The Central Idea

- Even incomplete observations provide *some* information
- Globally observable process can be estimated by sequentially incorporating locally unobservable measurements
Why SCAAT?

- Improved Accuracy
  - Simultaneity Assumption avoided
  - Concurrent device autocalibration

- Improved estimation rates and latencies

Simultaneity Assumption

- Measurements collected sequentially ....
- .... but treated as simultaneous for computation.
Simultaneity (some typical numbers)

Let, at time of first observation
\[ \tilde{x}(t) = [1,1] \, m \]

Then, at time of second observation
\[ \tilde{x}(t + \tau_m) = [1.005,1.005] \, m \]

Consequently,
\[ u_1 = -3.627e-3 \, m \, , \, u_2 = 3.828e-3 \, m \]

So, our position estimate would be
\[ \tilde{x}(t + \tau_m) = [1.003,1.026] \, m \]
**Estimation Rates and Latencies**

- **Nyquist Criterion**
  Sampling Frequency \( > (2 \times \text{Target Motion BW}) \)

- **Virtual Environment Systems** - Visual feedback latency

---

**SCAAT Helps ....**

\[ \tau_m \rightarrow \text{Measurement time per observation} \]

\[ N \rightarrow \text{Number of Observations} \]

\[ \tau_c(N) \rightarrow \text{Computation Time} \]

\[ \text{Estimation Rate}, \quad \rho_e = \frac{1}{N\tau_m + \tau_c(N)} \]

Sequential Tracker  
SCAAT tracker
Hybrid Systems and Multi-Sensor Data Fusion

**Sequential Systems**

**SCAAT System**

**AutoCalibration**

- New tracking estimate for each measurement
- Individual device imperfections segregated
SCAAD DEFINITIONS

**Process Model**

- Position-Velocity (PV) Model
- Accelerations $\tilde{\eta}[i]$ modeled as zero-mean white noise sources,
**Model State and Transitions**

- Inter-sample time \( \delta(t) \)
- Inter-state transition
  \[
  \dot{x}(t) = A(\delta t) \ddot{x}(t - \delta t) + \dot{\omega}(t)
  \]
- Complete target state
  - n-element internal state vector \((n = 12)\)
    \[
    \ddot{x}(t) = [x \ y \ z \ \dot{x} \ \dot{y} \ \dot{z} \ \dot{\phi} \ \dot{\theta} \ \dot{\psi} \ \phi \ \theta \ \psi]^T
    \]
  - 4-element external quaternion
    \[
    \ddot{x} = (\alpha_w, (\alpha_x, \alpha_y, \alpha_z))
    \]

**State Transition Matrix** \( A(\delta t) \)

- Implements
  \[
  \begin{align*}
  x(t) &= x(t - \delta t) + \dot{x}(t - \delta t) \delta t \\
  \dot{x}(t) &= \dot{x}(t - \delta t)
  \end{align*}
  \]
  \[
  A(\delta t) = \begin{bmatrix}
  1 & 0 & 0 & \delta t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 1 & 0 & 0 & \delta t & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 1 & 0 & 0 & \delta t & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \delta t & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & \delta t & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \delta t & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \delta t \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
  \end{bmatrix}
  \]
**Process Noise**

- $n$ dimensional process noise vector $\rightarrow \tilde{\omega}(t)$
- Normal distribution, zero-mean, white sequence

- Process noise covariance matrix

$$E\{\tilde{\omega}(t) \tilde{\omega}^T(t + \epsilon)\} = Q, \quad \epsilon = 0$$

$$= 0, \quad \epsilon \neq 0$$

**Measurement Model**

- Predict ideal, noise-free response of each sensor-source pair

- Distinguishes SCAAT Kalman filter from well-constrained Kalman filter
**SCAAT Measurements**

- Pure Sense → Single scalar measurements
- Practically → One multi-dimensional constraint per estimate

For sensor $\sigma$, $m_{\sigma}$–dimensional constraint.

**Heuristic for Choosing SCAAT Measurements**

- During each KF measurement update, observe single source-sensor pair
- Extract one geometric constraint per observation
Predicting Measurement from State

- For some sensor type $\sigma$,
  
  $\hat{z}_\sigma(t)$: Measurement Estimate Vector  
  $\tilde{h}_\sigma()$ : Measurement Function  

  $$\hat{z}_\sigma(t) = \tilde{h}_\sigma(\bar{x}(t), \bar{\alpha}(t), \bar{b}(t), \bar{c}(t)) + \bar{\nu}_\sigma(t)$$

  $\bar{b}(t)$ : Source parameter vector (beacon)  
  $\bar{c}(t)$ : Sensor parameter vector (camera)

Measurement Noise

- $\bar{\nu}_\sigma(t)$: $m_\sigma$ – dimensional measurement noise vector  
- $R_\sigma(t)$: $m_\sigma \times m_\sigma$ measurement noise covariance matrix

  $$E\{\bar{\nu}_\sigma(t) \bar{\nu}_\sigma^T(t + \epsilon)\} = R_\sigma(t), \quad \epsilon = 0$$

  $$= 0, \quad \epsilon \neq 0$$
**Measurement Jacobian**

- Signifies sensitivity of measurement

\[ H_o(\ddot{x}(t), \ddot{a}(t), \ddot{b}(t), \ddot{c}(t)) [i, j] = \frac{\partial}{\partial \ddot{x}[j]} H_o(\ddot{x}(t), \ddot{a}(t), \ddot{b}(t), \ddot{c}(t))[i] \]

**State Error Covariance**

- Reflects filter’s estimate of its own uncertainty

- Error in filter’s state estimate,
  \[ \hat{\epsilon}(t) = \ddot{x}(t) - \dot{x}(t) \]

- State Error Covariance,
  \[ P(t) = E\{\hat{\epsilon}(t)\hat{\epsilon}^T(t)\} \]
SCAAT ALGORITHM

Time Update
- *a priori* state and error covariance

\[
\dot{x}^- = A(\delta t) \dot{x}(t - \delta t)
\]

\[
P^- = A(\delta t) P(t - \delta t) A^T(\delta t) + Q(\delta t)
\]
**Time Update (Contd....)**

\[ P = \text{error covariance (density) before time update} \]

\[ P^- = \text{error covariance (density) after time update} \]

\[ = A(\delta t)P(t-\delta t)A^T(\delta t) + Q(\delta t) \]

---

**Measurement Prediction**

\[ \hat{z} = \tilde{h}_o(\hat{x}^- (t), \hat{\alpha}^- (t), \tilde{b}(t), \tilde{c}(t)) \]

\[ H = H_o(\hat{x}^- (t), \hat{\alpha}^- (t), \tilde{b}(t), \tilde{c}(t)) \]
**Measurement Residual**

\[ \Delta \tilde{z} = \tilde{z}_\sigma(t) - \hat{z} \]

**Kalman Gain**

- Weights the residual in correction cycle

\[ K = P^{-1} H^T (HP^{-1} H^T + R_\sigma(t))^{-1} \]
Update the State

- Compute \textit{a posteriori} state estimate
  \[
  \hat{x}(t) = \hat{x}^- + K \Delta \hat{z}
  \]

Update Error Covariance

- Compute \textit{a posteriori} error covariance.
  \[
  P(t) = (I - KH)P^-
  \]

\[
\text{state coordinate frame after time update}
\]

\[
\text{state coordinate frame after measurement correction}
\]

\[
\text{world coordinate frame}
\]

\[
P^- = \text{error covariance (density) after time update}
\]

\[
P = \text{error covariance (density) after measurement correction}
\]
**Update External Quaternion**

\[
\Delta \hat{\alpha} = \text{quaternion}(\hat{x}[\Delta \phi], \hat{x}[\Delta \theta], \hat{x}[\Delta \psi])
\]

\[
\hat{\alpha} = \hat{\alpha} \otimes \Delta \hat{\alpha}
\]

Zero the incremental orientation state elements.

\[
\hat{x}[\Delta \phi] = \hat{x}[\Delta \theta] = \hat{x}[\Delta \psi] = 0
\]

---

**Algorithm Summary**

- Given: Initial state estimate, \( \hat{x}(0) \)
  - External Orientation estimate, \( \hat{\alpha}(0) \)
  - Error Covariance estimate, \( P(0) \)

- For each measurement \( z_\sigma(t) \) from sensor \( \sigma \) (and corresponding source) at time \( t \):
  1. Compute elapsed time \( \delta t \)
  2. Predict state and error covariance

\[
\hat{x}^- = A(\delta t) \hat{x}(t - \delta t)
\]

\[
P^- = A(\delta t) P(t - \delta t) A^T(\delta t) + Q(\delta t)
\]
**Algorithm Summary (Contd....)**

3. Predict measurement and compute Jacobian
\[
\hat{z} = \tilde{h}_o(\hat{x}'(t), \dot{\hat{x}}'(t), \tilde{b}(t), \tilde{c}(t))
\]
\[
H = H_o(\hat{x}'(t), \dot{\hat{x}}'(t), \tilde{b}(t), \tilde{c}(t))
\]

4. Compute Kalman gain
\[
K = P^- H^T (HP^- H^T + R_o(t))^{-1}
\]

5. Compute measurement residual
\[
\Delta \tilde{z} = \tilde{z}_o(t) - \hat{z}
\]

**Algorithm Summary (Contd....)**

6. Correct the prediction
\[
\hat{x}(t) = \hat{x}' + K \Delta \tilde{z}
\]
\[
P(t) = (I - KH)P^-
\]

7. Update external orientation quaternion
\[
\Delta \hat{\alpha} = \text{quaternion}(\hat{x}[\phi], \hat{x}[\theta], \hat{x}[\psi])
\]
\[
\hat{\alpha} = \hat{\alpha} \otimes \Delta \hat{\alpha}
\]

8. Zero incremental orientation state elements
\[
\hat{x}[\Delta \phi] = \hat{x}[\Delta \theta] = \hat{x}[\Delta \psi]
\]
**SCAAAT AutoCalibration**
- In effect, a distinct device filter for each source or sensor to be calibrated.

- Calibrate parameters in $\tilde{b}(t)$ and $\tilde{c}(t)$

- Notation: $\hat{x} \rightarrow$ Augmented matrix/vector

---

**Device Filters**
- Device parameters: $\hat{x}_\pi[i], i = 0, \ldots, n_\pi - 1$

1. (a) Allocate $n_\pi$ dimensional state vector $\hat{x}_\pi$ for the device.
   (b) Initialize with a priori estimates (from design).

2. (a) Allocate $n_\pi \times n_\pi$ noise covariance matrix $Q_\pi(\hat{\sigma})$
   (b) Initialize with expected parameter variances.

3. (a) Allocate $n_\pi \times n_\pi$ error covariance matrix $P_\pi$
   (b) Initialize with level of confidence in 1. above
**Revised Tracking Algorithm**

- Form augmented state vector
  \[
  \hat{x}(t - \delta t) = [\hat{x}^T(t - \delta t) \ \hat{x}_b^T(t - \delta t) \ \hat{x}_c^T(t - \delta t)]
  \]

- Augmented error covariance matrix
  \[
  \begin{bmatrix}
  P(t - \delta t) & 0 & 0 \\
  0 & P_b(t - \delta t) & 0 \\
  0 & 0 & P_c(t - \delta t)
  \end{bmatrix}
  \]

- State Transition Matrix
  \[
  \begin{bmatrix}
  A(\delta t) & 0 & 0 \\
  0 & I & 0 \\
  0 & 0 & I
  \end{bmatrix}
  \]

- Process Noise Matrix
  \[
  \begin{bmatrix}
  Q(\delta t) & 0 & 0 \\
  0 & Q_b(\delta t) & 0 \\
  0 & 0 & Q_c(\delta t)
  \end{bmatrix}
  \]

**AutoCalibration (Contd....)**

- Follow the original algorithm.
- Extract and save device filter portions.
  \[
  \begin{align*}
  \hat{x}_b(t) &= \hat{x}(t)[i...j] \\
  P_b(t) &= \hat{P}(t)[i...j,i...,j] \\
  \hat{x}_c(t) &= \hat{x}(t)[k...l] \\
  P_c(t) &= \hat{P}(t)[k...l,k...,l]
  \end{align*}
  \]

where
\[
\begin{align*}
i &= n + 1 \\
j &= n + n_b \\
k &= k + n_b + 1 \\
l &= n + n_b + n_c
\end{align*}
\]
- Small angles
  \[
  \cos(\Delta \theta) \approx 1, \quad \sin(\Delta \theta) \approx \Delta \theta
  \]

- Symmetry of \( Q(\Delta t), R_\sigma, P \)

- Sparse Matrices - \( A(\Delta t) \) and Jacobian \( H \)

- Matrix inversion - reduced size of measurement vector

**Code Optimization Strategies**
Filter Stability

- If there be real numbers $\alpha_1, \beta_1 > 0$ and $\alpha_2, \beta_2 < \infty$ such that for all $k \geq N$, for some $N \geq n/m$

$$\alpha_1 I \leq \sum_{i=k-N}^{k-1} A(t_k - t_{i+1})Q(t_k - t_{i+1})A^T(t_k - t_{i+1}) \leq \alpha_2 I$$

$$\beta_1 I \leq \sum_{i=k-N}^k \Gamma(i,k) \leq \beta_2 I$$

where

$$\Gamma(i,k) = A^T(t_i - t_k)H^T(t_i)R^{-1}(t_i)H(t_i)A(t_i - t_k)$$

$$H(t_i) = H_{\sigma}(\tilde{x}(t_i),\tilde{b}(t_i),\tilde{c}(t_i))$$

then the global system given by our dynamic and measurement model is uniformly asymptotically stable.

Source and Sensor Ordering Schemes

- Use a measurement scheduling algorithm
  - Better resource utilization
  - Monitor and control uncertainty in state vector

- Round-robin implementation
EXPERIMENTS AND SIMULATIONS

Hi-Ball Tracker

- Inside-out tracking
Initialization – Tracker

- 15-element state vector: $\hat{x}(t) = [\hat{x}(t) \ \hat{x}_b(t)]$
- State Transition Matrix:
  - Main tracker filter: $A(\delta t)$
  - Beacon filter: 3x3 Identity matrix
- Noise covariances determined off-line.

- Beacon filter state $\rightarrow$ initialized to (erroneous) position estimates.
- Beacon Error covariance matrix $\rightarrow$ initialized to
  $$P_b(0)[i, j] = \begin{cases} (0.001)^2 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

Measurement Noise

- $R_\sigma(t) \rightarrow$ Uncertainty in actual camera measurement
  $$R_\sigma(t)[i, j] = \begin{cases} \lambda_c & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$
- Distance and angle-dependent variance $\lambda_c$
  $$\sqrt{\lambda_c} = \frac{\sqrt{\lambda_0} d_b^2}{a \alpha_b^3 + b \alpha_b^2 + c \alpha_b + 1}$$
- Use previous position estimate to compute $d_b$ and $\alpha_b$
Simulation - Accuracy

- Collinearity - typically uses $N = 10$ observations per estimate.

- SCAAT
  - Higher update rate
  - Kalman Filtering
  - Autocalibration

Autocalibration

- Stationary HiBall on stable platform.
- Estimate position with overhead beacons
- Deviation of estimates with time progressively decreases once autocalibration turned on.
** Blocked Cameras

- Occlusion
- Camera not facing ceiling

---

** The Weak Links ....

- **Cold Start**
  
  Initialization to completely erroneous state vector sometimes leads to divergence.
Thank You!!