A Multiple Model Approach to Track Head Orientation With Delta Quaternions

Henry Himberg, Yuichi Motai, *Member, IEEE*, and Arthur Bradley

***Abstract*—Virtual reality and augmented reality environments using helmet-mounted displays create a sense of immersion by closely coupling user head motion to display content. Delays in the presentation of visual information can destroy the sense of presence in the simulation environment when it causes a lag in the display response to user head motion. The effect of display lag can be minimized by predicting head orientation, allowing the system to have sufficient time to counteract the delay. In this paper, a new head orientation prediction technique is proposed that uses a multiple delta quaternion (DQ) extended Kalman filter to track angular head velocity and angular head acceleration. This method is independent of the device used for orientation measurement, relying on quaternion orientation as the only measurement data. A new orientation prediction algorithm is proposed that estimates future head orientation as a function of the current orientation measurement and a predicted change in orientation, using the velocity and acceleration estimates. Extensive experimentation shows that the new method improves head orientation prediction when compared to single filter DQ prediction.**

***Index Terms*—Delta quaternion (DQ), extended Kalman filter (EKF), head orientation, head tracking, interacting multiple model estimator (IMME), quaternion prediction.**

I. Introduction

VIRTUAL REALITY (VR) and augmented reality (AR) environments using head-mounted displays often use head tracking to determine the "look angle" of the user, controlling virtual objects and symbology to fit the scene that is being viewed. Symbology might include mission critical information such as target range or sensory input. Several studies have investigated the use of VR/AR environments for teleoperation interfaces to improve situational awareness through immersion [7], [20], [30], [37]. The value of VR/AR in systems using head tracking is directly impacted by the degree of user immersion; any perceived lag between head motion and scene response reduces presence in artificial environment [21]. Severe scene lag can disorient the individual, causing dizziness and, in extreme cases, simulation sickness [2], [18], [25], [35], [36], [33]. The scene rendering process in modern VR/AR environments is typically in the range of 50–100 ms, resulting in significant display lag. An effective method of lag compensation is to predict head orientation using head tracking data, rendering the next scene ahead of time.

Head motion is extremely unpredictable, ranging from stationary pose when studying a scene to rapid accelerations and decelerations when tracking moving objects. A single motion model cannot accurately handle all types of head motion, resulting in compromised performance [13], [21]. Multiple model estimation can be used to improve head tracking by combining different motion models that target specific types of head motion.

Multiple model algorithms can be divided into three generations: autonomous multiple models (AMMs), cooperating multiple models (CMMs), and variable structure multiple models (VSMMs) [23]. The AMM algorithm uses a fixed number of motion models operating autonomously. The AMM output estimate is typically computed as a weighted average of the filter estimates. The CMM algorithm improves on AMM by allowing the individual filters to cooperate. The well-known interacting multiple model estimator (IMME) algorithm is part of the CMM generation. The IMME makes the overall filter recursive by modifying the initial state vector and covariance of each filter through a probability weighted mixing of all the model states and probabilities [5], [6], [29]. The IMME approach was shown to improve performance in high-acceleration conditions, but similar to the modified AMM method, it caused larger overshoot (OS) and ringing. The VSMM algorithm builds on the CMM approach by varying the type of models operating in the filter at any given time. Models are dynamically added or deleted from the filter based on their performance, eliminating poorly performing ones and adding candidates for improved estimation.

The contribution of this paper is twofold: First, we propose a new method of estimating angular head velocity and angular head acceleration from orientation measurements, and second, we present a new approach to orientation prediction. This paper proposes the multiple model delta quaternion (DQ) (MMDQ) algorithm for latency compensation. The MMDQ estimates angular head velocity and acceleration from orientation data using an IMME. For this study, we have modified the IMME to improve overall performance by adding provisions to avoid numerical underflow/overflow conditions and an adaptive transition probability matrix (TPM). The MMDQ uses three extended Kalman filters (EKFs) to estimate velocity and acceleration from the change in head orientation expressed as the DQ (Δq). An adaptive prediction algorithm then uses the velocity and

Manuscript received November 8, 2011; revised March 18, 2012; accepted April 8, 2012. This work was supported in part by the School of Engineering at Virginia Commonwealth University, by Polhemus Inc., Colchester, VT, and by National Science Foundation Division of Electrical, Communications and Cyber Systems #1054333.

H. Himberg is with Polhemus, Inc., Colchester, VT 05446 USA (e-mail: hhimberg@polhemus.com).

Y. Motai is with the Department of Electrical and Computer Engineering, Virginia Commonwealth University, Richmond, VA 23284 USA (e-mail: ymotai@vcu.edu).

A. Bradley is with the Langley Research Center, National Aeronautics and Space Administration, Hampton, VA 23681 USA (e-mail: arthur.t.bradley@nasa.gov).

Digital Object Identifier 10.1109/TSMCB.2012.2199311

acceleration estimates to predict future orientation across a user-specified time interval. This method differs from other EKF-based approaches in that the prediction time is not a multiple of the data rate but rather can be matched to display lag without consideration of the data rate. The decoupling of the prediction interval from the orientation measurement rate allows the prediction process to make full use of the faster update rate of modern orientation measurement systems.

This paper is organized as follows. In Section II, the theoretical background for the proposed algorithm is briefly discussed. In Section III, the algorithm implementation is presented in detail. Section IV presents and discusses experimental results. A summary of the performance of the proposed method is presented in Section V.

A. Related Work

Our first study on this subject presented two adaptive approaches to using the EKF for head orientation prediction [14]. The first adaptive method modified the EKF predicted error covariance to improve the tracking performance when head motion changed. Although the tracking performance improved, the fading memory algorithm also resulted in increased noise in the predicted orientation. A second adaptive method (R-adaptive) modified the measurement noise covariance of the EKF in response to the noise level of the predicted orientation. The R-adaptive successfully controlled the output noise level while improving tracking for benign head motion but increased the prediction error with aggressive head motion.

The authors have previously presented the DQ EKF as a new approach to head orientation prediction [15]. The DQ method removes the quaternion orientation from the EKF, significantly reducing the computation requirements. Our study found that the DQ EKF was superior to the well-known quaternion EKF [3], [22] for aggressive head motion but was slightly inferior for moderate head motion.

A modified AMM algorithm was used by Kyger and Maybeck [21] to compensate for latency. Individual filters were designed for look-angle tracking based on first-order Gauss–Markov acceleration, first-order Gauss–Markov velocity, and constant-position models. The three filters ran independently and were reinitialized when divergence was detected. A restart algorithm was added to the AMM framework to keep the individual filter state vectors in the locality of the overall filter state vector, allowing rapid transition between filters as the type of motion changed. The modified AMM filter reduced lag significantly but suffered from increased OS and ringing. The filter used one-step prediction to compensate for latency, thus limiting the frame rate to the required prediction time in the application. Additionally, the approach did not provide complete orientation data, choosing to supply the look angle only. Liang *et al.* developed a head motion prediction method based on Kalman filtering [25]. The proposed system predicted head orientation using a filter model that decoupled the four quaternion components, filtered them independently with separate EKFs, and then recombined them to form the predicted quaternion value. A study of predictive filtering methods was conducted by Rhijn *et al.* [32]. Their work found that the EKF provided the same performance in typical VR/AR applications as other predictive filtering methods, including particle filters and the unscented Kalman filter. Yang *et al.* studied the use of the EKF in single filter and multiple model frameworks for conflict detection algorithms [41]. Their study found that the single Kalman filter provided some advantages during mode transitions but the multiple model approach performed better overall.

Ali *et al.* used DQs to control the attitude in the Mars Exploration Rover [1]. Their approach estimates the change in orientation and then corrects it using measurement data from a variety of instruments, including accelerometers and gyroscopes. Cheon and Kim estimated spacecraft attitude using quaternion orientation [9]. Their work used a magnetometer and a gyroscope to estimate quaternion orientation with an unscented Kalman filter. Marins *et al.* used DQs with Kalman filtering to construct a magnetic, angular rate, and gravity sensor [26]. A study by Sabatini combined a gyroscope, an accelerometer, and a magnetometer to measure orientation for biomedical applications [33]. Each of these studies uses angular rate data to estimate quaternion orientation with Kalman filtering. In our study, we estimate angular rate from orientation data and then use it to predict orientation.

The concept of DQ, which hinges on the idea of building an error quaternion using quaternion composition rather than quaternion subtraction, is at the heart of what is known as the multiplicative EKF (MEKF) [3], [27], [31]. The MEKF has been used not only to estimate the quaternion but also to estimate angular velocities, sensor errors, biases, and alignments.

II. Background

A. DQ Filtering

A quaternion provides a unique representation of orientation when it is normalized to the unit sphere (1) [10], [34], [39]. The DQ filter predicts future head orientation from the change in quaternion orientation, computing the change in orientation (Δq) as a function of the estimated head velocity. To rotate an object, the orientation $q[k]$ of the object is multiplied by the desired change in rotation, i.e., the DQ $\Delta q[k]$ defined as (2). Note that the product is determined using a quaternion multiplication $(\otimes)$

$$q = \left[\cos\left(\frac{\theta}{2}\right) \quad \hat{u}\cdot\sin\left(\frac{\theta}{2}\right)\right]^T \tag{1}$$

$$\Delta q[k] = q[k]\otimes\left(q[k-1]\right)^{-1}. \tag{2}$$

The DQ filter converts quaternion data to DQs in real time and then applies Kalman filtering, removing the quaternion orientation from the filter state variable and reducing the computational load when compared to quaternion filtering. The average angular velocity between measurements is estimated as a Euler value (azimuth, elevation, and roll) and then corrected with the measured change in orientation. The relationship between the

DQ and average angular velocity (ω) given by Chou [10] is used to convert Euler velocity to delta quaternions.

$$\Delta q(\omega,\tau) = \left[\cos\left(\frac{\theta}{2}\right) \quad \left(\frac{2\omega}{\|\omega\|}\right)\cdot\sin\left(\frac{\theta}{2}\right)\right]^T$$
$$\theta = \sqrt{\omega^T\cdot\omega} \tag{3}$$

When acceleration values are available, they are used to modify the average velocity.

B. EKF

The EKF uses the prediction-correction behavior of the Kalman filter to nonlinear systems [28], [38]. The state equation $f(x[k-1],u[k-1],w[k-1])$ relates the state at time $k(x[k])$ to the previous state $(x[k-1])$ (4). Additional parameters are a driving function b and process noise w, where w has the property of the zero-mean white Gaussian noise. The measurement (5) relates the state vector $(x[k])$ to the measurement data through the measurement function $h(x[k],v[k])$

$$x[k] = f\left(x[k-1],b[k-1],w[k-1]\right) \tag{4}$$
$$z[k] = h\left(x[k],v[k]\right). \tag{5}$$

In reality, the process noise is not exactly known at time k, so the state $(x[k|k-1])$ is an approximation of the true next state $(x[k])$ as a function of the previously corrected state $(x[k-1|k-1])$. Here, we have used the notation $x[k|k-1]$ to express the state vector at time step k conditioned on the previous state vector at time step $k-1$. Similarly, the measurement function produces an approximation $(z[k|k])$ of the measurement data $(z[k])$ due to the unknown value of the measurement noise v, where v has the property of the zero-mean white Gaussian noise at time k. The governing equation for the EKF state estimate approximates the true state vector $(x[k])$ and the true measurement $(z[k])$ using a Taylor expansion about conditional state $(x[k|k-1])$.

$$x[k]\approx x[k|k-1]+A[k]\cdot\varepsilon[k-1]+W[k]\cdot w[k-1] \tag{6}$$
$$z[k]\approx z[k|k]+H[k]\cdot\varepsilon[k-1]+V[k]\cdot v[k]$$
$$\varepsilon[k-1]=x[k-1]-x[k-1|k-1] \tag{7}$$

The A and W in (6) are Jacobian matrices of the state (4) with respect to the state vector x and the process noise w, respectively. The true measurement $(z[k])$ relates to the approximate measurement $(z[k|k])$ using the two matrices (H and V) and the measurement noise v (6). The H and V in (7) are Jacobian matrices of the function h with respect to the state vector x and measurement noise v, respectively.

III. Filter Design

A. MMDQ Design

The MMDQ filter builds upon our previous work with the DQ filter [15], improving on the DQ framework by replacing the single EKF with a three-model modified IMME [4] and changing the prediction algorithm to take advantage of the additional resources of the multiple model (MM) state vector. The overall MMDQ filter can be broken into seven separate processes: DQ computation, TPM update, probability and state mixing, extended Kalman filtering, weighting coefficient computation, state vector combination, and orientation prediction. The MMDQ filter does not include the quaternion state in the filter state vector, significantly reducing the complexity of the Kalman filters (see Fig. 1).

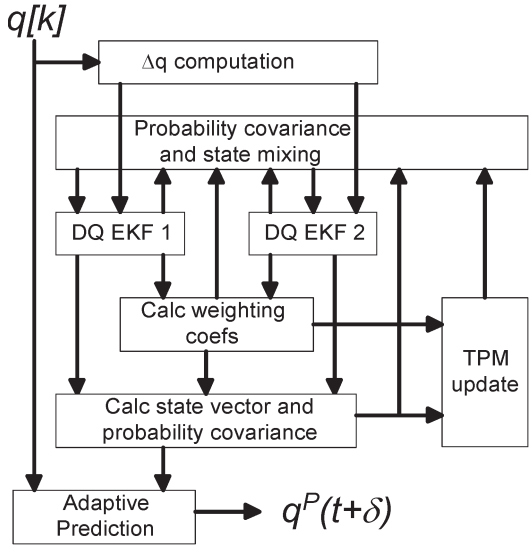

Fig. 1. MMDQ uses multiple DQ EKF filters in an IMME framework to provide improved head tracking. The IMME is modified to include an adaptive TPM for improved tracking. An adaptive algorithm predicts future orientation from the IMME state estimate and the measured orientation.

The IMME mode-switching process is assumed to be a Markov chain with a known mode TPM. The TPM can be estimated from offline data as a function of the expected sojourn time in each model [7], [23]. Although a fixed TPM can provide good results, the widely varying nature of a head tracking application presents large demands on the TPM estimation procedure [16]. For this experiment, we use an adaptive algorithm for TPM estimation operating on an initial estimate. Our equation notation will use subscripts for individual matrix elements $(M_{i,j})$ and bracketed superscript to identify matrix columns $(M^{\langle i\rangle})$.

A cost-effective method of computing the online TPM using a quasi-Bayesian estimator was presented by Li and Jilkov [23] and Marins *et al.* [26]. This method first computes the mixture probability density function (pdf) $g_{i,j}$ for the jth state element of the ith model from the likelihood function (L), the weighting coefficients (μ), and the previous TPM $(\prod)$ (8). Next, the Dirichlet distribution parameters γ are calculated from the pdf (g) and the previous parameters (9). A new TPM $(\prod)$ is computed as the average of the Dirichlet distribution parameters over the previous k frames (10). Equation (10) has been modified to prevent TPM elements from reaching zero by enforcing a minimum value [24]

$$g_{i,j}[k+1]=1+\frac{\mu_i[k]\cdot\left(L_j[k]-\left(\left(\Pi[k]\right)^{\langle j\rangle}\right)^T\cdot L[k]\right)}{(\mu[k])^T\cdot\Pi[k]\cdot L[k]} \tag{8}$$

$$\gamma_{i,j}[k+1] = \gamma_{i,j}[k] + \frac{\gamma_{i,j}[k] \cdot g_{i,j}[k]}{\sum_j (\gamma_{i,j}[k] \cdot g_{i,j}[k])} \tag{9}$$

$$\Pi_{i,j}[k+1] = \max\left[10^{-50}, \gamma_{i,j}[k]/(k+1)\right]. \tag{10}$$

The initial value of the TPM will be determined through the analysis of a data set that is representative of a typical head tracking application. The TPM elements are computed as the single step probability of each mode transition. The aforementioned adaptive computations are inserted in the IMME structure before the probability mixing stage.

The probability covariance and state vector of each EKF in the IMME are biased toward the overall solution of the IMME before the filters iterate. Each EKF filter is adjusted to the overall solution to prevent filter divergence, keeping the filter state near the operating point of the IMME. The MMDQ modifies this stage by applying a minimum value to the mixing coefficients and weighting coefficients to prevent any value from reaching zero. The recursive nature of the IMME can result in a filter being dropped from use once its weighting coefficient reaches zero [24]. The addition of a lower limit to the mixing process assures that the filter can effectively remove individual EKF solutions without permanently dropping the filter from the MMDQ.

Head motion is very unpredictable, ranging from benign stationary pose to erratic aggressive target tracking. The MMDQ deals with these wide variations by switching between multiple filters, each designed to handle a specific type of head motion. In our work, we evaluated the use of the constant velocity (CV) and constant acceleration (CA) models, which are described in Section III-B1 and B2. The weighting coefficients are computed using the standard IMME method, computing each coefficient as the product of the previous frame coefficient and the likelihood function. After computation, the weighting coefficients are normalized, and a lower bound is applied to avoid zero values. The state vector mixing uses the weighting coefficients to generate the state and probability covariance.

B. DQ Filter Design

Multiple model approaches are often used to improve prediction by using multiple instances of the same model, each tuned to handle a different type of head motion [13]. We have chosen to use two CV filters and a CA filter, each with different process noise. The high data rate of the simulation environment (120 Hz or more) allows us to use simple motion models such as the CV and CA for head tracking, reducing the complexity of the Kalman filters. The first CV filter will have low-level white noise and will be designed for slow changing and stationary orientation. The second CV filter will have high-level white noise and is intended for moderately changing head orientation. The CA filter will have high-level white noise to handle large changes in acceleration such as starts and stops.

1) CV Filter: The CV filter model uses a state vector (x_{CV}) containing the corrected average angular velocity $(\omega[k|k])$ to estimate the DQ Δq.

$$x_{CV}[k] = \omega[k|k] \tag{11}$$

The CV model state equation $f_{CV}(x, w)$ predicts the next state vector $(x_{CV}[k|k-1])$ as a function of the corrected state vector from the previous frame $(x_{CV}[k-1|k-1])$ and the process noise (w) (12). Since the CV model assumes that velocity does not change between measurements, the estimated velocity $(\omega[k|k-1])$ is a linear function of the corrected angular velocity state $(\omega[k-1|k-1])$, process noise (ω), and the frame period (τ)

$$f_{CV}\left(x[k-1], w[k-1]\right) = \omega[k-1|k-1] + w[k-1] \cdot \tau. \tag{12}$$

The measurement equation $h(x, v)$ converts the estimated angular velocity to a DQ (13). Note that the DQ CV filter has a linear state (12) but a nonlinear measurement (13). The measurement equation is identical in both of our motion models and therefore does not carry a model subscript

$$h\left(x[k], v[k]\right) = \Delta q\left([k|k-1], \tau\right) + v[k]. \tag{13}$$

The A matrix for the CV model (A_{CV}) is the partial derivative of (12) at time k with respect to state (x) which reduces to the identity matrix (14). The W matrix for the CV model (W_{CV}) is the partial derivative of (12) with respect to process noise (w), reducing to the frame time (τ) multiplied by the identity matrix (15). The V matrix is the partial derivative of (13) with respect to measurement noise (v), evaluated at the current state which reduces V to the identity matrix (16)

$$A_{CV}[k] = \frac{\partial}{\partial x_{CV}}\left[f_{CV}\left(x_{CV}[k-1], 0\right)\right] \quad A_{CV} = 1 \tag{14}$$

$$W_{CV}[k] = \frac{\partial}{\partial w}\left[f_{CV}\left(x_{CV}[k-1], 0\right)\right] \quad W_{CV} = 1 \tag{15}$$

$$V[k] = \frac{\partial}{\partial v}\left[h_{CV}\left(x[k|k-1], 0\right)\right] \quad V = 1. \tag{16}$$

The H matrix at time step k is the partial derivative of (13) at time step k with respect to the state variable (x). Expressing H as three column vectors, the general form is a function of the estimated DQ $(\Delta q(\omega[k|k-1], \tau))$, the estimated angular velocity $(\omega[k|k-1])$, and the sample period (τ).

$$\left(H_{CV}[k]\right)^{\langle i \rangle} = \begin{bmatrix} (-\tau/4) \cdot \Delta q_{i+1} \\ \omega_0 \cdot a + \delta_{i,0} \cdot b \\ \omega_1 \cdot a + \delta_{i,1} \cdot b \\ \omega_2 \cdot a + \delta_{i,2} \cdot b \end{bmatrix} \tag{17}$$

where

$$a = \frac{(\tau \cdot \omega_i \cdot \Delta q_0 - \Delta q_{i+1})}{(\omega^T \cdot \omega)} \quad b = \frac{\Delta q_{i+1}}{\omega_i}$$

$$\Delta q = \Delta q(\omega[k|k-1], \tau) \qquad \omega = \omega[k = k-1]$$

and $\delta_{i,j}$ is the Dirac function $\delta_{i,j} = \begin{cases} 1 & if\ i = j \\ 0 & if\ i \neq j \end{cases}$.

2) CA Filter: The CA filter models the changes in quaternion orientation as incremental accelerations between measurements [11]. The state vector at time k $(x_{CA}[k])$ contains

the corrected angular velocity $(\omega[k|k])$ and corrected angular acceleration $(\omega[k|k])$.

$$x_{CA}[k] = [\omega[k|k] \quad \alpha[k|k]]^T \tag{18}$$

The CA state equation $f_{CA}(x, w)$ expresses the predicted velocity $(\omega[k|k-1])$ as the sum of the velocity state $(\omega[k-1|k-1])$ and the product of the angular acceleration state $(\alpha[k-1|k-1])$, the frame time (τ), and the process noise (w) (19). The predicted angular acceleration $(\alpha[k|k-1])$ is the sum of the current acceleration state $(\alpha[k-1|k-1])$ and the product of the process noise (w) and the frame time (τ). The CA filter uses that same measurement equation as the CV filters (13)

$$f_{CA}\left(x[k-1], w[k-1]\right) = \begin{bmatrix} \omega + \alpha \cdot \tau + w \cdot \tau^2/2 \\ \alpha + w \cdot \tau \end{bmatrix}^T. \tag{19}$$

The A and W Jacobian matrices for the CA filter (A_{CA}, W_{CA}) can be derived by inspection from the expanded form. The A_{CA} matrix (20) and W_{CA} matrix (21) are derived separately for the CA filter, but the V matrix is unchanged since we are using the same measurement model (13)

$$A_{CA}[k] = \frac{\partial}{\partial x_{CA}} \left[f_{CA}\left(x_{CA}[k-1], 0\right)\right]$$

$$A_{CA} = \begin{bmatrix} I & \tau \cdot I \\ 0 & I \end{bmatrix} \tag{20}$$

$$W_{CA}[k] = \frac{\partial}{\partial w} \left[f_{CA}\left(x_{CA}[k-1], 0\right)\right]$$

$$W_{CA} = \begin{bmatrix} \frac{\tau^2}{2} \cdot I \; \tau \cdot I \end{bmatrix}^T. \tag{21}$$

The H Jacobian for the CA filter (H_{CA}) contains the partial derivatives of the measurement (13) with respect to each of the state variables (18). The general form of H_{CA} uses the same three column vectors of (17) but with the CA model used to compute the predicted angular velocity (22). The three partial derivatives of (13) with respect to acceleration (α) are functions of estimated angular velocity $(\omega[k|k-1])$, estimated DQ $(\Delta q(\omega[k|k-1], t))$, and the sample period (τ). The H_{CA} matrix can be expressed in a compact form by noting that the partial derivative with respect to angular acceleration (α) differs only in a term of τ from the partial's with respect to angular velocity (ω)(23)

$$H_{CA}[k] = \begin{bmatrix} \frac{\partial}{\partial \omega} \left[h_{CA}\left(x_{CA}[k|k-1], 0\right)\right] \\ h_{CA}\left(x_{CA}[k|k-1], 0\right) \end{bmatrix}^T \tag{22}$$

$$H_{CA}[k] = [I \quad \tau \cdot I] \cdot \frac{\partial}{\partial \omega} \left[h_{CA}\left(x_{CA}[k|k-1], 0\right)\right]. \tag{23}$$

C. Orientation Prediction

The DQ filters predict head orientation by estimating the change in orientation (Δq^P) across the prediction interval (δ)

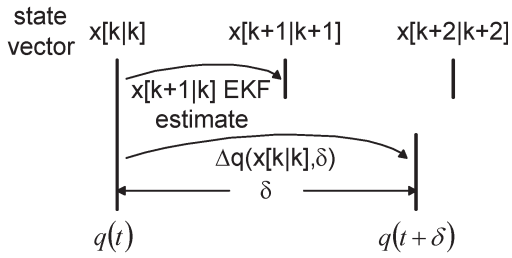

Fig. 2. Corrected EKF state vector $(x[k|k])$ is used to predict the change in head orientation (Δq) across a time interval δ to estimate future orientation at time t.

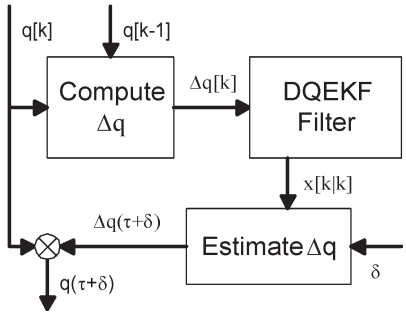

Fig. 3. Orientation is predicted from the DQ EKF by assuming that the current state estimate $(x[k|k])$ is constant across the prediction interval δ.

(see Fig. 2). The DQ is applied to the current quaternion measurement to predict orientation

$$q(t+\delta) = \Delta q\left(x[k|k], \delta\right) \otimes q(t). \tag{24}$$

D. DQ Prediction

The DQ EKF filters (CV and CA) predict the change in orientation across the entire prediction interval (d) in one computation (see Fig. 3). Each of the filters assumes that the current state vector is constant throughout the prediction interval when the average angular velocity is computed. A DQ representing the change in orientation across the prediction interval is constructed from the state vector (3).

The predicted orientation $(q(t+\delta))$ is calculated as the quaternion product of the current orientation $(q(t))$ and the predicted change in orientation (25). The prediction function for the CV model EKF assumes that the corrected angular velocity estimate $(\omega[k|k])$ is constant throughout the prediction interval. A DQ is generated from the velocity estimate (26) and prediction time based on Chou's work (3). The CA model uses both the corrected velocity and acceleration to predict future head orientation. The prediction function assumes that the estimated acceleration is constant during the prediction interval and computes the DQ accordingly (27)

$$q(t+\delta) = \Delta q\left(x[k|k], \delta\right) \otimes q(t) \tag{25}$$

$$\Delta q_{CV}\left(x[k|k], \delta\right) = \Delta q\left(\omega[k|k], \delta\right) \tag{26}$$

$$\Delta q_{CA}\left(x[k|k], \delta\right) = \Delta q\left(\omega[k|k] + 0.5 \cdot \alpha[k|k] \cdot \tau, \delta\right). \tag{27}$$

1) MMDQ Filter: The MMDQ filter estimates future orientation by computing the DQ (Δq) expected during the prediction interval (δ) from the corrected state estimate $(x_{MM}[k|k])$

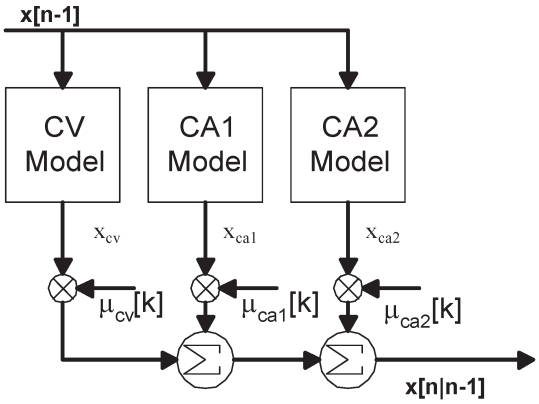

Fig. 4. MMDQ3 prediction function mixes the multiple MMDQ filter state estimates using the most recent weighting coefficients (μ). This process is repeated for each sample period (τ) in the prediction interval (δ).

and applying it to the current quaternion state estimate ($q[k]$) (see Fig. 4). The prediction interval is divided in $N = \delta/\tau$ time steps, and the predicted state for each of the N steps is computed by applying the CV and CA models to the current MMDQ state estimate ($x_{MM}[k|k]$) (28). The state estimate for step n in the prediction interval ($\omega_{MM}[n|k]$) is computed as the weighted average of the previous state estimate ($x[n-1|k]$) (29). The predicted quaternion (q_{MM}) is generated by recursively applying the DQ for each step in the prediction interval (30) and a partial step for any remaining part of the prediction time (δ)

$$x_{MM}[n|k] = \begin{bmatrix}\mu_{CV}[k]\\ \mu_{CA}[k]\end{bmatrix}^T \cdot \begin{bmatrix}x_{CV}[n-1|k]\\ x_{CA}[n-1|k]\end{bmatrix} \tag{28}$$

$$\omega_{MM}[n|k] = \begin{bmatrix}\mu_{CV}[k]\\ \mu_{CA}[k]\end{bmatrix}^T \cdot \begin{bmatrix}\omega_{CV}[n-1|k]\\ \omega_{CA}[n-1|k] + \alpha_{CA}[n-1|k]\cdot\tau\end{bmatrix} \tag{29}$$

$$q_{MM}[n|k] = \Delta q\left(\omega_{MM}[n|k], \tau\right) \otimes q_{MM}[n-1|k]. \tag{30}$$

The MMDQ prediction dynamically mixes the two motion models to create the predicted orientation, allowing the algorithm to adapt to the current state of the system each time the filter is iterated.

IV. Experimental Results

Head motion data were collected using a Polhemus Liberty ac magnetic tracker to provide measured data for the experiment. The experiment setup used a single sensor attached to the rear of a helmet with the magnetic source rigidly mounted approximately 0.2 m from the sensor (see Fig. 5). Five data sets were collected, three motion specific (benign, moderate, and aggressive) and two full motion (motion 1 and motion 2) (see Table I). Each of the collected data sets contains 100 000 sequential head orientation samples collected at a 120-Hz measurement rate.

A quaternion orientation data set was collected for three specific head motion categories (benign, moderate, and aggressive motion). The three motion categories were chosen to correlate with those used by Kyger and Maybeck [21] in their experiment with MM head orientation prediction. Kyger and Maybeck assembled three motion categories from data captured during simulator missions with experienced pilots at Armstrong laboratories. We created head orientation data for each of these categories to closely match that of the Kyger experiment by carefully controlling head motion during data collection. The normalized histogram of the motion specific data sets (see Fig. 6) indicates that each data set occupies a separate region of the angular velocity range.

The benign motion data set has a distribution that is sharply defined with very little acceleration content as would be expected when the pilot studies a stationary object. The moderate data set has a range of values that represent smooth motion as the pilot scans the airspace. The aggressive data set distribution is very broad and represents fast erratic motion.

Two additional data sets (Motion 0 and Motion 1) were taken with a full range of head motion for performance evaluation.

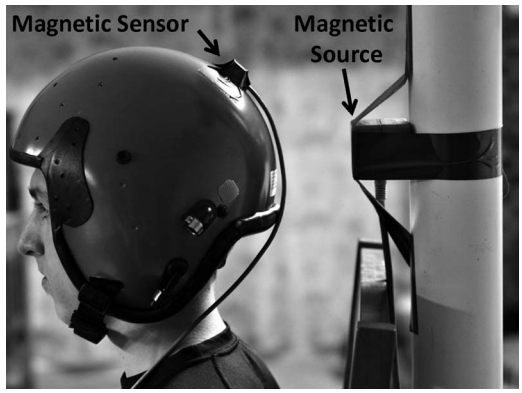

Fig. 5. Experimental setup uses a flight helmet with a sensor taped to the rear. The magnetic source is positioned on a stationary pole just behind the helmet to approximate a typical head tracking environment. This geometry minimizes the source/sensor spacing during data capture, reducing measurement noise.

TABLE I
Angular Motion by Data Set

Dataset	*Velocity (radians/s)*			*Acceleration (radians/s²)*		
	mean	*stdev*	*Max.*	*mean*	*stdev*	*Max.*
Benign	3.8e-3	2.0e-3	0.08	0.06	0.03	1.02
Moderate	0.32	0.03	0.76	0.47	0.25	3.27
Aggressive	1.05	0.22	2.83	4.00	3.07	18.6
Motion 0	0.27	2.92	1.02	0.98	3.90	10.8
Motion 1	0.36	2.82	2.92	1.20	4.37	14.6

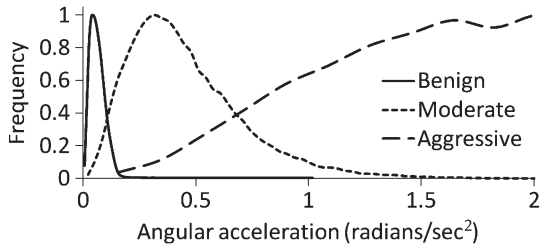

Fig. 6. Histogram of the benign, moderate, and aggressive motion sets normalized to the same frequency scale. Note that the three motion models define specific ranges of angular acceleration with overlapping regions. The aggressive motion histogram is only partially shown.

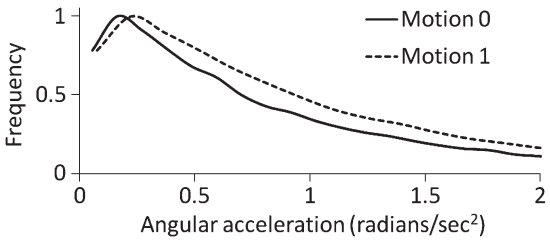

Fig. 7. Histogram of the angular head acceleration for the two full motion data sets shows that the head is generally experiencing moderate or benign motion. Note that we have not shown the "tail" of the two histograms to emphasize the peak near 0.25 rad/s^2.

TABLE II
MANUEVERING INDEX (λ) DISTRIBUTION

	1st Quartile	Median	3rd Quartile	98 percentile
Motion 0	0.16	0.26	0.48	1.07
Motion 1	0.17	0.28	0.49	1.07

Each of the full motion data sets features a complete range of head motion data from benign stationary pose to wildly aggressive tracking motions, similar to that expected in a simulation environment (see Fig. 7). The angular acceleration has a large standard deviation ($\sim$4 rad/s^2) and a small mean value ($\sim$1 rad/s^2) indicating short bursts of high acceleration.

We implemented the MMDQ in a two model (MMDQ2) and a three model (MMDQ3) configuration based on the analysis of the collected data sets. A CV model DQ EKF (DQEKF-CV) and a CA model DQ EKF (DQEKF-CA) are implemented to provide a performance comparison. The different filter types are evaluated for tracking error under various conditions. We then run an experiment with the proposed prediction algorithm.

A. MMDQ Configuration

The full motion data sets (Motion 0 and Motion 1) were evaluated to determine if the maneuvering index [19] of the collected data sets requires the Interactive Multiple Model (IMM) Kalman filter (see Table II). The maneuvering index (λ) is the ratio of the standard deviation of the process noise to the standard deviation of the measurement noise. We computed the CV process noise of each point using a 12-point sliding window centered on the point. The measurement noise was found by converting the DQ measurement to angular velocity and calculating the standard deviation.

As shown in Fig. 8, both motion data sets contain points with a maneuvering index greater than 0.5, indicating that the IMM will provide improved tracking over a single EKF [19]. The distribution of λ shows that a single EKF has adequate bandwidth for more than 75% of the samples. The three-filter MMDQ (MMDQ3) will use a CV filter for $(0.25 \leq \lambda < 0.75)$ (CV1), a CV filter for $(0.75 \leq \lambda < 1.25)$ (CV2), and a CA filter for $(\lambda > 1.25)$ (CA1). The two-filter configuration (MMDQ2) that uses a CV filter tuned for midrange $(0.25 < \lambda < 0.75)$ (CV1) and a CA filter for the moderate motion $(0.25 \leq \lambda < 0.75)$ will also be implemented. Each filter is tuned to the midpoint of its assigned maneuvering index range.

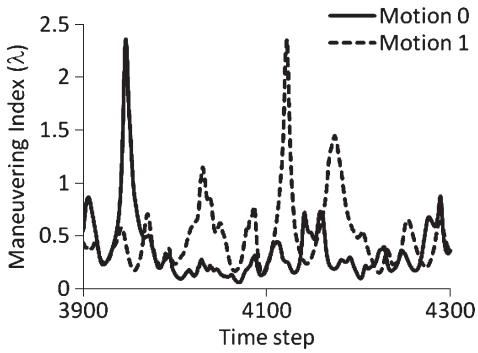

Fig. 8. Segment of the Motion 0 and Motion 1 data sets that display a large maneuvering index value due to an aggressive head motion. The large dynamic range of the data as it changes from benign motion to aggressive motion cannot be handled by a single EKF without large errors.

The initial value of the TPM was estimated by assuming that all state changes were the result of a single step Markov chain. The assigned ranges of λ were then used to assign a filter to each frame of the Motion 0 data set. The probability of a transition from filter i to filter j $(\prod_{i,j})$ is the number of transitions from i to j $(N_{i,j})$ in the data set divided by the total transitions from filter i $(\Sigma_j N_{i,j})$ (31). A TPM was generated for the IMM3 (32) and MMDQ2 (33) configurations using this process

$$\Pi_{i,j} = \frac{N_{i,j}}{\sum_j N_{i,j}} \tag{31}$$

$$\Pi_{MMDQ3} = \begin{bmatrix} 0.97 & 0.03 & 0.00 \\ 0.07 & 0.92 & 0.01 \\ 0.00 & 0.25 & 0.75 \end{bmatrix} \tag{32}$$

$$\Pi_{MMDQ2} = \begin{bmatrix} 0.97 & 0.03 \\ 0.07 & 0.94 \end{bmatrix}. \tag{33}$$

The probability of transitions between the CV1 filter and the CA1 filter in the MMDQ3 (32) was initially set to zero based on the procedure outlined earlier. We reasoned that direct high data rate of the tracker (120 Hz) was eliminating direct transitions between these two filters, forcing them to transit through the CV2 filter. Experimentation with the TPM showed that allowing transitions from CV1 to CA1 improved tracking by reducing the mode transition time at the onset of acceleration. We determined through empirical methods that a small transition probability in the CV1/CA1 location was sufficient to allow CV1/CA1 mode changes. The initial TPM value reduced the mode changes of the MMDQ, slowing the transition time between filters. This behavior reduced the mean error at the expense of the maximum error. To reduce maximum errors, we chose to use a more balanced TPM that increases the probability of mode changes and reduced the transition time between filters [4].

$$\Pi_{MM3} = \begin{bmatrix} 0.80 & 0.10 & 0.10 \\ 0.10 & 0.80 & 0.10 \\ 0.10 & 0.10 & 0.80 \end{bmatrix} \tag{34}$$

$$\Pi_{MM2} = \begin{bmatrix} 0.80 & 0.10 \\ 0.10 & 0.80 \end{bmatrix} \tag{35}$$

TABLE III
MMDQ FILTERS VERSUS MANEUVERING INDEX (λ)

	MMDQ3	MMDQ2
CV1	0.5	0.5
CV2	1.0	-
CA1\2	8.0	0.5

During simulation, it was determined that the adaptive TPM algorithm was moving the TPM toward (32) and (33); therefore, it was disabled to retain the balanced mixing of (34) and (35). The change to a fixed TPM with the balanced values resulted in reduced errors with the MMDQ filters.

B. Filter Tuning

1) Measurement Noise: The measurement noise $(v(k))$ is common to all three filters since the measurement equations are identical. We find $v(k)$ for a data set by subtracting a smoothed version of the data set from the measurement. This method was chosen so as to include dynamic errors of the tracker in the computation. Stationary measurement data originally taken for this process were found to have very little noise (< -90 dB) and were not representative of what we were seeing in the motion data. For this experiment, we assume that the individual components of the DQ are independent variables, allowing us to use the standard deviation (29) for the filters instead of the complete covariance matrix. These are typical values used for the experiments; the actual values were determined during the simulations to support the use of multifold cross-validation

$$\sigma_v = [2.09e-06 \quad 0.22e-03 \quad 8.22e-05 \quad 7.92e-05]^{\mathrm{T}}. \tag{36}$$

2) Process Noise: The process noise for each filter was determined from an assigned maneuvering index (λ) that represents the range of motion that we expect the filter to cover. For the MMDQ filters, we use the values from Table III where we have centered each filter in the range of λ that it is designed to support. For the EKF filter, we choose $\lambda = 0.5$ for both the CV model version (EKF-CV) and the CA model implementation (EKF-CA) to provide coverage for moderate motion.

In both the EKF filters, we bias the tuning toward moderate motion to provide better performance during aggressive motion. The MMDQ tuning allows us to choose a process noise for the CV1 filter that fits approximately 75% of the data sets without sacrificing performance during highly aggressive motion. The standard deviation of the process noise (ωw) was found by applying the measurement noise (ωv) and time step to λ. Note that the equation for the CV filter model process noise (σ_{CV}) (37) is slightly different than that for the CA filter model (σ_{CA}) (38).

$$\sigma_{CV} = \frac{\lambda \sigma_{WCV}}{\tau} \tag{37}$$

$$\sigma_{CA} = \frac{\lambda \sigma_{WCA}}{\tau^2} \tag{38}$$

The final tuning values for each of the filters are shown in Table IV as row vectors with each vector containing ω_v values for the azimuth, elevation, and roll components of the angular velocity (in that order). We developed these tuning values through manually reviewing simulation results as we "walk" the values to optimize results. Our values are tuned to the experimental setup which is modeled after an aircraft simulation application; other applications may benefit from different tuning value derived from training data in the simulation environment.

TABLE IV
PROCESS NOISE FILTER TUNING VALUES

	CV1	CA1	CA2
MMDQ3	[1.5, 0.6, 0.7]	[353, 152, 175]	[708, 304,350]
MMDQ2	[1.5, 0.6, 0.7]	[353, 152, 175]	-
EKF-CV	[1.5, 0.6, 0.7]	-	-
EKF-CA	-	-	[176, 76, 86]

The final MMDQ2 values have the CV1 filter identical to the individual DQEKF-CV and a high process noise CA filter. The MMDQ3 uses the same tuning for the CV1 and the CA1 filter, adding the CA2 filter for very aggressive motion. Notice that the DQEKF-CA uses a much lower process noise than the MMDQ2-CA1. The high gain of the CA1 filter in the MMDQ precludes its use as a stand-alone filter like the DQEKF-CA due to compromised performance with low acceleration data like the benign motion data set.

C. Angular Velocity Estimation

We evaluated the performance of the four filters using a tenfold cross-validation process with a 10 K sample validation interval and a nonoverlapping 90 K sample training interval. For the full motion data sets, the filters were tuned from the training set; performance was measured using the results obtained by running the validation data. The motion specific data sets used a similar approach except that tuning was determined by a training interval in the Motion 0 data set.

The two single stage filters (DQEKF-CV and DQEKF-CA) had different performance profiles that illustrated how their respective motion models favor specific types of head motion (see Table V). With benign motion, the CV filter had lower median error than the CA (3.5 versus 4.5 mrad). Under moderate motion, the CV again performs better, but in the aggressive motion, the two filters have similar performance. The differing results for these two filters illustrate how the CV motion model is suited for benign motion while the CA model is best suited for aggressive motion.

The two MMDQ filters had similar performance across all motion categories with the MMDQ3 providing the best performance. The MMDQ3 improves upon the DQEKF-CV median error results with the benign motion (3.0 versus 3.5 mrad) with better results than the DQEKF-CA. For full range motion (combined Motion 0 and Motion 1 data sets), the MMDQ3 was slightly better than the MMDQ2 due to improved performance with aggressive data. With the combination of a CV filter for benign to moderate motion and a CA filter for moderate to aggressive motion, the MMDQ improves performance with aggressive motion when compared to the two EKF filters. The addition of a third filter to the MMDQ structure for the MMDQ3 provided little improvement in structure for the

TABLE V
VELOCITY ESTIMATION ERROR BY MOTION CATEGORY (IN MILLIRADIANS PER SECOND)

	Full Motion		Benign Motion		Moderate Motion		Aggressive Motion	
	median	max	median	max	median	max	median	max
MMDQ2	18.5	207	3.0	40.6	14.9	79.3	32.3	295
MMDQ3	17.1	205	3.0	40.6	13.8	74.3	31.5	295
DQEKF-CV	15.6	233	3.5	38.8	11.8	66.8	35.1	259
DQEKF-CA	17.4	218	4.5	47.4	14.2	89.5	35.8	299

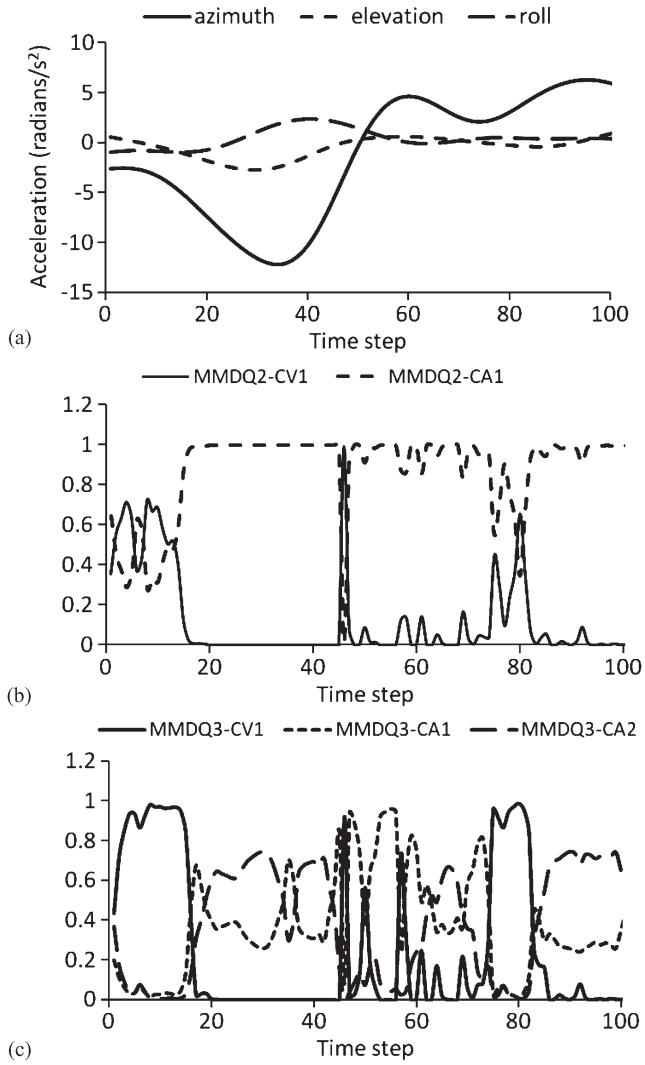

Fig. 9. Segment of the Motion 0 data set acceleration illustrates mode switching in the MMDQ filters. Here, we see the following: (a) Plot of angular head acceleration, (b) mixing weights for MMDQ2, and (c) model mixing weights for MMDQ3. The MMDQ2 is generally in one model or the other due to the wide spacing of the filter tuning ranges while the MMDQ3 switches in smooth curves, mixing two or more filter outputs due to the overlapping tuning ranges of the filters.

MMDQ3 and provided little improvement in the aggressive data set results as compared to that of the MMDQ2. The lack of improvement with the CA2 filter is mostly likely due to the limited dynamic range of our aggressive data set.

In their experiment with multiple model prediction, Kyger and Maybeck [21] used an aggressive motion data set with angular acceleration approaching 35 rad/s^2 while our aggressive data set had a maximum acceleration of 18 rad/s^2. An examination of the model weighting in the two MMDQ filters (see Fig. 9) illustrates that the MMDQ3 is using the CA2 model for high accelerations that use the CA1 model in the MMDQ2. The MMDQ3 is mixing the CA1 and CA2 models for these intervals while the MMDQ2 is relying on the CA1 filter alone. Based on these results, the MMDQ2 would be the better choice for our application since it has all the performance of the MMDQ3 without the additional computational overhead. However, a data set with higher acceleration (like that used by Kyger) may require more process noise than the CA1 can provide, requiring a switch to the MMDQ3.

D. Prediction Performance

Each of the four filters was evaluated for head orientation prediction over a range of prediction intervals from 0 to 100 ms. The DQEKF-CV predicts future orientation by assuming that the estimated head velocity is constant throughout the prediction interval. The DQEKF-CA applies a CA motion model using the estimated head velocity and acceleration. The MMDQ2 and MMDQ3 filters use our proposed prediction algorithm (28)–(30).

Prediction performance was evaluated in terms of the Euclidian norm of the Euler angle error and presented as median error, median OS error, and maximum. OS was defined as any estimate with more than 17.5 mrad (1°) of orientation error. We will concentrate on OS errors in our discussion of prediction performance since this kind of error causes the "swimming" effect that occurs with large display lag.

The prediction error for the full motion data sets (a combination of the Motion 0 and Motion 1 data sets) shows that the DQEKF-CA, MMDQ2, and MMDQ3 filters all provided better performance than the original DQEKF-CV filter (see Table VI). The MMDQ2 filter had the best performance, reducing maximum OS error by 41.2% for the 50-ms case as compared to the DQEKF-CV. The MMDQ3 and DQEKF-CA filters both reduced maximum OS by approximately 18% for the same case. With 75 ms of prediction, the MMDQ2 provides a 21.5% reduction in maximum OS while the MMDQ3 and DQEKF-CA have a 10% improvement. At 100 ms of prediction, all three filters had approximately a 7% reduction in maximum OS as compared to DQEKF-CV.

The improved performance with small prediction times indicates that the filters with an acceleration estimate are more responsive to changing head orientation than the velocity-only-based DQEKF-CV. Increased prediction times average any acceleration and provide a better fit with the CV motion model. This behavior is evidenced more clearly with the aggressive motion data (see Table VII) where the improvement in maximum OS is 54.9% for MMDQ2, 29.2% for MMDQ3, and

TABLE VI
MMDQ PREDICTION ERROR WITH FULL MOTION (IN MILLIRADIANS)

	50 ms			75ms			100 ms		
	median	OS median	OS Max	median	OS median	OS Max	median	OS median	OS Max
MMDQ2	2.34	18.9	23.7	4.91	21.9	59.0	8.34	28.0	107
MMDQ3	2.60	19.1	33.4	5.28	22.6	67.6	8.75	25.2	107
DQEKF-CV	2.75	20.8	40.3	5.49	23.3	75.2	9.00	25.9	115
DQEKF-CA	1.99	20.6	33.0	4.36	22.9	67.0	7.58	25.1	106

TABLE VII
MMDQ PREDICTION ERROR WITH AGGRESSIVE MOTION (IN MILLIRADIANS)

	50 ms			75ms			100 ms		
	median	OS median	OS Max	median	OS median	OS Max	median	OS median	OS Max
MMDQ2	1.94	0.00	10.0	3.81	17.5	17.5	6.19	19.3	28.2
MMDQ3	1.98	0.00	8.72	3.86	0.00	16.2	6.25	19.2	26.7
DQEKF-CV	1.99	0.00	9.38	3.88	0.00	17.4	6.27	20.0	28.3
DQEKF-CA	1.60	0.00	10.9	3.32	18.5	19.4	5.53	20.0	30.6

TABLE VIII
MMDQ PREDICTION ERROR WITH MODERATE MOTION (IN MILLIRADIANS)

	50 ms			75ms			100 ms		
	mean	OS median	OS Max	median	OS median	OS Max	median	OS median	OS Max
MMDQ2	3.80	19.5	21.8	9.00	22.4	61.0	16.3	25.6	117
MMDQ3	5.20	19.8	34.0	11.1	23.6	80.5	19.1	27.6	143
DQEKF-CV	6.45	21.3	48.3	12.9	24.6	100	21.5	29.2	168
DQEKF-CA	4.57	19.6	33.6	10.1	23.7	78.3	17.9	26.9	139

30% for DQEKF-CA with 50 ms of prediction. Increased prediction with aggressive motion reduces the improvement over the DQEKF-CV filter, but large improvements are still seen at 100 ms of prediction (MMDQ2: 30%; MMDQ3: 14.9%; DQEKF-CA: 17.2%). The moderate motion case (see Table VIII) shows very little difference between the four filters at any of the three prediction times. In the moderate case, the head motion is smooth, allowing the CV model to provide a good estimate of head velocity.

The CA motion model allows for quick response to changes in head velocity, but the CV model provides the best performance for most head motion. The majority of head motion is gradually changing smooth motion interspersed with sudden accelerations. The CA motion model is a good choice to track the accelerations but is less suited to the constant motion case. The DQEKF-CA filter had the lowest median error in all test cases, but the MMDQ filters had the lowest maximum OS. The ability of the MMDQ filters to switch to the CV motion model during periods of low acceleration provides better tracking of head velocity, allowing better predictions. The MMDQ2 filter had the best OS performance, nearly matching the median error of the DQEKF-CA while reducing maximum OS significantly (a 28% reduction in maximum OS as compared to DQEKF-CA for full motion with 50 ms of prediction).

The difference in performance between the MMDQ2 and MMDQ3 is due to tuning and motion model selection. For the MMDQ2, we chose a CV1/CA1 combination tuned with the same value as the equivalent DQ filters.

For the MMDQ3, we chose a CV1/CA1/CA2 combination with the CA2 for highly aggressive motion. The inclusion of a third motion model caused the MMDQ3 filter to spend more time transitioning between models than the MMDQ2 (see Fig. 5). The short bursts of acceleration in the head motion data were well suited to the MMDQ2 tuning, resulting in quick transitions between models for improved performance.

The two MMDQ filters, particularly the MMDQ2, provide improved performance under all condition, but it is the reduced OS that is important to VR/AR applications. The perception of display lag increases as head motion becomes more aggressive when the predictor overshoots, causing oscillopsia [2]. In this situation, the user perceives VR environment instability as the predictor first overshoots and then corrects, causing the display to "swim" as it catches up with the head motion. Under aggressive motion with 50 ms of prediction (a typical VR/AR system delay), the MMDQ2 filter reduced maximum OS by 35.1% for the DQEKF-CA and 54.8% for the DQEKF-CV. The large reduction in OS indicates that the MMDQ filters will provide a significant reduction in oscillopsia caused by display lag.

E. Computational Requirements

The MMDQ contains two (MMDQ2) or three (MMDQ3) DQ EKF filters, so we would expect it to consume additional bandwidth when compared to the DQ EKF. As shown in Table IX, the MMDQ imposes a significantly larger

TABLE IX
MMDQ COMPUTATIONAL REQUIREMENTS

Filter	Cycle Count	Execution Time (us)	Normalized Bandwidth requirement
MMDQ2	29572	295.7	3.11
MMDQ3	44360	443.6	4.66
EKF-CV	9512	95.1	1.00
EKF-CA	9622	96.2	1.01

Notes: Cycle counts are for a single iteration of the filter and were measured by implementing each algorithm in "C" on an Analog Devices ADSP-21161N floating point DSP operating at 100MHz Computational Requirements.

computation load than that of the DQ EKF, but the increase is small enough that it can easily be included in the firmware of a typical orientation tracker. For example, the MMDQ3 was added to the Polhemus Liberty tracker used in this study, maintaining the standard 240-Hz data rate of the tracker while improving prediction performance.

V. CONCLUSION

In this paper, we have developed an MMDQ filter for head orientation prediction. The proposed MMDQ filters integrate CV and CA DQ filters in an IMME framework to provide improved head orientation prediction. Two versions of the MMDQ filter were compared to the original DQ EKF with a CV motion model and a DQ EKF with a CA motion model. The two model MMDQ (MMDQ2) was shown to provide the best performance of the four filters, reducing maximum OS of the prediction orientation by 41% with 50 ms of prediction. Reduced maximum OS was seen for all prediction times as compared to the DQEKF-CV. A three model MMDQ (MMDQ3) reduced maximum OSs by as much as 17% when compared to the DQEKF-CV. The CA version of the DQ EKF had performance similar to that of the MMDQ3.

The large reduction in OS provided by the MMDQ filter indicates a correspondingly large reduction in oscillopsia in the VR/AR environment. This improved performance comes at the cost of a roughly 400% increase in computations, but the computational load of the prediction is still small when compared to that of the display generation.

ACKNOWLEDGMENT

The authors would like to thank Polhemus, Inc. (Colchester, Vermont), for their support of this study.

REFERENCES

[1] K. Ali, C. Vanelli, J. Biesiadecki, M. Maimone, U. Y. Cheng, A. Martin, and J. Alexander, "Attitude and position estimation on the Mars Exploration Rovers," in *Proc. IEEE Conf. Syst., Man, Cybern.*, Oct. 2005, vol. 1, pp. 20–27.

[2] R. S. Allison, L. R. Harris, M. Jenkin, U. Jasiobedzka, and J. E. Zacher, "Tolerance of temporal delay in virtual environments," in *Proc. IEEE Virtual Reality*, 2001, pp. 247–254.

[3] I. Y. Bar-Itzhack and Y. Oshman, "Attitude determination from vector observations: Quaternion estimation," *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-21, no. 1, pp. 128–136, Jan. 1985.

[4] Y. Bar-Shalom, X. Rong Li, and T. Kirubarajan, *Estimation With Applications to Tracking and Navigation*. New York: Wiley, 2001.

[5] W. D. Blair and G. A. Watson, "IMM algorithm for solution to benchmark problem for tracking maneuvering targets," in *Proc. Acquisition, Tracking, Pointing IX, SPIE 2221*, Orlando, Florida, 1994, pp. 476–488.

[6] H. Bom and Y. Bar-Shalom, "The interacting multiple model algorithm for systems with Markovian switching coefficients," *IEEE Trans. Autom. Control*, vol. 33, no. 8, pp. 780–783, Aug. 1988.

[7] G. Burdea, "Invited review: The synergy between virtual reality and robotics," *IEEE Trans. Robot. Autom.*, vol. 15, no. 3, pp. 400–410, Jun. 1999.

[8] L. Campo, P. Mookerjee, and Y. Bar-Shalom, "State estimation for systems with sojourn-time-dependent Markov model switching," *IEEE trans. Autom. Control*, vol. 36, no. 2, pp. 238–243, Feb. 1991.

[9] Y. Cheon and J. Kin, "Unscented filtering in a unit quaternion space for spacecraft attitude estimation," in *Proc. IEEE Int. Symp. Ind. Elect.*, Jun. 2007, pp. 66–71.

[10] J. Chou, "Quaternion kinematic and dynamic differential equations," *IEEE Trans. Robot. Autom.*, vol. 8, no. 1, pp. 53–64, Feb. 1992.

[11] E. Daeipour and Y. Bar-Shalom, "Adaptive beam pointing of phased array radar using IMM estimator," in *Proc. Amer. Control Conf.*, Baltimore, MD, 1994, pp. 2093–2097.

[12] E. Derbez, B. Remillard, and A. Jouan, "A comparison of fixed gain IMM against two other filters," in *Proc. IEEE Int. Conf. Inf. Fusion*, Jul. 2000, vol. 1, pp. THB2/3–THB2/9.

[13] C. Hilde, T. Moore, and M. Smith, "Multiple Model Kalman Filtering for GPS and Low-Cost INS Integration," Inst. Eng., Survey. Space Geodesy, Univ. Nottingham, Nottingham, U.K., 2004.

[14] H. Himberg, Y. Motai, and C. Barrios, "R-adaptive Kalman filtering approach to estimate head orientation for driving simulator," in *Proc. IEEE Conf. Intell. Transp. Syst.*, Sep. 2006, pp. 851–857.

[15] H. Himberg and Y. Motai, "Head orientation prediction: Delta quaternions versus quaternions," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 6, pp. 1382–1392, Dec. 2009.

[16] V. P. Jilkov and X. R. Li, "Bayesian estimation of transition probabilities for Markovian jump systems by stochastic simulation," *IEEE Trans. Signal Process.*, vol. 52, no. 6, pp. 1620–1630, Jun. 2004.

[17] P. Jilkov and X. R. Li, "Adaptation of transition probability matrix for multiple model estimators," in *Proc. Inter. Conf. Inf. Fusion*, Montreal, QC, Canada, Aug. 2001, pp. ThB 1.3–ThB 1.10.

[18] J. Y. Jung, B. Adelstein, and S. R. Ellis, "Discriminability of prediction artifacts in a time-delayed virtual environment," in *Proc. IEA*, 2000, pp. 499–502.

[19] T. Kirubarajan and Y. Bar-Shalom, "Kalman filter versus IMM estimator: When do we need the latter?" *IEEE Trans. Aerosp. Electron. Syst.*, vol. 39, no. 4, pp. 1452–1457, Oct. 2003.

[20] J. Kofman, X. Wu, T. Luu, and S. Verma, "Teleoperation of a robot manipulator using vision-based human–robot interface," *IEEE Trans. Ind. Electron.*, vol. 52, no. 5, pp. 1206–1219, Oct. 2005.

[21] D. Kyger and P. Maybeck, "Reducing lag in virtual displays using multiple model adaptive estimation," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 34, no. 4, pp. 1237–1248, Oct. 1998.

[22] E. J. Lefferts, F. L. Markley, and M. D. Shuster, "Kalman filtering for spacecraft attitude estimation," *J. Guid. Control, Dyn.*, vol. 5, no. 5, pp. 417–429, Sep.–Oct. 1982.

[23] X. R. Li and V. P. Jilkov, "A survey of maneuvering target tracking—Part V: Multiple model methods," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 41, no. 4, pp. 1255–1321, Oct. 2005.

[24] X. R. Li and Y. Zhang, "Numerically robust implementation of multiple-model algorithms," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 36, no. 1, pp. 266–278, Jan. 2000.

[25] J. Liang, C. Shaw, and M. Green, "On temporal-spatial realism in the virtual reality environment," in *Proc. Symp. User Interface Softw. Technol.*, 1991, pp. 19–25.

[26] J. Marins, X. Yun, E. Bachmann, R. B. McGhee, and M. J. Zyda, "An extended Kalman filter for quaternion-based orientation using MARG sensors," in *Proc. Int. Conf. Intell. Robots Syst.*, Nov. 2001, vol. 4, pp. 2003–2011.

[27] F. L. Markley, "Attitude error representations for Kalman filtering," *J. Guid., Control Dyn.*, vol. 26, pp. 311–317, Mar.–Apr. 2003.

[28] P. Maybeck, *Stochastic Models, Estimation and Control*. New York: Academic, 1979.

[29] E. Mazor, A. Averbuch, Y. Bar-Shalom, and J. Dayan, "Interacting multiple model methods in target tracking: A survey," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 34, no. 1, pp. 103–123, Jan. 1998.

[30] C. Nielson, M. Goodrich, and R. Ricks, "Ecological interfaces for improving mobile robot teleoperation," *IEEE Trans. Robot.*, vol. 23, no. 5, pp. 927–941, Oct. 2007.
[31] M. E. Pittelkau, "An analysis of the quaternion attitude determination filter," *J. Astronaut. Sci.*, vol. 51, no. 1, pp. 103–120, Jan.–Mar. 2003.
[32] A. van Rhijn, R. van Liere, and J. D. Mulder, "An analysis of orientation prediction and filtering methods for VR/AR," in *Proc. IEEE Virtual Reality Conf.*, 2005, pp. 67–74.
[33] A. Sabatini, "Quaternion-based extended Kalman filter for determining orientation by inertial and magnetic sensing," *IEEE Trans. Biomed. Eng.*, vol. 53, no. 7, pp. 1346–1356, Jul. 2006.
[34] M. Shuster, "A survey of attitude representations," *J. Astronaut. Sci.*, vol. 41, no. 4, pp. 439–517, Oct.–Dec. 1993.
[35] R. So, W. Lo, and A. Ho, "Effects of navigational speed on motion sickness caused by an immersive virtual environment," *Human Factors*, vol. 43, no. 3, pp. 452–461, 2001.
[36] R. H.-Y. So, Lag Compensation by Image Deflection and Prediction: A Review on the Potential Benefits to Virtual Training Applications for Manufacturing Industry, unpublished.
[37] R. Thompson, I. Reid, L. Munoz, and D. Murray, "Providing synthetic views for teleoperation using visual pose tracking in multiple cameras," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 31, no. 1, pp. 43–54, Jan. 2001.
[38] G. Welch and G. Bishop, "An introduction to the Kalman filter," in *Proc. SIGGRAPH*, 2001, pp. 5–47.
[39] J. Wen and K. Kreutz-Delgado, "The attitude control problem," *IEEE Trans. Autom. Control*, vol. 36, no. 10, pp. 1148–1162, Oct. 1991.
[40] J. Wu and M. Ouhyoung, "On latency compensation and its effects on head-motion trajectories in virtual environments," *Vis. Comput.*, vol. 16, no. 2, pp. 79–90, 2000.
[41] L. C. Yang, J. H. Yang, and E. M. Feron, "Multiple model estimation for improving conflict detection algorithms," in *Proc. IEEE Conf. Syst., Man, Cybern.*, Oct. 2004, vol. 1, pp. 242–249.

Henry Himberg received the B.S. degree in electrical and computer engineering from Clarkson University, Potsdam, NY, in 1982, the M.S. degree from the University of Vermont, Burlington, in 2005, and the Ph.D. degree in electrical and computer engineering from Virginia Commonwealth University, Richmond, in 2010.

He is currently a Principal Engineer with Polhemus, Colchester, VT. His research interests include signal processing, position/orientation measurement, and motion prediction.

Yuichi Motai (M'01) received the B.Eng. degree in instrumentation engineering from Keio University, Tokyo, Japan, in 1991, the M.Eng. degree in applied systems science from Kyoto University, Kyoto, Japan, in 1993, and the Ph.D. degree in electrical and computer engineering from Purdue University, West Lafayette, IN, in 2002.

He is currently an Assistant Professor of electrical and computer engineering with Virginia Commonwealth University, Richmond, VA, and a Director of the Sensory Intelligent Laboratory. His research interests include the broad area of sensory intelligence, particularly in medical imaging, pattern recognition, computer vision, and robotics.

Arthur Bradley received the B.S., M.S., and Ph.D. degrees in electrical engineering from Auburn University, Auburn, AL, in 1990, 1992, and 1999, respectively.

He is currently a Senior Analog Engineer with the Electronics Systems Branch of Langley Research Center (LaRC) at National Aeronautics and Space Administration, Hampton, VA, and a Director of LaRC's Robotics and Intelligent Machines Laboratory. His research interests include robotic systems, cable noise reduction techniques, and solid-state sensors. His current activities include the development of a novel amorphous fluid-filled robot and the integration of multispectral optical instruments onto an outdoor omnidirectional rover.