A Comparison of the Performance of Prediction Techniques in Curtailing Uplink Transmission and Energy Requirements in Mobile Free-Viewpoint Video Applications

Clifford De Raffaele and Carl James Debono
Department of Communications & Computer Engineering
University of Malta
Msida, MSD 2080, Malta
cderaffaele@ieee.org and c.debono@ieee.org

Abstract - **The rapid deployment of multimedia services on mobile networks together with the increase in consumer demand for immersive entertainment have paved the way for innovative video representations. Amongst these new applications is free-viewpoint video (FVV), whereby a scene is captured by an array of cameras distributed around a site to allow the user to alter the viewing perspective on demand, creating a three-dimensional (3D) effect. The implementation on mobile infrastructures is however still hindered by intrinsic wireless limitations, such as bandwidth constraints and limited battery power. To this effect, this paper presents a solution that reduces the number of uplink requests performed by the mobile terminal through view prediction techniques. The implementation and performance of four distinct prediction algorithms in anticipating the next viewpoint request by a mobile user in a typical FVV system are compared and contrasted. Additionally, each solution removes the jitter experienced by the user whilst moving from a view pattern to another by allowing some hysterisis in the convergence signal. Thus, this technique enhances the performance of all the algorithms by taking into consideration the fact that the user adapts to the presented views and will react accordingly. Simulation results illustrate that an uplink transmission reduction of up to 96.7% can be achieved in a conventional FVV simulation scenario. Therefore, the application of prediction schemes can drastically reduce the mobile terminal's power consumption and bandwidth resource requirements on the uplink channel.**

Keywords - Free-Viewpoint; Multiview video; Prediction algorithms; Wireless transmission.

I. INTRODUCTION

Video streaming solutions have experienced endless development in the course of time, from a relatively poor image quality, as far as the human senses are concerned, to forms of presentation which strive to present an increasingly better quality of experience [2], [3]. Driven by technological developments, the drastic reduction in the cost of imaging hardware equipment, together with the intensification of clientele expectations, the interest in interactive multiview services has augmented from academic and industrial perspectives alike [4]–[6]. Multiview video technologies provide the potential for innovative applications to be developed in order to facilitate and enhance the experience of scenes from a 3D perspective without the burden of restricting hardware [7], [8]. The latter feature, aids the commercial implementation of such a technology, especially in the end user entertainment market, by systems such as Free-Viewpoint television (FTV) [9].

Free-Viewpoint Video (FVV) provides the potential to expand the viewers experience far beyond what is presented by current conventional multimedia systems [10]. In such an innovative visual media technology, the user can observe a three-dimensional panoramic scene by freely changing perspective [11], [12]. Acquisition techniques for the realization of FVV entail a unique scene captured from multiple views via the deployment of a number of cameras distributed around the site, [13], as portrayed in Fig. 1. The hardware needed for multi-camera systems and for displays is rapidly developing, with new solutions being experimentally deployed [7], [9], [14], and [15].

Architectures that utilize a dense scene capturing framework can render a more complete FVV experience to the user. Nevertheless, feasibility constraints demand that the amount of scene sampling hardware employed is restricted [16], as this presents a linear increase in raw video data that necessitates processing [17]. In addition, spatial proximity of cameras must adhere to the physical limitations of the equipment [18]. Thus, the sole method for FVV viewers to observe a video stream of uninterrupted standpoints without being constrained to the actual camera locations is by the adoption of Video Based Rendering (VBR) techniques [19]. These methodologies apply computationally intensive Intermediate View Rendering (IVR) algorithms to synthesize virtual viewpoints between actual cameras as illustrated in Fig. 1. This process is therefore compulsory to provide gradual view changes in perspective from one actual camera location to another and its quality is directly responsible for the vividness of FVV experience [20].

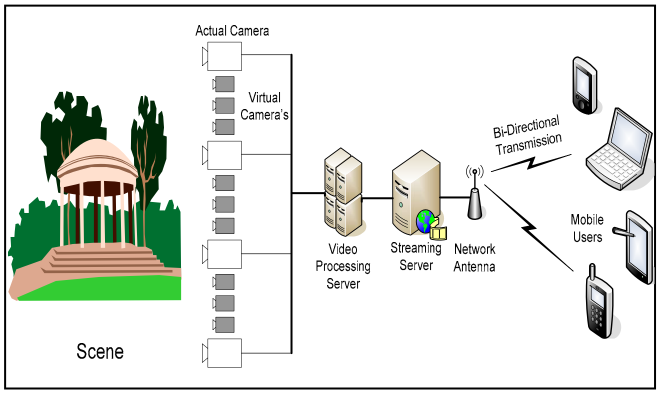

Figure 1. Block diagram of the entire system representing the scene being captured by multiple cameras, video-based rendering servers, the streaming server, and the wireless network.

Simultaneous to the advancements registered in multimedia, the field of mobile computing has witnessed a parallel growth rate. Nowadays, emerging mobile client devices are all fitted with a liquid crystal display screen and sufficient processing power to allow real-time presentation of multimedia information [21]. A similar trend is furthermore witnessed in the infrastructure of emerging and future wireless systems, which provide sufficient bit rates for the implementation of video communication applications [14]. Thus, this opens the way for motion video to become one of the major multimedia applications [6]. Nonetheless, this technology imparts a significant burden on the network infrastructure due to its strict latency requirements and wireless bandwidth restrictions. Furthermore, the harsh wireless transmission environment presents a supplementary range of peculiar technical challenges such as attenuation, fading, multi-user interference and spatio-temporal varying channel conditions [22]. Such issues, together with current business models in wireless systems, whereby the end-user's costs are proportional to the reserved bit-rate or the number of bits transmitted over a radio link [23], drive future multimedia network systems to provide a more efficient framework for the deployment of services.

A further impediment encountered in the implementation of FVV technology is intrinsic to mobile devices, since the latter are severely constrained in energy resources, storage capacity and processing power [24]. The combination of these obstacles makes the transmission of several views for virtual viewpoint rendering at the mobile terminal impractical under several perspectives. Thus, a sensible implementation for FVV is to implement real-time video-based rendering (VBR) techniques on customized processing architectures at the server's side and transmit only the required view to the mobile terminal [25], [26]. Inherently, this strategy demands the implementation of a feedback channel from the mobile device to request the required view perspective stipulated by the user. This necessitates that a request is made on the uplink channel at every video frame interval to request a perspective view. Alas, such a situation leads to substantial service delays, bandwidth usage and terminal power consumption.

This paper builds on [1] to present a solution which reduces the amount of feedback transmissions generated by the mobile terminal during free-viewpoint operation. Several prediction strategies are thoroughly investigated for adoption on the multimedia server to forecast the ensuing user's view request. The algorithms necessitate that feedback uplink packets are only sent when the received perspective does not match the users demand viewpoint. Hence, the server interprets the lack of feedback as a confirmation that the correct estimate was transmitted. If a feedback packet is received, the server is notified of the viewpoint prediction error, at which point the algorithm is retrained to converge to the new FVV pattern. Such a strategy presents the infrastructure with a reduced amount of transmission from the mobile terminals, which in turn preserves their battery power and reduces the uplink bandwidth utilization. The algorithms implemented also manage to minimize the round-trip delays incurred by the system due to the otherwise continuous transmissions on the uplink channel.

This paper is organized as follows; the prediction algorithms studied in this work are examined and discussed in Section II together with the details on the respective parameters employed. Section III discusses the implementation strategy adopted for each algorithm. Following this, Section IV presents the simulation results and highlights the curtailment in feedback transmission attained by each solution. To further aid understanding, this section also gives a quantitative measure description of the battery power saved. Finally, comments and conclusions are presented in Section V.

II. Viewpoint Prediction Algorithms

The successful implementation of prediction algorithms for system behavior estimation demands the utilization of a subset of the observed readings acquired previously from the system, to construct the original set of data within some pre-defined precision tolerance [27]. Viewpoint prediction is achieved by exploiting a combination of received data from the feedback channel, preceding data predictions, and a priori knowledge of the system's operation methodology. This work studies and compares the performance of the following four algorithms; Least Mean Squares, Kalman Filter, Recursive Mean Square, and Linear Regression, adopted specifically for FVV application in an effort to reduce uplink transmission requests and increase the battery lifetime of mobile terminals.

A. Least Mean Square Algorithm

Originally proposed in [28], the least mean square (LMS) algorithm consists of an iterative process of successive corrections on a weight vector which ultimately lead to the minimum square error between the received and derived signals. The algorithm provides several advantages for implementation in real-time scenarios particularly due to its low computational complexity. Furthermore, inherent to its iterative nature, the algorithm is suitable for slow time-varying environments since it exhibits a stable and robust performance [29].

The general function of the LMS algorithm is defined as:

$$\hat{x}(n) = \sum_{i=1}^{N} w_i(n) \times x(n-i) \qquad (1)$$

where $\hat{x}$ denotes the estimated input signal n, w_i represents the current system weight vector, and $x(n – i)$ corresponds to the set of delayed inputs.

In order to derive the prediction error, *e(n)*, the estimated value is subtracted from the real input value *x(n)*:

$$e(n) = x(n) - \hat{x}(n) \qquad (2)$$

The system weight vector is then modified using:

$$w(n+1) = w(n) + \mu x(n)e(n) \qquad (3)$$

As can be seen in (3), the rate of adaptation of the algorithm is directly dependent on the step size μ, which influences the speed of convergence, and on the order of the algorithm *N*. The computational complexity for executing each iteration can be summarized as *2N+1* multiplications and *2N* additions. After simulation trials, a value of five filter weights was deemed apt, as this was able to sufficiently restrain the processing load whilst still providing sufficient precision as to avoid excessive overshoots whilst the filter weight vectors were converging to the reference signal. The values for learning rate variable μ were determined through heuristic techniques. This parameter was furthermore adapted during run-time by initially assigning a large value to speed up convergence towards the reference signal, and then decrease it to a more temperate one, to reduce overshoot and allow more precise corrections as the weight vectors approach convergence.

B. Kalman Filter Algorithm

The Kalman Filter introduced in [30] is primarily a recursive algorithm notably suitable to address the estimation problem for linearly evolving systems [31]. Apart from its practical demands being apposite for real-time applications, a convergence to a stable steady state is guaranteed by the filter [32].

Prediction of the forward state is performed by multiplying the current system state x_k with the state transition matrix *A* as illustrated in (4):

$$\hat{x}_{k+1} = A \times x_k \qquad (4)$$

The Kalman filter also forecasts the predicted error covariance and makes use of this value together with an observation matrix *H* to compute the Kalman gain *K*. Following the acquisition of a measurement of the system output z_k, this is used to update the estimate value x_{k+1} using:

$$x_{k+1} = \hat{x}_{k+1} + K\left(z_k - H\hat{x}_{k+1}\right) \qquad (5)$$

The obtained positional measurement of the scene is compared with the previous value, and since the measurements are done within constant time intervals, the view change request rate can be calculated. This new value is employed to update the input vector x_{k+1}. Finally, the error covariance is amended for the next iteration.

C. Recursive Least Square Algorithm

The Recursive Least Square (RLS) algorithm aims at achieving the minimization of the sum of squares difference between the modeled filter output and the desired signal [33] by calculating the optimum filter weights. This is attained using an exponentially weighted estimate of the input autocorrelation and cross-correlation [34]. Owing to the prediction filter nature of the algorithm, the learning mode operation entails an iterative process whereby current weight vectors are used to generate data prediction in the future, and subsequently measurement data is considered as reference for updating the internal weight vectors.

An intrinsic property of the adapted RLS is the capability to pursue fast convergence in time-varying environments even in cases where the eigenvalue spread of an input signal correlation matrix is large [35]. Unfortunately, the latter benefit is achieved at the cost a substantial increase in complexity, which during implementation has an order of $\mathcal{O}(N^2)$ FLOPS per sample, with *N* being the filter length [36], and an increased sensitivity to mismatch is registered in comparison to the structurally similar least square algorithm [37]. When new samples of the incoming signal are received, the coefficient vector updates the solution for the least squares problem in recursive form using [38]:

$$\underline{w}(k+1) = \underline{w}(k) + \underline{g}(k)e(k), \qquad (6)$$

where *w(k)* is the coefficient vector of the adaptive filter at time *k*, *N* is the length of the latter vector, *e(k)* symbolizes the difference between the generated filter output and the desired signal, and *g(k)* represents the Kalman gain. To render the algorithm feasible for real-time applications, and thus reduce the computational cost incurred by the RLS to execute the necessary matrix manipulations upon every epoch, the matrix inversion lemma technique [33] is applied to obtain a simple update for the inverse of the data input equation:

$$P^{-1}(k) = \frac{1}{\lambda}\left[P^{-1}(k-1) - \underline{g}(k)\underline{x}^T(k)P^{-1}(k-1)\right], \qquad (7)$$

where *x(k)* is the input data vector, and $\lambda \in (0,1]$ is called the forgetting factor. This parameter has direct influence on the memory of the algorithm, with the upper bound value of unity assigned to imply infinite memory and is hence only suitable for statistically stationary systems. The algorithm contains no a priori information on the system at initialization, thus, an approximate initialization technique is employed for the covariance matrix by setting $P^{-1}=\delta I$, which is representative of a scaled version of the identity matrix [38].

D. Linear Regression Algorithm

The statistical method defined by the Linear Regression (LR) algorithm is capable of modeling the relationship between two or more variables, by deriving a linear equation to fit the observed data using a least squares metric [39]. The implementation of a linear model provides several advantages for a real-time system, particularly due to its computational simplicity and inherent ease of use [40]. The validity of this statistical technique is held under the assumption that the output has a linear dependence on the input [41], thus the system model can be composed in the form:

$$\hat{Y} = \beta_1 \times X + \beta_0 + \varepsilon, \tag{8}$$

where the output predicted value $\hat{Y}$ is expressed for a given dependent variable input *X*. The additional amount ε represents the residual error from the regression line, whilst the variables in the model function β_0 and β_1 are referred to as the model parameters and are estimated from a training set of *n* observations, in the form of $(X_1,Y_1),(X_2,Y_2),\ldots,(X_n,Y_n)$, using [42]:

$$\beta_1 = \frac{\sum_{i=1}^{n}(X_i - \overline{X})\times(Y_i - \overline{Y})}{\sum_{i=1}^{n}(X_i - \overline{X})^2}, \tag{9}$$

$$\beta_0 = \hat{Y} - b_1 \times \overline{X}, \tag{10}$$

where the statistical values $\overline{X}$ and $\overline{Y}$ represent the means of the respective variables.

Equations (9) and (10) are only computed whilst the algorithm is operating in the training phase so that the model functions converge. Once the aforementioned parameters are established, the algorithm operates in offline mode, and the computational load for executing the LR algorithm on the server, is only that of computing a single multiplication and two additions to derive the predicted output.

III. Implementation of the Prediction Algorithms

The prediction techniques studied attempt to reduce the amount of feedback transitions necessary on the uplink channel. This is performed by combining a priori information regarding the system implementation together with dynamic data of the user's previous viewing motion to forecast the user's future observation points.

The characteristics of each algorithm are scrutinized and their implementation for the specific scenario presented in FVV is analyzed. The various parameters demanded by either technique are optimized in respect of the unique features of the FVV structure by applying heuristic approaches or taking into consideration the intrinsic nature of the system.

A. Least-Mean Square Algorithm

The LMS algorithm was implemented in a dual topology configuration were two identical adaptation algorithms were executed simultaneously on the mobile terminal and server alike. Implementation of the system involved the adoption of the LMS algorithm to estimate the view required by the user. The prediction of the desired perspective is achieved from interpretation of the preceding readings as well as the adapted LMS weight coefficients. Via this implementation, both sides of the network are able to generate equivalent estimates that keep the system synchronized whilst adapting to the dynamic user perspective demands.

At the mobile terminal, a comparison between the local predicted and the actual input readings is executed iteratively yielding a dynamic error assessment. When the current error in prediction exceeds a pre-defined tolerance threshold, the mobile terminal interprets the situation as a change in the user's input pattern. Although, by adapting the learning rate parameter during execution, the system quickly re-aligns its weight vectors to the new input pattern, the inconsistencies of the weight vectors during the conversion still allow for a suitable error signal to be detected by a comparator; hence triggering a change in the state of the network. In this situation, the mobile terminal attempts to re-synchronize its LMS algorithm with that of the server by training both filters for twenty iterations. During this period of online operation, the current user input pattern is considered as the reference signal, and transmitted to the server such that the weight vectors of both LMS algorithms converge to the new pattern.

Subsequent to the elapse of the training phase, both systems are turned offline again, whereby the reference signal for the local LMS algorithm is taken to be the former predicted value. During this state, provided that no new feedback from the mobile terminal is received at the server, the server assumes that its estimate is correct and thus transmits the predicted view. Via this methodology, the server is capable of tracking the predictions done by the mobile terminal without the requirement of constant transmission requests to update the current viewpoint state.

B. Kalman Filter Algorithm

The Kalman Filter system topology involved the implementation of the algorithm solely on the network server. The filter toggles between an online state and a stand-alone mode during operation. Initially, the mobile terminal commences by transmitting on the uplink channel a training sequence in order to converge the filter's output, computed at the broadcasting server node, to the pattern being viewed by the user. Following this initialization epoch, the algorithm converts its operation mode to a stand-alone state. In this form, the Kalman filter computes the prediction algorithm by utilizing the previous state vector values and error covalence as the observed measurement. In this way, the server forecasts the view number that would be demanded by the user and transmits the respective video perspective to the mobile terminal.

A comparison between the predicted and the actual input reading is executed on the mobile terminal during every iteration, yielding a dynamic error assessment. The latter value determines the state in which the system will operate by reference to a threshold error value. When the error value exceeds this limit, the mobile terminal transmits the first packet of data on the uplink, and subsequently the system implemented on the server node moves to online mode. In this training condition, the Kalman filter receives a sequence of the current user inputs for the desired viewpoint sequence as its reference signal from the mobile terminal via the feedback channel. An array of fifteen values is needed by the Kalman filter to converge to the new pattern of views. When the pre-defined amount of training iterations elapses, the system is redirected to offline stand-alone mode, where no information is required from the mobile terminal, thus reducing the feedback transmissions. In this mode, unless a transmission is received at the server, the previous prediction is considered as correct, and this computed value adopted as the observed measurement for the following epoch.

C. *Recursive Least Square Algorithm*

Similar to the Kalman filter infrastructure, the RLS system involves the adaptation of the algorithm on the server node to track the user demands for FVV viewing. Each time a discrepancy between the received view and the user's demanded one is noted, a training set composed of eight samples is transmitted from the mobile terminal. Upon receiving the initial feedback packet, the server turns the RLS algorithm in online mode and commences a training routine. A recursive approach is utilized during this period by the algorithm to adapt to the new linear viewing pattern. Following the successful convergence, the RLS algorithm is turned back offline, and the server predicts the views that will be demanded by the user using the algorithm's compiled values.

D. *Linear Regression Algorithm*

To track the FVV viewpoint user demands, the linear regression algorithm was also implemented solely on the server node. Due to the inherent nature of the FVV system operation, the algorithm was limited to a linear first-order model, to reduced complexity whilst still providing reliable accuracy in the regression line generated. Since observations done at the mobile terminal are inputted directly by the user, there are no sources of error during the acquisition of information. This feature was exploited to curtail the training set of measurements used by the algorithm to only two samples, which is the minimum amount of observations required for the LR algorithm to produce a robust regression line.

After training, the mobile terminal simply checks whether the received view matches that demanded by the user. If the prediction is correct, the mobile terminal refrains from transmitting feedback, and the server assumes that the estimate it has calculated is correct. Otherwise, feedback view information is delivered to the server which will wait for the second data packet to be transmitted before restarting the execution of the algorithm.

IV. Simulation and Results

Free-Viewpoint Video systems provide the user with the ability to autonomously decide upon the perspective that is observed in a particular scene. This level of interactivity intrinsically implies that a substantial amount of information is exchanged between the user and the host of the system. In the mobile video applications scenario, such data requirements entail bi-directional communication between the mobile terminal and the network server on the wireless infrastructure. Establishing such an infrastructure, demands bandwidth utilization to support both the streaming video sequences on the downstream as well as viewpoint requests on the uplink channel.

A. *Simulation Overview*

To objectively simulate and analyze the employment of the free-viewpoint structure embedded by the prediction algorithms, a typical situation was modeled using the Maltab$^{®}$ platform for two separate FVV usage profiles. It is assumed that the FVV user is not consistently changing views at a fast rate. Such a scenario is not practical as the user will not be able to follow the content of the video. Thus, our system considers the feasible implementation whereby users alter viewing patterns at a practical rate. The FVV system considered consisted of a number of adjacent cameras, which allowed the rendering of nine distinct virtual views in between each actual capturing location. These virtual views were generated on the server node and provided to the users upon demand, to enhance the FVV experience attained when altering the viewpoint between two locations by providing a gradual transition. The mobile terminal employs its user interface to display the video streams as well as receive input from the client as regards the viewpoint request. The latter is considered to be in the form of a vertical slider which allows the user to shift his view perspective between a finite number of cameras with an unrestricted range of motion speed as illustrated in Fig. 2.

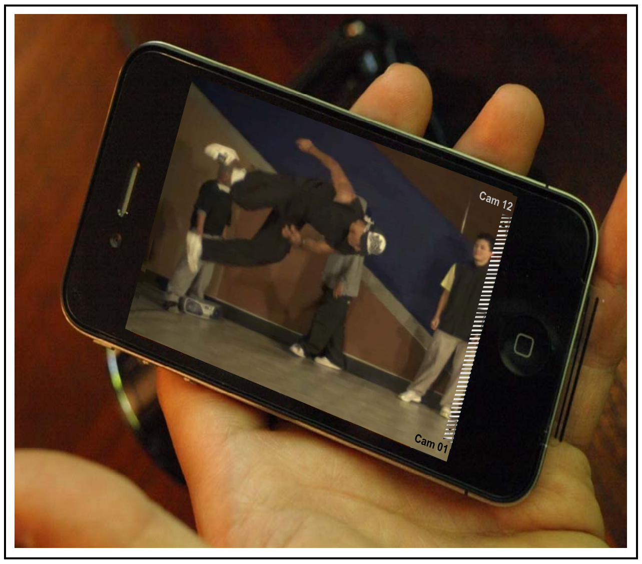

Figure 2. End-user mobile interface for free-viewpoint video streaming.

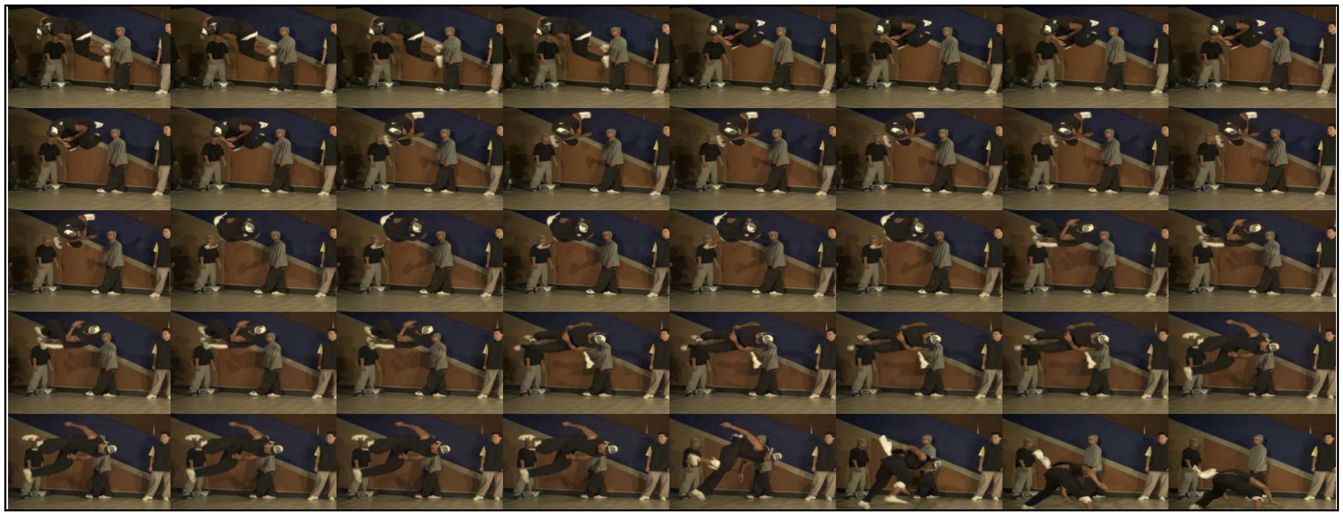

Figure 3. Linear free-viewpoint video motion from one perspective to another via consequent intermediate frames [43].

The simulated scenarios start with the user receiving a view from a single camera and not requesting any view changes. At an arbitrary point in time, the user starts changing the viewing angle at a particular velocity, thus performing free-viewpoint operation. Subsequently, this motion pattern will change casually, with each epoch having a different rate of change and duration of the varying FVV perspective. This results in a linear motion as illustrated in the video sequence shown in Fig. 3, whereby the change of perspective is gradual from one camera outlook to the next, and this can be modeled with lines of constant gradient [43].

The FVV simulated usage profiles can be described as: (i) pattern A which shows a velocity pattern in which free-viewpoint motion can be either static or else change with at least one virtual view per time step, and (ii) pattern B which shows a pattern whereby the user can scroll through views more leniently and hence take more than one time step to change from a virtual standpoint to a neighboring one. Due to the quantized nature by which view changes can be requested, the reference pattern generated by a user making requests similar to pattern B can introduce a jittered pattern of motion as depicted in Fig. 4. In this situation, the view will remain constant for a small period of time and then alter by the minimum quantized amount, giving the impression of a slow changing perspective. Thus, the speed for this scenario is proportional to the number of time steps in which the view remains constant between changes.

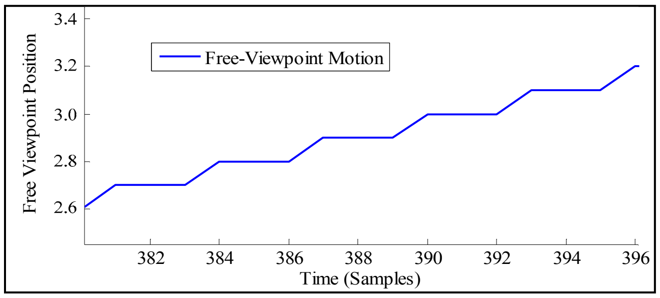

Figure 4. Free-viewpoint pattern which is slower than one virtual perspective per time frame

The performance evaluation of the proposed algorithms is compared to the system specifications described by the reference FVV architecture found in [25]. This reference methodology, whose view request profiles for patterns A and B are shown in Fig. 5(a) and Fig. 6(a) respectively, portrays the operation of an FVV system where the mobile terminal transmits to the server in each time step to demand the view being requested by the user. The server processes the requested viewpoint and transmits the perspective back to the terminal. The image from the modified outlook would then be viewed on the mobile terminal in the subsequent video time frame.

In order to further improve the FVV algorithms described in [1], a modified request pattern is employed at the server node for the prediction filters to converge too. This technique is adopted to eliminate the jitter experienced by the user during the first video-frame of a new FVV motion pattern. The utilized pattern achieves this by allowing some hysterisis on the signal, which results in delaying the user's request by a constant view separation. The latter is derived from the prediction errors generated by any of the algorithms upon motion pattern alteration, whereby for a particular time step, the received frame and that demanded by the user are not synchronized. With this enhanced approach, instead of converging directly to the user pattern, and hence yielding a nuisance jitter in FVV motion, the system, takes into consideration the video frames already viewed at the mobile terminal, and converges the algorithms to the modified signal, which directly reflects the same motion pattern evoked by the user's input. This amendment is able to significantly improve the quality of experience delivered to the client, by feasibly considering the implementation scenario and hence striving to smooth the free-viewpoint motion video received. Furthermore, the small delay registered is only of one video frame, usually 40 milliseconds, and thus goes unnoticed by the user who will naturally adapt to the viewed perspective pattern.

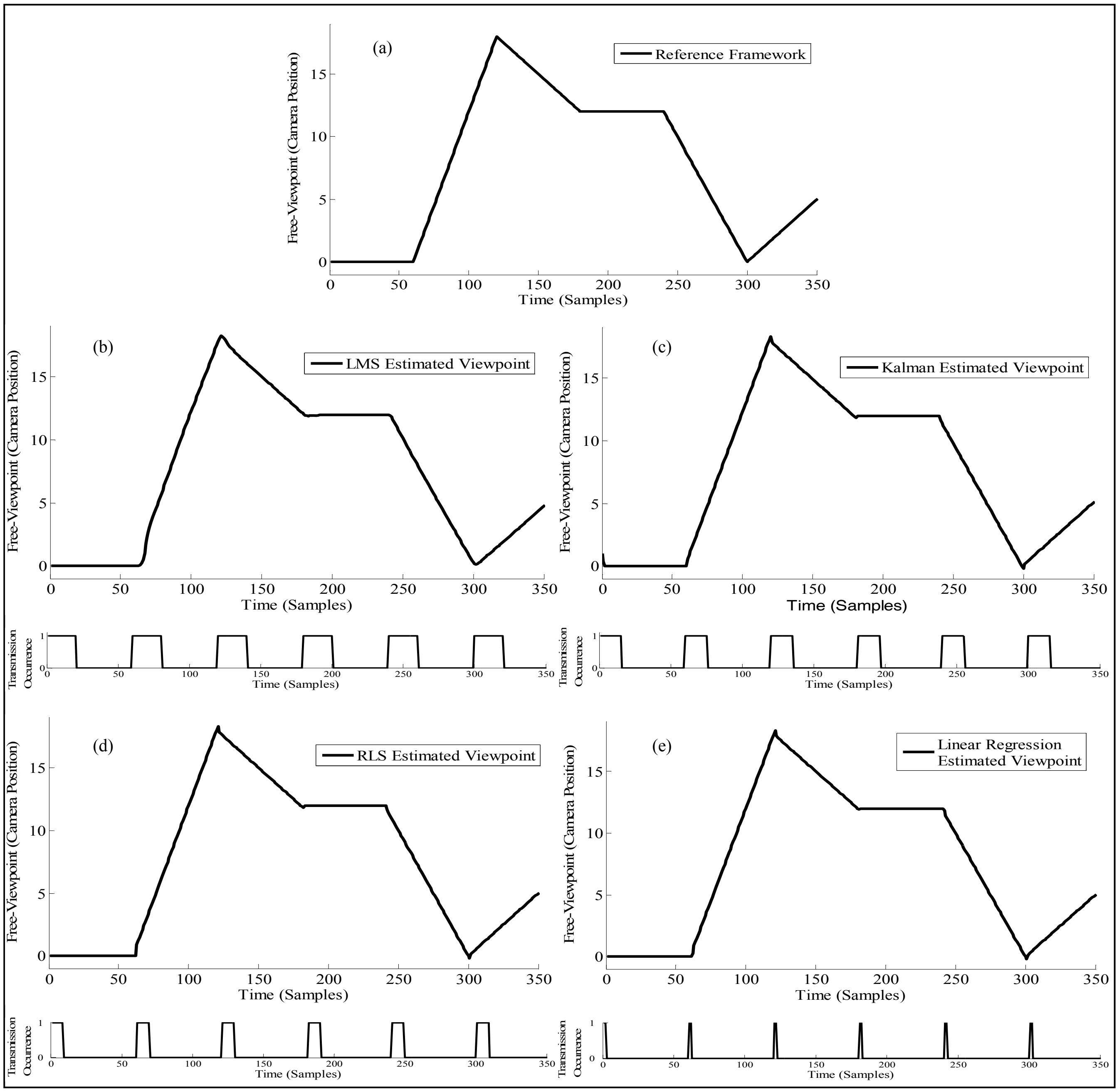

Figure 5. Simulation results and the associatae transmission occurrences comparing the proposed algorithms in a typical FVV scenario for pattern A: (a) reference user input, (b) Least Mean Squares, (c) Kalman Filter, (d) Recursive Least Squares, (e) Linear Regression.

B. *Analysis of Pattern A*

The simulation of the LMS for pattern A is presented in Figure 5(b). The inherent slow convergence speed of the algorithm is evident in the simulation results. The transmission occurrence signal, beneath the same figure, indicates that when the input view pattern was altered, the converged weights lost validity and hence yielded an error in the predicted viewpoint thereby re-initiating the training phase. Even though the learning rate parameter was adaptively modified in the initial stage of training, a certain amount of time steps have to elapse before the algorithm's internal memory buffers are occupied and all the weight vectors updated accordingly. Thus, the LMS algorithm necessitates the delay of an initial settling time before being employed successfully in FVV applications. Alas, the same compromise made on the parameter μ hinders the maximum frequency at which the system can alter the viewing pattern and the maximum change in gradient the system is able to adhere to.

With respect to the standard reference pattern, the LMS algorithm provides a considerable improvement in feedback transmission reduction. With a minimal increase on the computational resources of both the mobile terminal and the server, an average reduction of 65% in feedback transmissions is obtained. This technique thus reduces the traffic passed over the uplink channel from 350 transmissions to approximately 120 in the considered scenario.

The pattern A results for the Kalman Filter, illustrated in Fig 5(c), implicitly expose the benefits derived from the powerful adaptation of this algorithm. In comparison to the LMS technique, the initial convergence speed is greatly enhanced, drastically minimizing the necessary setup time. The Kalman filter algorithm, although inducing a significantly larger computational cost at the server, relieves the execution cost from the mobile terminal, which must only perform a comparison between the view perspectives received and those demanded by the viewer. The prediction properties of the latter methodology are also more robust requiring a set of only fifteen training samples to adapt to the new pattern with a significantly reduced error such that offline operation is sustained. This enables the system to sustain a higher frequency of change in view pattern in a stable manner. For the observed scenario, the Kalman filter solution results in an uplink transmission reduction of 74%.

Implementation of the Kalman filter on the server node is nonetheless a computationally complex task which poses imminent scalability issues. On the other hand, transmission occurrence was considerably reduced with respect to the LMS algorithm as this parameter inferred directly on the achievement of adopting prediction techniques. A compromise between these algorithms was found by the implementation of the RLS algorithm. The latter is able to adopt the advantages of the Kalman filter's convergence speed and robustness whilst attaining a reduction in implementation complexity.

The simulation results for the RLS algorithm on Fig. 5(d) illustrate that although the filter length was pruned to five as to conserve the memory footprint, convergence towards the desired output is still achieved at a satisfactory speed. In comparison to the LMS algorithm, the RLS implementation still requires a small number of training samples to be processed to fill the weight vector with relevant values of the new pattern. Nonetheless, the RLS algorithm can achieve a much quicker convergence to the reference signal as is evident from Fig. 5(d), where substantial convergence is achieved after the first couple of epochs. This benefit is mainly derived from the manipulation of the forgetting factor λ, which by means of adaptive variations can concentrate more heavily on the new input measurements whilst allocating less influence to the irrelevant values. The recursive nature of the algorithm also aids to provide a stable convergence, as was a principle advantage in the Kalman Filter. The same feature however, presents the need for an additional amount of samples to be computed prior to running the RLS algorithm in offline mode. Nevertheless, this technique allows the algorithm to converge successfully after processing only eight training samples. Over the considered typical input pattern scenario, the RLS is able to yield an efficiency gain of 87% over the standard architecture defined in [25], whilst requiring a notably reduced computational cost with respect to the Kalman Filter implementation.

The final investigated solution employs the Linear Regression algorithm on the server to predict the desired view sequences requested by the user. The results in Fig. 5(e) implicitly expose the benefits derived from the adaptation characteristics of this algorithm. Instead of attempting to predict the next viewpoint, this LR implementation predicts the velocity by which the viewpoints are changing between defined video time steps, thus generating a regression line which indicates the rate of change demanded by the user. When comparing the results of the LR algorithm to those attained by the previous filter algorithms for the pattern A, it is evident that the Linear Regression considerably outperforms the former. The algorithm converges faster than its counterparts to the desired signal output, since it takes direct advantage of the linear nature in which the FVV viewpoint is altered. Hence for the same scenario, the LR algorithm registered a decrease in the transmission requirements by the mobile terminal of 96.5% compared to the standard, and requires only 12 transmissions throughout the 350 time steps.

C. *Analysis of Pattern B*

Moreover, the performance enhancement of the LR algorithm is even further pronounced in the scenario expressed with input pattern B. This usage profile, visualized in the simulation results of Fig. 6, depicts the futile efforts performed by the LMS, Kalman Filter and RLS algorithms in Fig. 6(b), Fig. 6(c) and Fig. 6(d) respectively. The latter struggle to adapt to the stepped motion pattern of this scenario, resulting in a large amount of prediction failures and subsequent re-initiation of training epochs. The periodic gradient change presents the filters with an input sample incoherent with the previous stationary samples, yielding the algorithms to incorrectly update their weight vector.

The LR algorithm however, successfully manages to cope with the demand for slow free-viewpoint movement that is presented by pattern B. As evidenced in Fig. 6(e), when the algorithm detects a unitary gradient change subsequently followed by a stationary request, the implementation keeps track of the amount of time steps that the user spends on a fixed viewpoint between view changes by means of a dedicated counter. This information is used once an error is retransmitted by the mobile terminal, at which point the linear regression parameters are computed taking into account the new position and the previous samples. Since the system's FVV viewpoints are quantized in nature, the regression line generated performs well as the output is truncated to the resolution of the views. Hence, the algorithm is able to successfully predict any user's input pattern after a maximum of two error signals.

A consistent performance gain is also registered with the linear regression algorithm in the more flexible scenario represented by pattern B, were on average, the uplink transmission reduction achieved amounted to 92%.

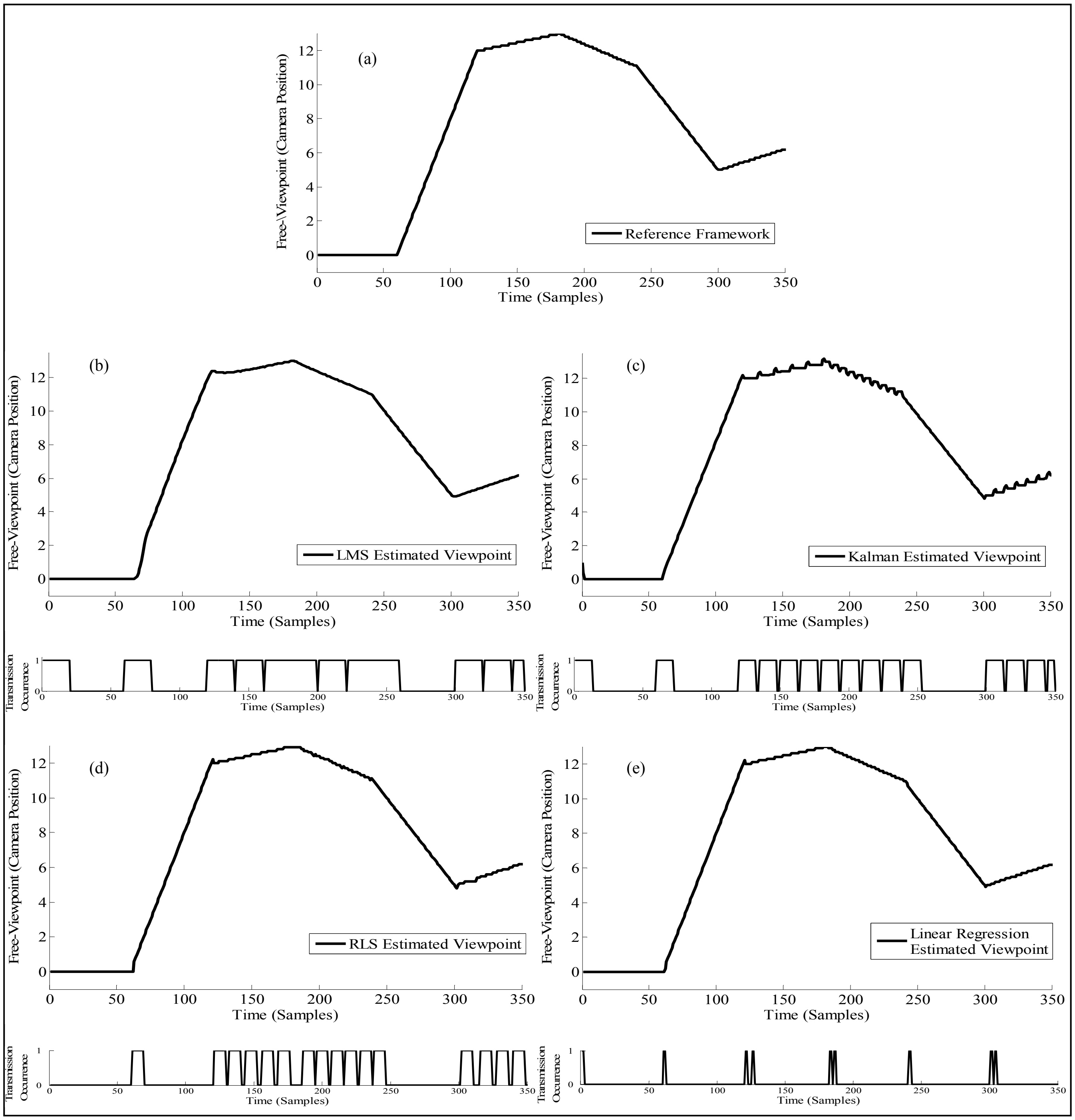

Figure 6. Simulation results and the associatae transmission occurrences comparing the proposed algorithms in a typical FVV scenario for Pattern B: (a) reference user input, (b) Least Mean Squares, (c) Kalman Filter, (d) Recursive Least Squares, (e) Linear Regression.

The small discrepancy between both scenarios adopting the LR algorithm is due to the additional transmissions required to cater for the slow movements in viewpoint patterns. Furthermore, since the LR algorithm necessitates only of a small amount of training samples to converge, this methodology is also able to better accommodate highly dynamic view pattern changes.

D. Battery Consumption Simulation

To obtain a quantitative analysis of the energy resources saved at the mobile terminal by the implementation of the proposed prediction algorithms, a simulation of the energy consumed by the transceiver operation of a mobile station employed only for FVV usage was performed. Although the proposed system is able to operate over any networking topology, a scenario involving a currently implemented wireless architecture was simulated so as to attain appreciation of the algorithm's performance.

The mobile device is considered to be equipped with a 1500mAh battery and consumes an average current of 25mA during transmission and an average current of 20mA during reception of data. The considered scenario is that of a terminal operating in a network employing HSPA technology with a downlink and uplink bandwidth of 7.2Mbits/s and 1.4Mbits/s respectively. The free-viewpoint operation requires a single packet of 50 bytes to send view requests and a mean of four 200 byte packets on the downlink channel. The transmitted data constitutes a continuous video stream in CIF standard resolution employing the H.264/AVC baseline profile.

The drastic reduction in the necessary transmissions on the uplink channel achieved by the employment of view prediction algorithms significantly reduces the battery power consumed by the mobile terminal during FVV operation, as shown in Fig. 7. Even though the amount of data which is transmitted on the uplink channel is much smaller than that of the received video, the energy saved is still significant for a resource limited device. This considerable drop in energy consumption occurs because the transmitter module is the most power hungry component of any mobile terminal. Furthermore, the prediction solutions enhance the Quality of Service (QoS) provided by the FVV system. This occurs because the algorithms offer a pro-active network implementation by predicting and providing the required view perspective to the user instantaneously, thus reducing the round-trip delays incurred by the reference method.

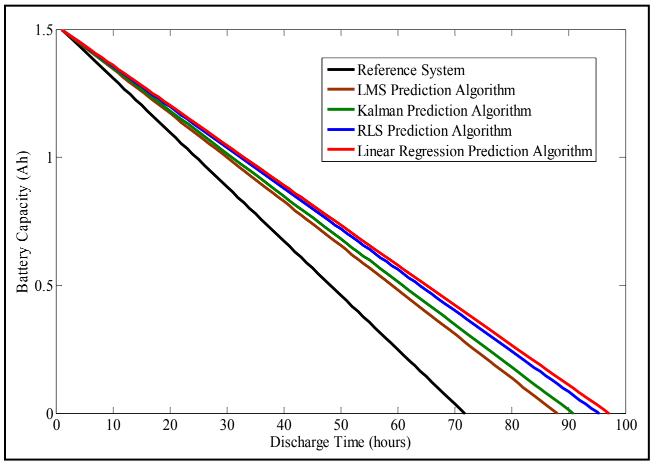

Figure 7. Comparison of the battery discharge time on a mobile terminal when adopting the prediction algorithms

V. CONCLUSION

This paper has presented a detailed study in the adaptation of prediction algorithms to free-viewpoint video technology to reduce the amount of uplink transmission. Four distinct algorithms were analyzed and implemented in several simulation scenarios representing typical FVV architectures. The minor increase in computational costs incurred at the server and/or mobile node are justified by a drastic reduction of up to 96.7% in the amount of feedback packets transmitted on the wireless uplink channel. The requests from the mobile client demanding a different perspective of the scene in a typical FVV usage profile are replaced by the server's predictions through one of the prediction algorithms. The benefits achieved from such systems enable a considerable gain in terms of power conservation for the multimedia mobile terminal as well as a reduced utilization of the uplink bandwidth. Moreover, hysterisis is introduced in the algorithm's converging pattern, making the real-time experience more pleasing and avoiding any video jitter being presented to the mobile client.

REFERENCES

[1] C. De Raffaele and C.J. Debono, "Applying Prediction Techniques to Reduce Uplink Transmission and Energy Requirements in Mobile Free-Viewpoint Video Applications," in Proc. of the IARIA 2010 Second International Conference on Advances in Multimedia (MMEDIA 2010), pp. 55-60, Jun 2010.

[2] C. Fehn, P. Kauff, M. Op de Beeck, F. Ernst, W.I. Jsselsteijn, M. Pollefeys, L. Van Gool, E. Ofek, and I. Sexton, "An Evolutionary and Optimised Approach on 3D-TV," in Proc. of Int. Broadcast Conf., pp. 357–365, Amsterdam, Netherlands, 2002.

[3] J. Lim, J. Kim, K.N. Ngan, and K. Sohn, "Advanced Rate Control Technologies for 3D-HDTV," in IEEE Trans. on Consumer Electronics, vol. 49, no. 4, pp. 1498-1506, Nov. 2003.

[4] S.U. Kum, and K. Mayer-Patel, "Intra-Stream Encoding for Multiple Depth Streams," in Proc. of the 16th Int. Workshop on Network and Operating Systems Support for Digital Audio and Video, pp. 62–67, Newport, USA, May 2006.

[5] S. Tan, A. Aksay, C. Bilen, G.B. Akar, and E. Arikan, "Error Resilient Layered Stereoscopic Video Streaming," in Proc. of the Int. Conf. on True Vision Capture, Transmission and Display of 3D Video, pp. 1–4, Kos Island, Greece, May 2007.

[6] A. Kubotka, A. Smolic, M. Magnor, M. Tanimoto, T. Chen, and C. Zhang, "Multiview Imaging and 3DTV," in IEEE Signal Process Mag., vol. 24, no. 6, pp. 10–21, Nov. 2007.

[7] J. Konrad and M. Halle, "3-D Displays and Signal Processing," in IEEE Signal Processing Mag., vol. 24, no. 7, pp. 97–111, Nov. 2007.

[8] H. Kalva, L. Christodoulou, L. M. Mayron, O. Marques, and B. Furht, "Design and Evaluation of a 3D Video System Based on H.264 View Coding," in Int. Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV), pp. 12–12, May 2006.

[9] T. Fujii, and M. Tanimoto, "Free-Viewpoint TV (FTV) System," in Proc. of Pacific Rim Conf. on Multimedia, pp. 497-504, Tokyo, 2004.

[10] A. Smolic, H. Kimata, and A. Vetro, "Development of MPEG Standards for 3D and Free Viewpoint Video," in Proc. of the SPIE Conf. Optics East 2005: Communications, Multimedia & Display Technologies, vol. 6014, pp. 262–273, Nov. 2005.

[11] M. Tanimoto, M.P. Tehrani, T. Fujii, and T. Yendo, "Free-Viewpoint TV," in IEEE Signal Process Mag., vol. 28, no. 1, pp. 67–76, Jan. 2011.

[12] X. Guo, Y. Lu, F. Wu, W. Gao, and S. Li, "Free Viewpoint Switching in Multi-View Video Streaming using Wyner-Ziv Video Coding," in Proc. of SPIE Visual Communications and Image Processing 2006, vol. 6077, pp. 298–305, California, USA, Jan. 2006.

[13] A. Smolic, and P. Kauff, "Interactive 3D Video Representation and Coding Technologies," in IEEE Special Issue on Advances in Video Coding and Delivery, vol. 93, no. 1, Jan. 2005.

[14] J. Berent and P. Luigi Dragotti, "Plenoptic Manifolds," in IEEE Signal Processing Mag., vol. 24, no. 7, pp. 34–44, Nov. 2007.

[15] A. Vetro, W. Matusik, H. Pfister, and J. Xin, "Coding Approaches for End-to-End 3D TV Systems," in Proc. of the 23rd Picture Coding Symposium (PCS '04), pp. 319–324, San Francisco, California, USA, Dec. 2004.

[16] C. Zhang, L. Huo, C. Xia, W. Zeng, and W. Gao, "A virtual view generation method for free-viewpoint video system," in Proc. of the 2007 Int. Symp. on Intelligent Signal Processing and Communication Systems, Xiamen, 2007, pp. 361–364.

[17] S.U. Yoon, E.K. Lee, S.Y. Kim, and Y.S.Ho, "A Framework for Multi-View Video Coding using Layered Depth Images," in Proc. of the Pacific Rim Conf. on Multimedia, pp. 431–442, Jeju Island, Korea, 2005.

[18] J. Carranza, C.M. Theobalt, M. Magnor, and H.P. Seidel, "Free-Viewpoint Video of Human Actors," in ACM Trans. on Graphics, vol. 22, no. 3, pp. 569–577, Jul. 2003.

[19] V. Nozick and H. Saito, "On-Line Free-Viewpoint Video: From Single to Multiple View Rendering," in Int. Journal of Automation and Computing, vol. 5, no. 3, pp. 257–267, Jul. 2008.

[20] J. Starck, J. Kilner and A. Hilton, "Objective Quality Assessment in Free-Viewpoint Video Production," in Proc. 3DTV Conf.: The True Vision – Capture, Transmission and Display of 3D Video, pp. 225–228, Istanbul, 2008.

[21] T. Stockhammer, and M.M. Hannuksela, "H.264/AVC Video for Wireless Transmission," in IEEE Trans. Wireless Communications, vol. 12, pp. 6–13, Aug. 2005.

[22] T. Stockhammer, M.M. Hannuksela, and T. Wiegand, "H.264/AVC in Wireless Environments," IEEE Trans. Circuits and Systems for Video Technology, vol. 13, pp. 657–673, Jul. 2003.

[23] T. Stockhammer, and T. Wiegand, "H.264/AVC for wireless applications," in Proc. of the IEEE Inter. Workshop on Mobile Multimedia Communications, Munich, 2003.

[24] H.K. Arachchi, and W.A.C. Fernando, "H.264/AVC in mobile environment: a review," SPIE Optical Engineering, vol. 44, pp. 097006.1–097006.7, Sep. 2005.

[25] J. Kwon, M. Kim, and C. Choi, "Multiview video service framework for 3D mobile devices," in Proc. of the Int. Conf. on Intelligent Information Hiding and Multimedia Signal Processing, pp. 1321–1234, Harbin, 2008.

[26] C.J. Debono and C. De Raffaele, "Minimizing Uplink Data in Wireless Free-Viewpoint Video Transmission Applications," in Proc. of the 2009 Third Int. Conf. on Next Generation Mobile Applications, Services and Technologies (NGMAST 2009), pp. 298-302, Cardiff, September 2009.

[27] C.J. Debono, and N.P. Borg, "The implementation of an adaptive data reduction technique for wireless sensor networks," in Proc. of the IEEE 8th Int. Symp. on Signal Processing and Information Technology, pp. 402–406, Sarajevo, 2008.

[28] B. Widrow, and M.E. Hoff, "Adaptive switch circuits," in IRE WESCON Convention Record, Vol. 55, part 4, pp. 96–104, New York, NY, USA, Aug. 1960.

[29] A. Zhang, and N. Gong, "A Novel Variable Step Size LMS Algorithm Based on Neural Network," in Proc. of the Int. Conf. on Intelligent Systems and Knowledge Engineering (ISKE 2007), Chengdu, China, Oct. 2007.

[30] R.E. Kalman, "A New Approach to Linear Filtering and Prediction Problems", in Trans. of the ASME- Journal of Basic Engineering, Vol. 82, pp. 35–45, Mar. 1960.

[31] S.S. Xiong, and Z.Y. Zhou, "Neural Filtering of Colored Noise Based on Kalman Filter Structure," in IEEE Trans. on Instrumentation and Measurement, vol. 52, no. 3, pp. 742–747, Jun. 2003.

[32] X. Huan, and S. Mannor, "A Kalman Filter Design Based on the Performance/Robustness Tradeoff," in Proc. of the 44th Annual Allerton Conf. on Communication, Control, and Computing, pp. 59–63, Sep. 2007.

[33] G.C. Goodwin, and R.L. Payne, Dynamic System Identification: Experiment Design and Data Analysis. Academic Press, New York, 1977.

[34] P.C. Wei, J.R. Zeidler, and W.H. Ku, "Adaptive recovery of a chirped signal using the RLS algorithm," IEEE Trans. Signal Processing. vol. 45, pp. 363–376, Feb. 1997.

[35] P.S.R. Diniz, Adaptive Filtering: Algorithms and Practical Implementation, 3rd ed. Springer Publishers, New York, 2008.

[36] Y.V. Zakharov, G.P. White, and J. Liu "Low-complexity RLS algorithms using dichotomous coordinate descent iterations," IEEE Trans. Signal Processing, vol. 56, pp. 3150–3161, Jul. 2008.

[37] Z. Tian, K.L. Bell, and H.L. Van Trees "A recursive least squares implementation for LCMP beamforming under quadratic constraint," IEEE Trans. Signal Processing. vol.49, pp. 1138–1145, Jun. 2001.

[38] S. Haykin, Adaptive Filter Theory 3rd ed. Prentice Hall, New Jersey, 1996.

[39] K.B. Song, S.K. Ha, J.W. Park, D.J. Kweon, and K.H. Kim, "Hybrid load forecasting method with analysis of temperature sensitivities," IEEE Trans. Power Systems, vol. 21, pp. 869–876, May 2006.

[40] M.R. Gupta, E.K. Garcia, and E. Chin, "Adaptive local linear regression with application to printer color management," IEEE Trans. Image Processing, vol. 17, pp. 936–945, Jun. 2008.

[41] A. Gu, and A. Zakhor, "Optical proximity correction with linear regression," IEEE Semiconductor Manufacturing, vol. 21, pp. 263–271, May 2008.

[42] N.R. Draper, and H. Smith, Applied Regression Analysis, 3rd ed. John Wiley & Sons Inc., New York, 1998.

[43] C. De Raffaele, "Efficient Transmission of Free-Viewpoint Video Services on Mobile Devices," *M.Sc. ICT Dissertation*, University of Malta, 2010.