a test data set and the results showed that the NN-based classifier slightly outperforms the SVM-based classifier for the case of multipath pattern recognition. The trained classifiers were used to select and tune the receiver’s tracking strategy. Stochastic-gradient-based adaptive filters were tested using a set of real data including both Galileo and GPS signals. The test results showed that the classification-based adaptive selection and tuning algorithms resulted in improvement of 5% to 35% in the estimated position RMS errors.

REFERENCES


