State estimation for distributed systems with sensing delay

Harold L. Alexander

Department of Aeronautics and Astronautics
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139, USA

ABSTRACT

Control of complex systems such as remote robotic vehicles requires combining data from many sensors where the data may often be delayed by sensory processing requirements. The number and variety of sensors make it desirable to distribute the computational burden of sensing and estimation among multiple processors. Classic Kalman filters do not lend themselves to distributed implementations or delayed measurement data. The alternative Kalman filter designs presented in this paper are adapted for delays in sensor data generation and for distribution of computation for sensing and estimation over a set of networked processors.

2. SENSING AND KALMAN ESTIMATION

Most state estimator designs are rather inflexible due to the close integration of sensing and estimation. [3] A classic Kalman filter estimator depends on a fixed set of plant measurements taken each sampling period in order to generate and maintain state estimates. Such an estimator is not designed to cope with missing data, asynchronous measurements, or significant sensor-related computation delays. Also, the close coupling of measurement, estimation, and control hinders distribution of the overall computational burden among multiple processors.

Many systems such as autonomous remote vehicles depend on suites of sensors such as vision processors, optical range finders, and proximity sensors for navigation. [1,4] Such sensors often require extensive computation, and the associated delays may result in control lags and limited sampling rates that threaten the quality of state estimation and control. Further, data dropouts may occur as each sensor loses and re-acquires tracking targets. Flexibility and tolerance for change are required in such systems, requiring an alternative estimator architecture.

Such an estimator must have the following qualities: (1) it must be able to accept measurements delivered asynchronously from a variety of sensors and to incorporate them into the state estimate; (2) it must function independently of the specific characteristics of each sensor; (3) it must be able to incorporate delayed data reflecting prior rather than current plant states into the state estimate; and (4) it must operate quickly to manage a potentially large number of data sources. The latter requirement motivates transfer of a substantial part of the burden of estimator calculation to the measurement processors associated with individual sensors.

The system described herein is based primarily on the theory of extended Kalman filters [2,3] with two modifications. The first introduces a consistent intermediate data format for outlying processors to use in reporting measurements, that is closely related to the system state representation. This eases distribution of the estimation task. The second modification allows the system to calculate the "relevance" of delayed data to reflect the current plant state, permitting correction of the current state based on the delayed data.

3. KALMAN FILTER ESTIMATION

The presentation here is based on a linearized representation of a multi-input, multi-output, continuous-time, time-varying nonlinear plant with state $\mathbf{x} = [x_1 x_2 ... x_n]^T$, dynamics matrix $F(\mathbf{x}, t)$, and control distribution matrix $G(\mathbf{x}, t)$:

$$\dot{\mathbf{x}}(t) = F(\mathbf{x}, t)\mathbf{x}(t) + G(\mathbf{x}, t)\mathbf{u}(t) + \mathbf{v}(t) \tag{1}$$

where the control input vector $\mathbf{u}$ is an $m \times 1$ vector of control inputs and the process noise vector $\mathbf{v}(t)$ is an $n \times 1$ vector of Gaussian white noise signals with covariance matrix $Q(t) = E(\mathbf{v}(t)\mathbf{v}(t)^T)$. An indeterminant number of measurements $\mathbf{y}^i(t), i = 1, 2, ...$ are transmitted by sensors $1, 2, ...$ The measurements $\mathbf{y}_i$ each depend on the current state $\mathbf{x}(t)$ according to the corresponding state-dependent measurement matrices H^i and are corrupted by white Gaussian measurement noise $\mathbf{w}^i$ where the measurement noise covariance matrix $R^i = E(\mathbf{w}^i \mathbf{w}^{i^T})$ represents the level and kind of uncertainty in $\mathbf{y}^i(t)$:

$$\mathbf{y}^i(t) = H^i(\mathbf{x}(t), t)\mathbf{x}(t) + \mathbf{w}^i(t) \tag{2}$$

The architecture and mathematics of a discrete-time extended Kalman filter are next described below. Since the classic Kalman filter is capable of handling only one measurement vector which is generated at each sampling time, superscripts on $\mathbf{y}(t)$ and $H(\mathbf{x}(t), t)$ will be omitted for the rest of this section.

The discrete-time Kalman filter bases its state estimate on measurements taken at evenly spaced sampling times $t_k, k = 1, 2, ...$ The Kalman filter estimation process consists of two processes that each occur once during each sampling period, or cycle, of the estimation and control system. These processes act to maintain and update an estimate $\hat{\mathbf{x}}(t) = [\hat{x}_1 . \hat{x}_2, ..., \hat{x}_n]^T$ of the current state of the system and a measure $P(t)$ of the quality of the state vector estimate. $P(t)$ is a matrix of estimation-error covariances $E(\tilde{x}_i \tilde{x}_j)$ where $\tilde{x}_i$ is the estimation error $\hat{x}_i - x_i$ for the ith state:

$$\begin{aligned} P &= E(\tilde{\mathbf{x}}\tilde{\mathbf{x}}^T) = E[(\hat{\mathbf{x}} - \mathbf{x})(\hat{\mathbf{x}} - \mathbf{x})^T] \\ &= \begin{bmatrix} E(\tilde{x}_1^2) & E(\tilde{x}_1\tilde{x}_2) & ... & E(\tilde{x}_1\tilde{x}_n) \\ E(\tilde{x}_2\tilde{x}_1) & E(\tilde{x}_2^2) & ... & E(\tilde{x}_2\tilde{x}_n) \\ \vdots & \vdots & \ddots & \vdots \\ E(\tilde{x}_n\tilde{x}_1) & E(\tilde{x}_n\tilde{x}_2) & ... & E(\tilde{x}_n^2) \end{bmatrix} \end{aligned} \tag{3}$$

The diagonal elements represent the expected squared error of the corresponding states. A large diagonal element therefore implies that the current estimate of that state is expected to be far from correct. The off-diagonal elements reflect the expected cross-correlation between state errors, a large off-diagonal element actually implies that, although the two corresponding states are expected to diverge from the truth, the probable relative senses of the divergences are known. The diagonal elements are of course positive, and in fact the entire covariance matrix is always positive definite. A zero covariance matrix corresponds to a perfect estimate $\hat{\mathbf{x}}(t)$; a diagonal covariance matrix with large elements reflects a very poor estimate.

The first update step during each sampling time t_k is to correct the estimate $\hat{\mathbf{x}}$ by comparing the measurement vector $\mathbf{y}$ and the estimated measurement vector $\hat{\mathbf{y}} = H(\hat{\mathbf{x}}(t_k), t_k)\hat{\mathbf{x}}(t_k)$. Since both the measurement data and the current state estimate are subject to uncertainty, their respective uncertainties R and P are used to balance the two in obtaining a new estimate. The update equation is:

$$\hat{\mathbf{x}}(t_k^+) = \hat{\mathbf{x}}(t_k^-) + L(t_k)(\mathbf{y}(t_k) - H(t_k)\hat{\mathbf{x}}(t_k^-)) \tag{4}$$

where t_k^- and t_k^+ denote the instants just before and just after the measurement is made and incorporated. The measurement update gain matrix L is derived according to the requirement that any remaining errors $\tilde{\mathbf{x}}(t_k^+)$

should be uncorrelated with the measurement inconsistency $\mathbf{y}(t_k) - H\hat{\mathbf{x}}(t_k^-)$, which results in a formula balancing the measurement error covariance matrix $P(t_k^-)$ and the noise covariance matrix $R(t_k)$:

$$L(t_k) = P(t_k^-)H(\hat{\mathbf{x}}(t_k^-), t_k)^T(H(\hat{\mathbf{x}}(t_k^-), t_k)P(t_k^-)H(\hat{\mathbf{x}}(t_k^-), t_k)^T + R(t_k))^{-1} \tag{5}$$

or, more simply with omission of dependencies,

$$L = PH^T(HPH^T + R)^{-1} \tag{6}$$

The corresponding update in P, reflecting the improved quality of the estimate due to incorporation of the measurement, is

$$\begin{aligned} P(t_k^+) &= P(t_k^-) - P(t_k^-)H(\hat{\mathbf{x}}(t_k^-), t_k)^T L(t_k)^T - L(t_k)H(\hat{\mathbf{x}}(t_k^-), t_k)P(t_k^-) + \\ &\quad L(t_k)H(\hat{\mathbf{x}}(t_k^-), t_k)P(t_k^-)H(\hat{\mathbf{x}}(t_k^-), t_k)^T L(t_k)^T + L(t_k)R(t_k)L(t_k)^T \end{aligned} \tag{7}$$

or

$$P_{\text{new}} = P - PH^TL^T - LHP + LHPH^TL^T + LRL^T \tag{8}$$

The second update applied to $\hat{\mathbf{x}}$ and P each sampling period accounts for the evolution of the plant state over the duration of the sampling period. Given a state estimate $\hat{\mathbf{x}}(t_k^+)$ and an estimation error covariance matrix $P(t_k^+)$, this "aging" step results in a new estimate and covariance matrix as of the end of the sampling period, $\hat{\mathbf{x}}(t_{k+1}^-)$ and $P(t_{k+1}^-)$. According to the plant model, its state evolves as:

$$\begin{aligned} \mathbf{x}(t_{k+1}) &= \Phi(\mathbf{x}(t_k), t_k)\mathbf{x}(t_k) + \Gamma(\mathbf{x}(t_k), t_k)\mathbf{u}_k \\ &= \Phi_k\mathbf{x}(t_k) + \Gamma_k\mathbf{u}_k + \mathbf{v}_k^* \end{aligned} \tag{9}$$

where the discrete-time state transition matrix Φ is

$$\Phi(\mathbf{x}(t_k), t_k) = \Phi_k = e^{F(\mathbf{x}(t_k), t_k)(t_{k+1} - t_k)} \tag{10}$$

the discrete-time control distribution matrix Γ is

$$\Gamma(\mathbf{x}(t_k), t_k) = \int_{t_k}^{t_{k+1}} e^{F(\mathbf{x}(t_k), t_k)(t_{k+1} - t)}G(\mathbf{x}(t_k), t_k)dt \tag{11}$$

and the discrete-time process noise vector, representing the cumulative effect of the process noise over (t_k, t_{k+1}), is

$$\mathbf{v}_k^* = \int_{t_k}^{t_{k+1}} e^{F(\mathbf{x}(t_k), t_k)(t_{k+1} - t)}\mathbf{v}(t)dt \tag{12}$$

Note that $\mathbf{u}(t)$ is assumed constant (equal to $\mathbf{u}_k$) over the time interval (t_k, t_{k+1}), and that changes in F and G over that time are assumed negligible (or are modeled as part of the process nose $\mathbf{v}(t)$).

The state estimate is updated according to models of the state transition matrix Φ and Γ:

$$\hat{\mathbf{x}}(t_{k+1}^-) = \Phi(\hat{\mathbf{x}}(t_k^+), t_k)\hat{\mathbf{x}}(t_k^+) + \Gamma(\hat{\mathbf{x}}(t_k^+), t_k)u_k \tag{13}$$

where

$$\hat{\Phi}(\hat{\mathbf{x}}(t_k), t_k) = \hat{\Phi}_k = e^{\hat{F}(\hat{\mathbf{x}}(t_k), t_k)(t_{k+1} - t_k)} \tag{14}$$

and

$$\hat{\Gamma}(\hat{\mathbf{x}}(t_k), t_k) = \int_{t_k}^{t_{k+1}} e^{\hat{F}(\mathbf{x}(t_k), t_k)(t_{k+1} - t)}\hat{G}(\mathbf{x}(t_k), t_k)dt \tag{15}$$

The hats distinguishing the approximations $\hat{\Phi}$ and $\hat{\Gamma}$ from Φ and Γ will be omitted from now on. This time update of $\hat{\mathbf{x}}$ may also occur by a more sophisticated process of numeric integration for highly nonlinear plants, [2] allowing slower sampling rates, but the use of the state transition matrix Φ is important for purposes of this paper.

The error covariance matrix P is updated according to both $\Phi(\hat{\mathbf{x}}(t_k^+), t_k)$ and the process noise covariance matrix Q_k, which incorporates the predicted effect on the estimate uncertainty of random disturbances acting on the state of the system over the sampling period:

$$P(t_{k+1}^-) = \Phi(\hat{\mathbf{x}}(t_k^+), t_k) P(t_k^+) \Phi(\hat{\mathbf{x}}(t_k^+), t_k)^T + Q_k^* \tag{16}$$

where the discrete-time process noise covariance matrix Q_k^* is

$$Q_k^* = \int_{t_k}^{t_{k+1}} e^{F(\hat{\mathbf{x}}(t_k), t_k)(t-t_k)} Q(t_k) (e^{F(\hat{\mathbf{x}}(t_k), t_k)(t-t_k)})^T dt \tag{17}$$

The calculation of these updates to $\hat{\mathbf{x}}$ and P is straightforward for a linear, time-invariant plant where F, G, H, and the measurement and process noise covariance matrices are constant. Linearity and time invariance mean that $\Phi(\hat{\mathbf{x}}(t_k^+), t_k)$, $\Gamma(\hat{\mathbf{x}}(t_k^+), t_k)$, $R(t_k)$, and Q_k^* are constant for all k and may be precalculated. For nonlinear or time-varying plants the process is more complex, but for high enough sampling rates fairly simple approximations may be used for Φ and Γ and the noise covariance matrices may often be modeled as constant.

4. DISTRIBUTED KALMAN FILTERING

The adaptation of the Kalman filter to accept measurements from a number of sources is a fairly simple one. Suppose that during each sampling period k at most one measurement vector $\mathbf{y}^i(t_k)$ may arrive. For now this measurement is assumed to reflect data taken as of time t_k. The total number of scalar measurement quantities represented here is therefore $p_{\text{total}} = \sum_i p_i$ where each $\mathbf{y}^i(t_k)$ has p_i elements.

The Kalman filter may easily accomodate such measurements. At each time t_k the update is made in the usual way so that

$$\hat{\mathbf{x}}(t_k^+) = \hat{\mathbf{x}}(t_k^-) + L(t_k)(\mathbf{y}(t_k) - H^i(t_k)\hat{\mathbf{x}}(t_k^-)) \tag{18}$$

where the measurement update gain $L(t_k)$ is based on the measurement matrix $H^i(\hat{\mathbf{x}}(t_k), t_k)$ and the measurement noise covariance matrix $R^i(\hat{\mathbf{x}}(t_k), t_k)$ corresponding to the measurement $\mathbf{y}^i$:

$$L(t_k) = PH^{i^T}(H^i P H^{i^T} + R^i)^{-1} \tag{19}$$

This system of multi-sensor estimation permits relatively convenient distribution of the estimation task. Assuming that the estimation processor and the various sensor-related processors are distributed as nodes on a communications network, the estimation processor may broadcast the current estimate $\hat{\mathbf{x}}(t_k^-)$ of the plant state to all sensor nodes before each sampling time t_k. Any sensor node i that is prepared to make a measurement uses this state estimate to help generate a measurement $y^i(t_k)$ and to calculate $H^i(\hat{\mathbf{x}}(t_k^-), t_k)$ and reports both of these back to the estimator module for updating $\hat{\mathbf{x}}(t_k^-)$. The sensing processor is thus responsible for all sensor-specific calculations, and the estimator need only incorporate the reported measurements. Since the estimator module is freed from sensor-specific calculations it can operate more efficiently at its single purpose and the suite of sensors can made quite fluid and tolerant of sensor failures. Addition or deletion of particular sensors requires no modification to the central estimator node's programming.

This system is not quite ideal, however, since the estimator processor node must process various types of measurement vectors. A given vector $y^i(t_k)$ may incorporate a single measurement or many measurements, possibly more than the order of the system. The estimator processor therefore must handle many dimensionalities of y^i and R^i, and its computational burden will therefore vary from one time step to the next according to the sensor being handled, reducing its maximum attainable performance. In order to stabilize this computational burden, it will help to change the way that sensor data is presented to the estimator.

Suppose that the sensor processor node itself generates an estimate $\hat{\mathbf{x}}_m(t_k)$ of the state based solely on the measurement $y^i(t_k)$. This estimate assumes no prior knowledge of the state so that the a priori value of P is assumed equal to ψI where ψ is a large real number. The a priori state estimate is conveniently assumed to equal zero. Referring to Equation 4,

$$\begin{aligned} \hat{\mathbf{x}}_m(t_k) &= 0 + L_m(\mathbf{y}(t_k) - 0) \\ &= L_m\mathbf{y}(t_k) \end{aligned} \tag{20}$$

$$\begin{aligned} L_m &= \psi I H^{i^T}(H^i(\psi I)H^{i^T} + R^i)^{-1} \quad \psi \text{ large} \\ &\approx \psi H^{i^T}(\psi H^i H^{i^T})^{-1} \\ &= H^{i^T}(H^i H^{i^T})^{-1} \\ &= H^{i^\dagger} \end{aligned} \tag{21}$$

where $H^{i^\dagger}$ represents the pseudoinverse of H^i for $p_i < n$. [5] $\hat{\mathbf{x}}_m(t_k)$ is thus the least-squares solution to the underdetermined equation $\mathbf{y}(t_k) = H\hat{\mathbf{x}}_m(t_k)$. The overdetermined case where $p_i > n$ and $H^i H^{i^T}$ is singular will require an alternative calculation. The error covariance matrix P_m corresponding to $\hat{\mathbf{x}}_m(t_k)$, for $p_i < n$, is:

$$\begin{aligned} P_m(t_l) &= \psi I - \psi H^{i^T} L^T - LH^i\psi + LH^i\psi H^{i^T} L^T + LRL^T \\ &= \psi(I - L_m H^i)(I - H^{i^T} L_m^T) + L_m R L_m^T \\ &= \psi(I - H^{i^T}(H^i H^{i^T})^{-1}H^i)(I - H^{i^T}(H^i H^{i^T})^{-1}H^i) + H^{i^T}(H^i H^{i^T})^{-1}R(H^i H^{i^T})^{-1}H^i \\ &= \psi(I - H^{i^\dagger}H^i)(I - H^{i^\dagger}H^i) + H^{i^\dagger}RH^{i^{\dagger^T}} \end{aligned} \tag{22}$$

$\hat{\mathbf{x}}_m(t_k)$ and $P_m(t_k)$, having been calculated and reported by a sensing node, serve as measurement and measurement-error covariance matrix in updating $\hat{\mathbf{x}}(t_k^-)$ and $P(t_k^-)$ with the corresponding measurement matrix $H_m(t_k)$ equal to the identity matrix. The task of the estimator processor is thus greatly simplified and regularized.

5. ESTIMATION WITH DATA DELAYS

Sensor data delays may arise in two ways. First, "report collisions" caused by multiple sensor nodes reporting to the estimator during the same time step will result in data being delayed as one node is forced to wait. (Incorporating multiple sensor measurements each time step would be possible but would slow the processor.) Second, complex sensor-related calculations may delay the availability of measurement data.

It is therefore desirable to accommodate the arrival at time t_k of delayed sense data that reflects the plant state at a prior time $t_l < t_k$. As a specific example, let us consider a measurement y_2 taken at time t_2, which is not available for incorporation into the plant state estimate until time t_5. Under "normal" circumstances, whereby y_2 is incorporated into the estimate without delay at time t_2 and P is updated as well, the resulting state estimate $\hat{\mathbf{x}}'_{5-}$ is

$$
\begin{aligned}
\hat{\mathbf{x}}'_{5-} &= \Phi_4\hat{\mathbf{x}}'_{4+} + \Gamma_4 u_4 \\
&= \Phi_4[\hat{\mathbf{x}}'_{4-} + L'_4(y_4 - H_4\hat{\mathbf{x}}_{4-})] + \Gamma_4 u_4 \\
&= \Phi_4[(I - L'_4H_4)\hat{\mathbf{x}}_{4-} + L'_4 y_4] + \Gamma_4 u_4 \\
&= \Phi_4[(I - L'_4H_4)(\Phi_3\hat{\mathbf{x}}_{3+} + \Gamma_3 u_3) + L'_4 y_4] + \Gamma_4 u_4 \\
&= \Phi_4[(I - L'_4H_4)(\Phi_3[(I - L'_3H_3)\hat{\mathbf{x}}_{3-} + L'_3 y_3] + \Gamma_3 u_3) + L'_4 y_4] + \Gamma_4 u_4 \\
&= \Phi_4[(I - L'_4H_4)(\Phi_3[(I - L'_3H_3)(\Phi_2\hat{\mathbf{x}}'_{2+} + \Gamma_2 u_2) + L'_3 y_3] + \Gamma_3 u_3) + L'_4 y_4] + \Gamma_4 u_4 \\
&= \Phi_4[(I - L'_4H_4)(\Phi_3[(I - L'_3H_3)(\Phi_2[\hat{\mathbf{x}}_{2-} + L_2(y_2 - H_2\hat{\mathbf{x}}_{2-})] + \Gamma_2 u_2) + L'_3 y_3] + \Gamma_3 u_3) + L'_4 y_4] + \Gamma_4 u_4 \\
&= \Phi_4(I - L'_4H_4)\Phi_3(I - L'_3H_3)\Phi_2[\hat{\mathbf{x}}_{2-} + L_2(\mathbf{y}_2 - H_2\hat{\mathbf{x}}_{2-})] + \\
&\quad \Phi_4(I - L'_4H_4)\Phi_3(I - L'_3H_3)\Gamma_2 u_2 + \Phi_4(I - L'_4H_4)\Phi_3 L'_3\mathbf{y}_3 + \Phi_4(I - L'_4H_4)\Gamma_3\mathbf{u}_3 + \Phi_4 L'_4\mathbf{y}_4 + \Gamma_4 4 \qquad (23)
\end{aligned}
$$

The "primed" update gainsL'_3 and L'_4 are influenced by the update to P at time t_2. If the measurement is delayed and cannot be incorporated at time t_2 the resulting $\hat{\mathbf{x}}_{5-}$ is:

$$
\begin{aligned}
\hat{\mathbf{x}}_{5-} &= \Phi_4[(I - L_4H_4)(\Phi_3[(I - L_3H_3)(\Phi_2\hat{\mathbf{x}}_{2-} + \Gamma_2 u_2) + L_3 y_3] + \Gamma_3 u_3) + L_4 y_4] + \Gamma_4 u_4 \\
&= \Phi_4(I - L_4H_4)\Phi_3(I - L_3H_3)\Phi_2\hat{\mathbf{x}}_{2-} + \\
&\quad \Phi_4(I - L_4H_4)\Phi_3(I - L_3H_3)\Gamma_2 u_2 + \Phi_4(I - L_4H_4)\Phi_3 L_3\mathbf{y}_3 + \Phi_4(I - L_4H_4)\Gamma_3\mathbf{u}_3 + \Phi_4 L_4\mathbf{y}_4 + \Gamma_4 4 \qquad (24)
\end{aligned}
$$

In this case, L_3 and L_4 differ from L'_3 and L'_4 because they do not reflect any update of P at time t_2. The difference between the two versions, that is the effect on $\hat{\mathbf{x}}_{5-}$ of the measurement update at t_2, is:

$$
\begin{aligned}
\delta\hat{\mathbf{x}}_5 = \hat{\mathbf{x}}'_{5-} - \hat{\mathbf{x}}_{5-} &= \Phi_4(I - L'_4H_4)\Phi_3(I - L'_3H_3)\Phi_2 L_2(\mathbf{y}_2 - H_2\hat{\mathbf{x}}_{2-}) + \\
&\quad \Phi_4[(I - L'_4H_4)\Phi_3(I - L'_3H_3) - (I - L_4H_4)\Phi_3(I - L_3H_3)](\Phi_2\hat{\mathbf{x}}_{2-} + \Gamma_2\mathbf{u}_2) + \\
&\quad \Phi_4[(I - L'_4H_4)\Phi_3 L'_3 - (I - L_4H_4)\Phi_3 L_3]\mathbf{y}_3 + \\
&\quad \Phi_4[(I - L'_4H_4) - (I - L_4H_4)]\Gamma_3\mathbf{u}_3 + \\
&\quad \Phi_4[L'_4 - L_4]\mathbf{y}_4 \qquad (25)
\end{aligned}
$$

The first term on the right-hand side represents the direct effect of the measurement update at time t_2, as affected by subsequent measurement and process updates to the state estimate. The remaining terms represent indirect effects due to the update of P at t_2 and the resulting difference between L'_i and L_i for $i = 3, 4$.

Clearly, the ideal method would be to calculate at time t_5 the quantity $\delta\hat{\mathbf{x}}_5$ and to update $\hat{\mathbf{x}}_{5-}$ accordingly, so that $\hat{\mathbf{x}}_{5+} = \hat{\mathbf{x}}_{5-} + \delta\hat{\mathbf{x}}_5 = \hat{\mathbf{x}}'_{5-}$. The estimate $\hat{\mathbf{x}}_{5+}$ would then exactly incorporate the measurement y_2 as if it had been incorporated in a timely manner at t_2. The estimate error covariance matrix P would have to be updated as well. Calculating $\delta\hat{\mathbf{x}}_5$ directly, however, is quite difficult due to the fact that not updating P at t_2 affects each subsequent measurement update in a complex manner.

A compromise is possible: although calculating y_2 introduces delays of several sampling periods, calculating the measurement sensitivity matrix H_2 and the measurement noise covariance matrix R_2 is often much easier. If these quantities may be calculated ahead of time (before t_2), then P may be updated at time t_2 even though updating $\hat{\mathbf{x}}$ is delayed as before to t_5. The result is that measurement updates at t_3 and t_4 occur just as if the update of $\hat{\mathbf{x}}$ had taken place at t_2, and the update to be applied at t_5 is therefore simplified:

$$
\delta\hat{\mathbf{x}}'_5 = \Phi_4(I - L'_4H_4)\Phi_3(I - L'_3H_3)\Phi_2 L_2(\mathbf{y}_2 - H_2\hat{\mathbf{x}}_{2-}) \qquad (26)
$$

Note that this time the measurement update gain matrices L_3' and L_4' have exactly the same values as if the complete update had taken place at t_2. Therefore, the latter terms from the right-hand side of Equation 25 have disappeared. The remaining term is relatively easy to calculate, as we shall see.

The disadvantage to this technique is that the update of P alone at t_2 reflects a "misrepresentation" of the accuracy of the estimate $\hat{\mathbf{x}}$. That is, the accuracy of $\hat{\mathbf{x}}$ is represented as having been improved by a measurement at t_2 that has not yet be incorporated. As a result, subsequent measurement updates at t_3 and t_4 are underemphasized each time due to the exaggerated accuracy of the respective a priori state estimates. On the other hand, this effect is reversed at t_5 when the delayed measurement is actually incorporated.

How severe is this disadvantage? It is certainly of limited duration, lasting each time only from the moment of measurement (t_2 in this case) to the moment of measurement reporting (t_5). Its effect on the intervening state estimates will depend in part on the "overlap" between the delayed measurement and the measurements at t_3 and t_4. That is, if the measurement at t_2 provides information about the same degrees of freedom of the plant (or the same combinations of degrees of freedom) as those at t_3 and t_4, then the measurements at t_3 and t_4 will be strongly suppressed. If, on the other hand, the measurements are rather independent (do not overlap) then the effect will be relatively small. This statement may be made somewhat more rigorously in terms of the projections of the row spaces of H_3 and H_4 on that of H_2. It is important to note that no *systematic* error is introduced by the technique.

6. IMPLEMENTATION

The state-estimate adjustment $\delta\hat{\mathbf{x}}_5'$ in Equation 26 is comprised of a "relevance matrix" multiplied by the update that would normally have been applied at time t_2, $\delta\hat{\mathbf{x}}_2 = L_2(\mathbf{y}_2 - H_2\hat{\mathbf{x}}_{2-})$. The relevance matrix describes the relevance of this measurement update (or of any small perturbation in the state estimate) at time t_{2+} to the state at t_{5-}, as influenced by each intervening process update and measurement update:

$$R_{2+,5-} = \Phi_4(I - L_4'H_4)\Phi_3(I - L_3'H_3)\Phi_2 \tag{27}$$

Each measurement update reduces the relevance of the t_2 update by a factor $(I - L_i'H_i)$. Each process update changes that relevance by a factor Φ_i. Keeping track of these effects on the relevance matrix permits calculating the appropriate update to apply at t_5 according to the delayed measurement y_2.

In order to calculate the relevance matrix corresponding to any pair of measurement and delayed-update times, the estimator maintains a running calculation of $R_{0,k}$, the relevance matrix from some initial time t_0 to the current time t_k. At each process update, the matrix is multiplied on the left by the state transition matrix Φ, and at each measurement update the matrix is multiplied on the left by $(I - L_3'H_3)$. Any intermediate relevance matrix $R_{l+,k-}$, $0 < l < k$ may be calculated according to knowledge of $R_{0,l}$ and $R_{0,k}$:

$$R_{l+,k-} = R_{0,k-}R_{0,l-}^{-1} \tag{28}$$

In summary, the overall estimation procedure is as follows. Just prior to each time t_i, the estimator node processor updates and broadcasts the values of P_{i-}, $\hat{\mathbf{x}}_{i-}$, and $R_{0,i-}$. Each "ready" sensor node uses these quantities to calculate an update to P and $\hat{\mathbf{x}}$, and the estimator processor accepts one measurement-update report from some sensor node. (The method by which this sensor node is selected is a subject of ongoing research.) The report may result in a immediate update of both $\hat{\mathbf{x}}$ and P or only of P if the actual measurement data vector $\mathbf{y}$ is not yet available. In the latter case, corresponding to a delayed measurement, the sensory node records the broadcast value of $R_{0,i-}$ and updates it to form $R_{0,i+}$. It also stores the current state estimate $\hat{\mathbf{x}}_{x-}$.

At a later time t_k, when the sensor node completes calculating its delayed measurement, it records the new value R_{0,k^-} of the relevance matrix as broadcast by the estimator node, and, using R_{0,k^-} and the stored value of R_{0,i^+}, it calculates the update $\delta\hat{\mathbf{x}}_k$:

$$\begin{aligned} \delta\hat{\mathbf{x}}_k &= R_{0,k^-} R_{0,i^+}^{-1} L_i (H_i \hat{\mathbf{x}}_{i^-} - y_i) \\ &= R_{0,k^-} R_{0,i^+}^{-1} \delta\hat{\mathbf{x}}_i \end{aligned} \qquad (29)$$

This update value is passed by the sensor node to the estimator node to update the state estimate:

$$\hat{\mathbf{x}}_{k^+} = \hat{\mathbf{x}}_{k^-} + \delta\hat{\mathbf{x}}_k \qquad (30)$$

From time to time, as the running calculation of the relevance matrix $R_{0,i}$ threatens to produce an ill-conditioned value, the "start time" t_0 is reset to the current time. If any delayed measurements are pending at this time, the final value of the "old" relevance matrix is recorded and used to help reconcile such delayed measurements when their calculation is complete.

7. DISCUSSION AND CONCLUSIONS

The algorithms presented avobe permits distribution of the computational burden of Kalman-filter-based sensing and estimation as well as the employment of sensors for which computation delays would ordinarily result in unacceptable control lags and sampling rates. By relieving the central estimation processor from all sensor-specific computations the algorithm permits (1) sensor fusion from a variety of sources, (2) flexibility in adding and subtracting sensors from the system, and (3) simplification of programming of estimation and sensor-specific codes. It is particularly well-suited for high-performance network-based controllers that depend on vision, radar imaging, and other computationally intensive sensor systems including autonomous remote vehicles.

The algorithm relies on the high degree of inter-processor connectivity possible with modern local networks. Its structure allows a logical and convenient distribution of computation among network nodes, permitting very high sampling rates for the estimator. Sensor nodes are freed from having to make similarly quick measurements by providing for incorporation of delayed measurements.

Certain aspects of the algorithm may result in handling ill-conditioned matrices, particularly due to the introduction of the large covariance matrix ψI. Further algorithm development may result in alternative formulations for the associated parts of the algorithm that avoid such ill conditioning.

The ability to incorporate data from a large, possibly changing variety of sources and to accommodate sensing delays without degrading estimation speed or time response is extremely valuable in systems where the sensing process is computationally intensive. The author intends to apply the above algorithms to laboratory research with an unmanned, neutrally-buoyant submersible remote vehicle used to simulate free-flying space robots. This vehicle will use computer vision for real-time feedback control of three-dimensional position and orientation in conjunction with a set of inertial and depth sensors. The resulting vision-based control system will serve for stationkeeping and maneuvering both autonomously and under human supervisory control.

8. ACKNOWLEDGEMENTS

The author would like to gratefully acknowledge the support of the Bradley Foundation for the research leading to publication of this paper.

9. REFERENCES

[1] Alexander, Harold L., "Experiments in Teleoperator and Autonomous Control of Space Robotic Vehicles," *Proceedings of the International Symposium on Artificial Intelligence, Robotics, and Automation in Space (i-SAIRAS '90)*, Kobe, Japan, November 18-20, 1990.

[2] Hollars, Michael G., *Experiments in End-Point Control of Manipulators with Elastic Drives*, Ph.D. Dissertation, Stanford University Department of Aeronautics and Astronautics, Stanford, California, 1988.

[3] Kailath, Thomas, *Linear Systems*, Prentice-Hall, Inc, Englewood Cliffs, New Jersey, 1980.

[4] Spofford, J. and Akin, D.,"Redundancy Control of a Free-Flying Telerobot," *AIAA Journal of Guidance and Control*, Vol. 13, No. 3, May-June 1990.

[5] Strang, Gilbert, *Linear Algebra and its Applications*, Second Edition, Academic Press, New York, New York, 1980.