https://ntrs.nasa.gov/search.jsp?R=20170000373 2019-08-29T15:51:16+00:00Z

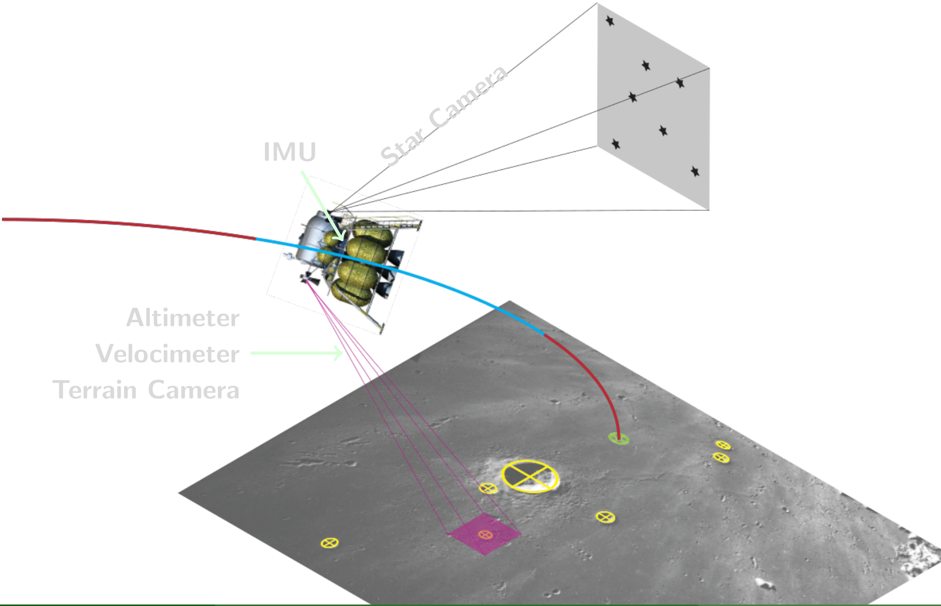
Comparison of Factorization-based Filtering for Landing Navigation

James S. McCabe[1] Aaron J. Brown[2] Kyle J. DeMars[1] John M. Carson III[2]

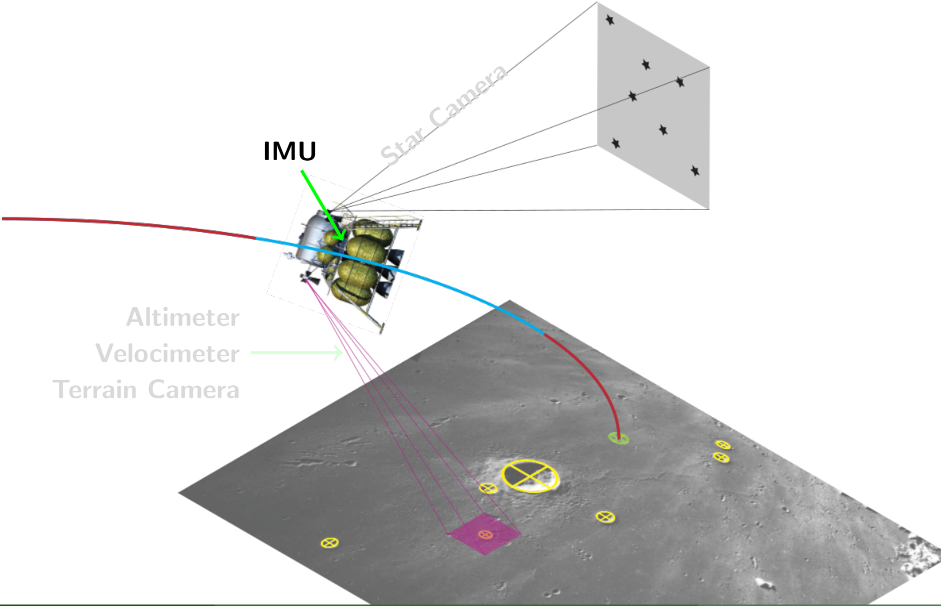
[1]Missouri University of Science and Technology

[2]NASA Johnson Space Center

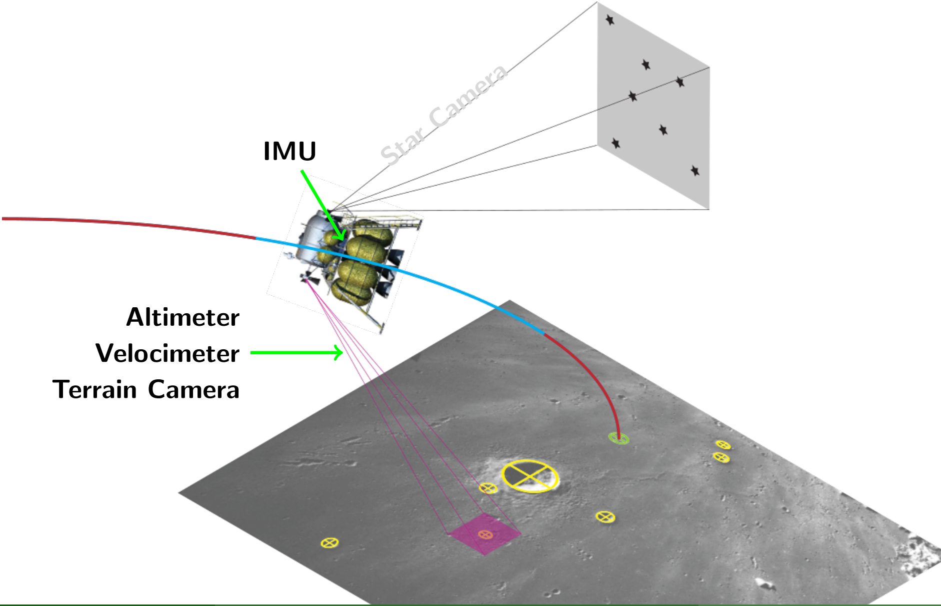
Motivation



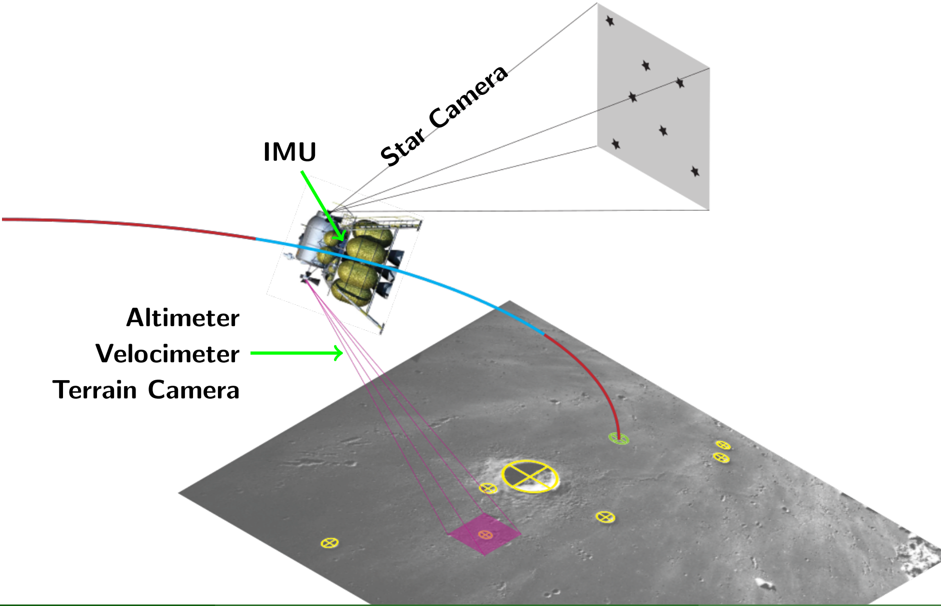
Motivation



Motivation



Motivation



Objective

- Blend data from sensors
  - inertial measurement unit
  - star camera
  - altimeter
  - velocimeter
  - terrain camera

Objective

- Blend data from sensors
  - inertial measurement unit
  - star camera
  - altimeter
  - velocimeter
  - terrain camera
- Produce estimates of position, velocity, and attitude
  - accurate
  - precise
  - consistent
  - robust

Objective

- Blend data from sensors
  - inertial measurement unit
  - star camera
  - altimeter
  - velocimeter
  - terrain camera
- Produce estimates of position, velocity, and attitude
  - accurate
  - precise
  - consistent
  - robust
- Work within minimum variance estimation framework
  - Kalman filter
  - extended Kalman filter
  - unscented Kalman filter

Scope of this Work

- Blend data from sensors
  - inertial measurement unit
  - star camera
  - altimeter
  - velocimeter
  - terrain camera
- Produce estimates of position, velocity, and attitude
  - accurate
  - precise
  - consistent
  - robust
- Work within minimum variance estimation framework
  - Kalman filter
  - extended Kalman filter
  - unscented Kalman filter

Scope of this Work

- Blend data from sensors
  - ✓ inertial measurement unit
  - ✓ star camera
  - ✓ altimeter
  - ○ velocimeter
  - ○ terrain camera
- Produce estimates of position, velocity, and attitude
  - ○ accurate
  - ○ precise
  - ○ consistent
  - ○ robust
- Work within minimum variance estimation framework
  - ○ Kalman filter
  - ○ extended Kalman filter
  - ○ unscented Kalman filter

Scope of this Work

- Blend data from sensors
  - ✓ inertial measurement unit
  - ✓ star camera
  - ✓ altimeter
  - ○ velocimeter
  - ○ terrain camera
- Produce estimates of position, velocity, and attitude
  - ○ accurate
  - ○ precise
  - ✓ consistent
  - ✓ robust
- Work within minimum variance estimation framework
  - ○ Kalman filter
  - ○ extended Kalman filter
  - ○ unscented Kalman filter

Scope of this Work

- Blend data from sensors
  - ✓ inertial measurement unit
  - ✓ star camera
  - ✓ altimeter
  - ○ velocimeter
  - ○ terrain camera
- Produce estimates of position, velocity, and attitude
  - ○ accurate
  - ○ precise
  - ✓ consistent
  - ✓ robust
- Work within minimum variance estimation framework
  - ○ Kalman filter
  - ○ extended Kalman filter
  - ○ unscented Kalman filter

Scope of this Work

- Blend data from sensors
  - ✓ inertial measurement unit
  - ✓ star camera
  - ✓ altimeter
  - ○ velocimeter
  - ○ terrain camera
- Produce estimates of position, velocity, and attitude
  - ○ accurate
  - ○ precise
  - ✓ consistent
  - ✓ robust
- Work within minimum variance estimation framework
  - ○ ~~Kalman filter~~
  - ○ ~~extended Kalman filter~~
  - ○ ~~unscented Kalman filter~~

Scope of this Work

- Blend data from sensors
  - ✓ inertial measurement unit
  - ✓ star camera
  - ✓ altimeter
  - ○ velocimeter
  - ○ terrain camera
- Produce estimates of position, velocity, and attitude
  - ○ accurate
  - ○ precise
  - ✓ consistent
  - ✓ robust
- Work within minimum variance estimation framework
  - ○ ~~Kalman filter~~
  - ○ ~~extended Kalman filter~~ multiplicative extended Kalman filter
  - ○ ~~unscented Kalman filter~~ multiplicative unscented Kalman filter

Scope of this Work

- Blend data from sensors
  - ✓ inertial measurement unit
  - ✓ star camera
  - ✓ altimeter
  - ○ velocimeter
  - ○ terrain camera
- Produce estimates of position, velocity, and attitude
  - ○ accurate
  - ○ precise
  - ✓ consistent
  - ✓ robust
- Work within minimum variance estimation framework
  - ○ ~~Kalman filter~~
  - ✓ ~~extended Kalman filter~~ multiplicative extended Kalman filter
  - ○ ~~unscented Kalman filter~~ multiplicative unscented Kalman filter

The Linear Problem

- Consider the linear state-space model

$$\boldsymbol{x}_k = \boldsymbol{F}_{k-1}\boldsymbol{x}_{k-1} + \boldsymbol{w}_{k-1}$$
$$\boldsymbol{z}_k = \boldsymbol{H}_k\boldsymbol{x}_k + \boldsymbol{v}_k$$

The Linear Problem

- Consider the linear state-space model

$$\boldsymbol{x}_k = \boldsymbol{F}_{k-1}\boldsymbol{x}_{k-1} + \boldsymbol{w}_{k-1}$$
$$\boldsymbol{z}_k = \boldsymbol{H}_k\boldsymbol{x}_k + \boldsymbol{v}_k$$

- The well-known Kalman filter produces the conditional mean and covariance through a two-stage recursion:

Initial Cond.	$\boldsymbol{m}_{k-1}^{+}$	$=$	$\boldsymbol{m}_0$
	$\boldsymbol{P}_{k-1}^{+}$	$=$	$\boldsymbol{P}_0$
Mean Prop.	$\boldsymbol{m}_k^{-}$	$=$	$\boldsymbol{F}_{k-1}\boldsymbol{m}_{k-1}^{+}$
Cov. Prop.	$\boldsymbol{P}_k^{-}$	$=$	$\boldsymbol{F}_{k-1}\boldsymbol{P}_{k-1}^{+}\boldsymbol{F}_{k-1}^{T} + \boldsymbol{Q}_{k-1}$
Kalman Gain	$\boldsymbol{K}_k$	$=$	$\boldsymbol{P}_k^{-}\boldsymbol{H}_k^{T}[\boldsymbol{H}_k\boldsymbol{P}_k^{-}\boldsymbol{H}_k^{T} + \boldsymbol{R}_k]^{-1}$
Mean Update	$\boldsymbol{m}_k^{+}$	$=$	$\boldsymbol{m}_k^{-} + \boldsymbol{K}_k(\boldsymbol{z}_k - \boldsymbol{H}_k\boldsymbol{m}_k^{-})$
Cov. Update	$\boldsymbol{P}_k^{+}$	$=$	$\boldsymbol{P}_k^{-} - \boldsymbol{K}_k\boldsymbol{H}_k\boldsymbol{P}_k^{-}$

The Linear Problem

- Consider the linear state-space model

$$\boldsymbol{x}_k = \boldsymbol{F}_{k-1}\boldsymbol{x}_{k-1} + \boldsymbol{w}_{k-1}$$
$$\boldsymbol{z}_k = \boldsymbol{H}_k\boldsymbol{x}_k + \boldsymbol{v}_k$$

- The well-known Kalman filter produces the conditional mean and covariance through a two-stage recursion:

Initial Cond.	$\boldsymbol{m}_{k-1}^{+} = \boldsymbol{m}_0$
	$\boldsymbol{P}_{k-1}^{+} = \boldsymbol{P}_0$
Mean Prop.	$\boldsymbol{m}_{k}^{-} = \boldsymbol{F}_{k-1}\boldsymbol{m}_{k-1}^{+}$
Cov. Prop.	$\boldsymbol{P}_{k}^{-} = \boldsymbol{F}_{k-1}\boldsymbol{P}_{k-1}^{+}\boldsymbol{F}_{k-1}^{T} + \boldsymbol{Q}_{k-1}$
Kalman Gain	$\boldsymbol{K}_{k} = \boldsymbol{P}_{k}^{-}\boldsymbol{H}_{k}^{T}[\boldsymbol{H}_k\boldsymbol{P}_{k}^{-}\boldsymbol{H}_{k}^{T} + \boldsymbol{R}_k]^{-1}$
Mean Update	$\boldsymbol{m}_{k}^{+} = \boldsymbol{m}_{k}^{-} + \boldsymbol{K}_k(\boldsymbol{z}_k - \boldsymbol{H}_k\boldsymbol{m}_{k}^{-})$
Cov. Update	$\boldsymbol{P}_{k}^{+} = \boldsymbol{P}_{k}^{-} - \boldsymbol{K}_k\boldsymbol{H}_k\boldsymbol{P}_{k}^{-}$

Covariance Constraints

- By definition, the covariance matrix **must** be
  - symmetric
  - positive definite
- A proper filtering recursion should always maintain these properties.
- For the linear case, where the Kalman filter is theoretically exact, do these properties hold?

Covariance Constraints

- By definition, the covariance matrix **must** be
  - symmetric
  - positive definite
- A proper filtering recursion should always maintain these properties.
- For the linear case, where the Kalman filter is theoretically exact, do these properties hold?

Symmetry: Propagation

Given that $\boldsymbol{P}_{k-1}^{+} = (\boldsymbol{P}_{k-1}^{+})^T$, it is clear from

$$\boldsymbol{P}_k^{-} = \boldsymbol{F}_{k-1}\boldsymbol{P}_{k-1}^{+}\boldsymbol{F}_{k-1}^{T} + \boldsymbol{Q}_{k-1}$$

that the propagated covariance matrix is algebraically symmetric.

Covariance Constraints

- By definition, the covariance matrix **must** be
  - symmetric ✓
  - positive definite
- A proper filtering recursion should always maintain these properties.
- For the linear case, where the Kalman filter is theoretically exact, do these properties hold?

Symmetry: Propagation

Given that $\boldsymbol{P}_{k-1}^{+} = (\boldsymbol{P}_{k-1}^{+})^T$, it is clear from

$$\boldsymbol{P}_k^- = \boldsymbol{F}_{k-1}\boldsymbol{P}_{k-1}^{+}\boldsymbol{F}_{k-1}^{T} + \boldsymbol{Q}_{k-1}$$

that the propagated covariance matrix is algebraically symmetric.

Covariance Constraints

- By definition, the covariance matrix **must** be
  - symmetric ✓
  - positive definite
- A proper filtering recursion should always maintain these properties.
- For the linear case, where the Kalman filter is theoretically exact, do these properties hold?

Symmetry: Update

Given that $\boldsymbol{P}_k^- = (\boldsymbol{P}_k^-)^T$, the update is given by

$$\begin{aligned}\boldsymbol{P}_k^+ &= \boldsymbol{P}_k^- - \boldsymbol{K}_k \boldsymbol{H}_k \boldsymbol{P}_k^- \\ &= \boldsymbol{P}_k^- - \boldsymbol{P}_k^- \boldsymbol{H}_k^T [\boldsymbol{H}_k \boldsymbol{P}_k^- \boldsymbol{H}_k^T + \boldsymbol{R}_k]^{-1} \boldsymbol{H}_k \boldsymbol{P}_k^-\end{aligned}$$

Therefore, the updated covariance matrix is algebraically symmetric.

Covariance Constraints

- By definition, the covariance matrix **must** be
  - symmetric ✓✓
  - positive definite
- A proper filtering recursion should always maintain these properties.
- For the linear case, where the Kalman filter is theoretically exact, do these properties hold?

Symmetry: Update

Given that $\boldsymbol{P}_k^- = (\boldsymbol{P}_k^-)^T$, the update is given by

$$\begin{aligned}\boldsymbol{P}_k^+ &= \boldsymbol{P}_k^- - \boldsymbol{K}_k\boldsymbol{H}_k\boldsymbol{P}_k^- \\ &= \boldsymbol{P}_k^- - \boldsymbol{P}_k^-\boldsymbol{H}_k^T[\boldsymbol{H}_k\boldsymbol{P}_k^-\boldsymbol{H}_k^T + \boldsymbol{R}_k]^{-1}\boldsymbol{H}_k\boldsymbol{P}_k^-\end{aligned}$$

Therefore, the updated covariance matrix is algebraically symmetric.

Covariance Constraints

- By definition, the covariance matrix **must** be
  - symmetric ✓✓
  - positive definite
- A proper filtering recursion should always maintain these properties.
- For the linear case, where the Kalman filter is theoretically exact, do these properties hold?

Symmetry: General Comment

In the worst case, brute-force symmetrization can be used:

$$\boldsymbol{P}_k = \frac{1}{2}(\boldsymbol{P}_k + \boldsymbol{P}_k^T)$$

Covariance Constraints

- By definition, the covariance matrix **must** be
  - symmetric ✓✓
  - positive definite
- A proper filtering recursion should always maintain these properties.
- For the linear case, where the Kalman filter is theoretically exact, do these properties hold?

Positive Definiteness: Propagation

Given that $\boldsymbol{P}_{k-1}^{+} > \mathbf{0}$ and that $\boldsymbol{F}_{k-1}$ is full rank, the noise-free propagation of covariance is guaranteed to be positive definite; therefore,

Covariance Constraints

- By definition, the covariance matrix **must** be
  - symmetric ✓✓
  - positive definite
- A proper filtering recursion should always maintain these properties.
- For the linear case, where the Kalman filter is theoretically exact, do these properties hold?

Positive Definiteness: Propagation

Given that $\boldsymbol{P}_{k-1}^{+} > \mathbf{0}$ and that $\boldsymbol{F}_{k-1}$ is full rank, the noise-free propagation of covariance is guaranteed to be positive definite; therefore,

$$\underbrace{\boldsymbol{F}_{k-1}\boldsymbol{P}_{k-1}^{+}\boldsymbol{F}_{k-1}^{T}}_{>\mathbf{0}}$$

Covariance Constraints

- By definition, the covariance matrix **must** be
  - symmetric ✓✓
  - positive definite
- A proper filtering recursion should always maintain these properties.
- For the linear case, where the Kalman filter is theoretically exact, do these properties hold?

Positive Definiteness: Propagation

Given that $\boldsymbol{P}_{k-1}^{+} > \mathbf{0}$ and that $\boldsymbol{F}_{k-1}$ is full rank, the noise-free propagation of covariance is guaranteed to be positive definite; therefore,

$$\boldsymbol{P}_k^{-} = \underbrace{\boldsymbol{F}_{k-1}\boldsymbol{P}_{k-1}^{+}\boldsymbol{F}_{k-1}^{T}}_{>\mathbf{0}} + \underbrace{\boldsymbol{Q}_{k-1}}_{\geq\mathbf{0}}$$

Algebraically, the propagated covariance is positive definite.

Covariance Constraints

- By definition, the covariance matrix **must** be
  - symmetric ✓✓
  - positive definite ✓
- A proper filtering recursion should always maintain these properties.
- For the linear case, where the Kalman filter is theoretically exact, do these properties hold?

Positive Definiteness: Propagation

Given that $\boldsymbol{P}_{k-1}^{+} > \mathbf{0}$ and that $\boldsymbol{F}_{k-1}$ is full rank, the noise-free propagation of covariance is guaranteed to be positive definite; therefore,

$$\boldsymbol{P}_k^{-} = \underbrace{\boldsymbol{F}_{k-1}\boldsymbol{P}_{k-1}^{+}\boldsymbol{F}_{k-1}^{T}}_{>\mathbf{0}} + \underbrace{\boldsymbol{Q}_{k-1}}_{\geq\mathbf{0}}$$

Algebraically, the propagated covariance is positive definite.

Covariance Constraints

- By definition, the covariance matrix **must** be
  - symmetric ✓✓
  - positive definite ✓
- A proper filtering recursion should always maintain these properties.
- For the linear case, where the Kalman filter is theoretically exact, do these properties hold?

Positive Definiteness: Update

Consider the measurement update that results from

$$\boldsymbol{H}_k = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1+\delta \end{bmatrix}, \quad \boldsymbol{P}_k^- = \boldsymbol{I}_3\,, \quad \text{and} \quad \boldsymbol{R}_k = \delta^2 \boldsymbol{I}_2$$

where $\delta^2 < \epsilon_{\text{roundoff}}$ but $\delta > \epsilon_{\text{roundoff}}$.

Covariance Constraints

- By definition, the covariance matrix **must** be
  - symmetric ✓✓
  - positive definite ✓
- A proper filtering recursion should always maintain these properties.
- For the linear case, where the Kalman filter is theoretically exact, do these properties hold?

Positive Definiteness: Update

In this case, although $\boldsymbol{H}_k$ clearly has a rank of 2,

$$\boldsymbol{H}_k \boldsymbol{P}_k^- \boldsymbol{H}_k^T + \boldsymbol{R}_k = \begin{bmatrix} 3 & 3+\delta \\ 3+\delta & 3+2\delta \end{bmatrix}$$

with roundoff, which is a singular matrix.

Covariance Constraints

- By definition, the covariance matrix **must** be
  - symmetric ✓✓
  - positive definite ✓✗
- A proper filtering recursion should always maintain these properties.
- For the linear case, where the Kalman filter is theoretically exact, do these properties hold?

Positive Definiteness: Update

In this case, although $\boldsymbol{H}_k$ clearly has a rank of 2,

$$\boldsymbol{H}_k \boldsymbol{P}_k^- \boldsymbol{H}_k^T + \boldsymbol{R}_k = \begin{bmatrix} 3 & 3+\delta \\ 3+\delta & 3+2\delta \end{bmatrix}$$

with roundoff, which is a singular matrix.

Covariance Constraints

- By definition, the covariance matrix **must** be
  - symmetric ✓✓
  - positive definite ✓✗
- A proper filtering recursion should always maintain these properties.
- For the linear case, where the Kalman filter is theoretically exact, do these properties hold?

Positive Definiteness: General Comment

- The update can fail because of numerical issues.
  - also true in propagation

Covariance Constraints

- By definition, the covariance matrix **must** be
  - symmetric ✓✓
  - positive definite ✗✗
- A proper filtering recursion should always maintain these properties.
- For the linear case, where the Kalman filter is theoretically exact, do these properties hold?

Positive Definiteness: General Comment

- The update can fail because of numerical issues.
  - also true in propagation

Covariance Constraints

- By definition, the covariance matrix **must** be
  - symmetric ✓✓
  - positive definite ✗✗
- A proper filtering recursion should always maintain these properties.
- For the linear case, where the Kalman filter is theoretically exact, do these properties hold?

Positive Definiteness: General Comment

- The update can fail because of numerical issues.
  - also true in propagation
- Enforcing positive definiteness is very challenging.

Covariance Constraints

- By definition, the covariance matrix **must** be
  - symmetric ✓✓
  - positive definite ✗✗
- A proper filtering recursion should always maintain these properties.
- For the linear case, where the Kalman filter is theoretically exact, do these properties hold?

Positive Definiteness: General Comment

- The update can fail because of numerical issues.
  - also true in propagation
- Enforcing positive definiteness is very challenging.
- Can be mitigated with factorization-based filtering methods.

Loss of Positive Definiteness

- Positive definiteness can be lost during filtering

Loss of Positive Definiteness

- Positive definiteness can be lost during filtering
  - Large prior uncertainty + precise measurements
  - Condition number of the covariance matrix

Loss of Positive Definiteness

- Positive definiteness can be lost during filtering
  - Large prior uncertainty + precise measurements
    - commonly encountered in landing navigation
    - uncertainties "grow" unabated for long periods of time
    - precise data, such as altimetry, becomes available
  - Condition number of the covariance matrix

Loss of Positive Definiteness

- Positive definiteness can be lost during filtering
  - Large prior uncertainty + precise measurements
    - commonly encountered in landing navigation
    - uncertainties "grow" unabated for long periods of time
    - precise data, such as altimetry, becomes available
  - Condition number of the covariance matrix
    - commonly encountered in large-state filters
    - estimate position, velocity, attitude, biases, etc.
    - units of states become important, but want to be agnostic to this

Loss of Positive Definiteness

- Positive definiteness can be lost during filtering
  - Large prior uncertainty + precise measurements
    - commonly encountered in landing navigation
    - uncertainties "grow" unabated for long periods of time
    - precise data, such as altimetry, becomes available
  - Condition number of the covariance matrix
    - commonly encountered in large-state filters
    - estimate position, velocity, attitude, biases, etc.
    - units of states become important, but want to be agnostic to this
- Factorization-based filtering mitigates loss of positive definiteness
  - Avoid working with covariance
  - Work with factors of covariance
  - Establish propagation/update equations for the factors

Loss of Positive Definiteness

- Positive definiteness can be lost during filtering
  - Large prior uncertainty + precise measurements
    - commonly encountered in landing navigation
    - uncertainties "grow" unabated for long periods of time
    - precise data, such as altimetry, becomes available
  - Condition number of the covariance matrix
    - commonly encountered in large-state filters
    - estimate position, velocity, attitude, biases, etc.
    - units of states become important, but want to be agnostic to this
- Factorization-based filtering mitigates loss of positive definiteness
  - Avoid working with covariance
  - Work with factors of covariance
  - Establish propagation/update equations for the factors
  - Examples:
    - UDU
    - Cholesky

Loss of Positive Definiteness

- Positive definiteness can be lost during filtering
  - Large prior uncertainty + precise measurements
    - commonly encountered in landing navigation
    - uncertainties "grow" unabated for long periods of time
    - precise data, such as altimetry, becomes available
  - Condition number of the covariance matrix
    - commonly encountered in large-state filters
    - estimate position, velocity, attitude, biases, etc.
    - units of states become important, but want to be agnostic to this
- Factorization-based filtering mitigates loss of positive definiteness
  - Avoid working with covariance
  - Work with factors of covariance
  - Establish propagation/update equations for the factors
  - Examples:
    - ✓ UDU (more details in paper)
    - ✓ Cholesky

Methods of Factorization-based Filtering

- Originated with Potter's idea of the square-root filter
  - Replace covariance with Cholesky factor
  - Propagate and update Cholesky factor
  - No process noise + scalar measurements

Methods of Factorization-based Filtering

- Originated with Potter's idea of the square-root filter
  - Replace covariance with Cholesky factor
  - Propagate and update Cholesky factor
  - No process noise + scalar measurements
- UDU Factorization
  - Factor $\boldsymbol{P}$ as $\boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^T$
  - $\boldsymbol{U}$ is upper diagonal with ones on the diagonal
  - $\boldsymbol{D}$ is diagonal
  - Propagate and update $\boldsymbol{U}$ and $\boldsymbol{D}$

Methods of Factorization-based Filtering

- Originated with Potter's idea of the square-root filter
  - Replace covariance with Cholesky factor
  - Propagate and update Cholesky factor
  - No process noise + scalar measurements
- UDU Factorization
  - Factor $\boldsymbol{P}$ as $\boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^T$
  - $\boldsymbol{U}$ is upper diagonal with ones on the diagonal
  - $\boldsymbol{D}$ is diagonal
  - Propagate and update $\boldsymbol{U}$ and $\boldsymbol{D}$
    - Modified Weighted Gram-Schmidt orthogonalization
    - Carlson rank-1 updates

Methods of Factorization-based Filtering

- Originated with Potter's idea of the square-root filter
  - Replace covariance with Cholesky factor
  - Propagate and update Cholesky factor
  - No process noise + scalar measurements
- UDU Factorization
  - Factor $\boldsymbol{P}$ as $\boldsymbol{UDU}^T$
  - $\boldsymbol{U}$ is upper diagonal with ones on the diagonal
  - $\boldsymbol{D}$ is diagonal
  - Propagate and update $\boldsymbol{U}$ and $\boldsymbol{D}$
    - Modified Weighted Gram-Schmidt orthogonalization
    - Carlson rank-1 updates
- Cholesky Factorization
  - Factor $\boldsymbol{P}$ as $\boldsymbol{SS}^T$
  - $\boldsymbol{S}$ is lower triangular
  - Propagate and update $\boldsymbol{S}$

Methods of Factorization-based Filtering

- Originated with Potter's idea of the square-root filter
  - Replace covariance with Cholesky factor
  - Propagate and update Cholesky factor
  - No process noise + scalar measurements
- UDU Factorization
  - Factor $\boldsymbol{P}$ as $\boldsymbol{UDU}^T$
  - $\boldsymbol{U}$ is upper diagonal with ones on the diagonal
  - $\boldsymbol{D}$ is diagonal
  - Propagate and update $\boldsymbol{U}$ and $\boldsymbol{D}$
    - Modified Weighted Gram-Schmidt orthogonalization
    - Carlson rank-1 updates
- Cholesky Factorization
  - Factor $\boldsymbol{P}$ as $\boldsymbol{SS}^T$
  - $\boldsymbol{S}$ is lower triangular
  - Propagate and update $\boldsymbol{S}$
    - QR decomposition
    - Cholesky rank-m downdate

The Cholesky Square-Root Filter

- For the nonlinear state-space model

$$\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1}) + \boldsymbol{w}_{k-1}$$
$$\boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{v}_k$$

The Cholesky Square-Root Filter

- For the nonlinear state-space model

$$\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1}) + \boldsymbol{w}_{k-1}$$
$$\boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{v}_k$$

- The Cholesky square-root filter is given by the recursion:

Mean Prop.	$\boldsymbol{m}_k^- = \boldsymbol{f}(\boldsymbol{m}_{k-1}^+)$
SRF Prop.	$\boldsymbol{S}_k^- = \mathrm{qr}\{[\boldsymbol{F}_{k-1}\boldsymbol{S}_{k-1}^+ \mid \boldsymbol{T}_{k-1}]^T\}^T$
Innov. SRF	$\boldsymbol{Y}_k = \mathrm{qr}\{[\boldsymbol{H}_k\boldsymbol{S}_k^- \mid \boldsymbol{L}_k]^T\}^T$
Cross Cov.	$\boldsymbol{C}_k = \boldsymbol{S}_k^-\left[\boldsymbol{H}_k\boldsymbol{S}_k^-\right]^T$
Update Factors	$\boldsymbol{U}_k = \boldsymbol{C}_k(\boldsymbol{Y}_k^-)^T$
Kalman Gain	$\boldsymbol{K}_k = \boldsymbol{U}_k\boldsymbol{Y}_k^{-1}$
Mean Update	$\boldsymbol{m}_k^+ = \boldsymbol{m}_k^- + \boldsymbol{K}_k(\boldsymbol{z}_k - \boldsymbol{h}(\boldsymbol{m}_k^-))$
SRF Update	$\boldsymbol{S}_k^+ = \mathrm{cholupdate}\{(\boldsymbol{S}_k^-)^T, \boldsymbol{U}_k, -1\}^T$

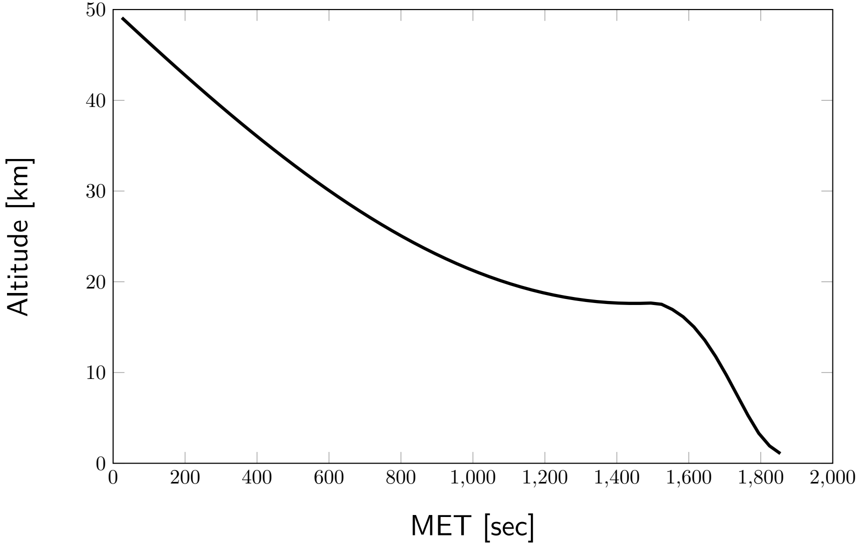
Comments on the Methods

- Cholesky Factorization
  - guarantees symmetry of the covariance matrix
  - can guarantee positive definiteness
  - requires square root operations
  - quite simple, structurally

Comments on the Methods

- Cholesky Factorization
  - guarantees symmetry of the covariance matrix
  - can guarantee positive definiteness
  - requires square root operations
  - quite simple, structurally
- UDU Factorization
  - guarantees symmetry of the covariance matrix
  - simple check for positive definiteness
  - does not require square root operations
  - more complicated, structurally

Trajectory


Altitude [km]
MET [sec]
0
10
20
30
40
50
0
200
400
600
800
1,000
1,200
1,400
1,600
1,800
2,000

IMU Model

- **Inertial Measurement Unit** output is given by

$$\Delta \boldsymbol{v}_{m,k} = \Delta \boldsymbol{v}_k + \boldsymbol{b}_v + \boldsymbol{w}_{v,k}$$
$$\Delta \boldsymbol{\theta}_{m,k} = \Delta \boldsymbol{\theta}_k + \boldsymbol{b}_\theta + \boldsymbol{w}_{\theta,k}$$

  where
  - $\Delta \boldsymbol{v}_k$ is the true, integrated, non-gravitational acceleration
  - $\Delta \boldsymbol{\theta}_k$ is the true, integrated angular velocity

- Sensor specifications

  Accelerometer
  - Bias $(1\sigma) = 300\mu g$
  - Noise $(1\sigma) = 35\mu g/\sqrt{\mathrm{Hz}}$
  - Frequency $= 40$ Hz
  - Active: always

  Gyro
  - Bias $(1\sigma) = 1°/\mathrm{hr}$
  - Noise $(1\sigma) = 0.07°/\sqrt{\mathrm{hr}}$
  - Frequency $= 40$ Hz
  - Active: always

Altimeter Model

- **Spherical Altitude** measurement is given by

$$z_k = (\|\boldsymbol{r}^i_{\text{alt},k}\| - r_{\text{sph}}) + b_{\text{alt}} + v_{\text{alt},k}$$

where

$$\boldsymbol{r}^i_{\text{alt},k} = \boldsymbol{r}^i_{\text{imu},k} + \boldsymbol{T}^i_{c,k}\boldsymbol{r}^c_{\text{alt/imu}}$$

- Sensor specifications
    - Bias $(1\sigma) = 0.5$ m
    - Noise $(1\sigma) = [500,\ 5]$ m
    - Frequency $= 10$ Hz
    - Active: $h \leq 15$ km

Star Camera Model

- **Quaternion Star Camera** measurement is given by

$$\bar{\boldsymbol{z}}_k = \bar{\boldsymbol{q}}_{\text{err},k} \otimes \bar{\boldsymbol{q}}_c^{\text{sc}} \otimes \bar{\boldsymbol{q}}_{i,k}^c$$

where

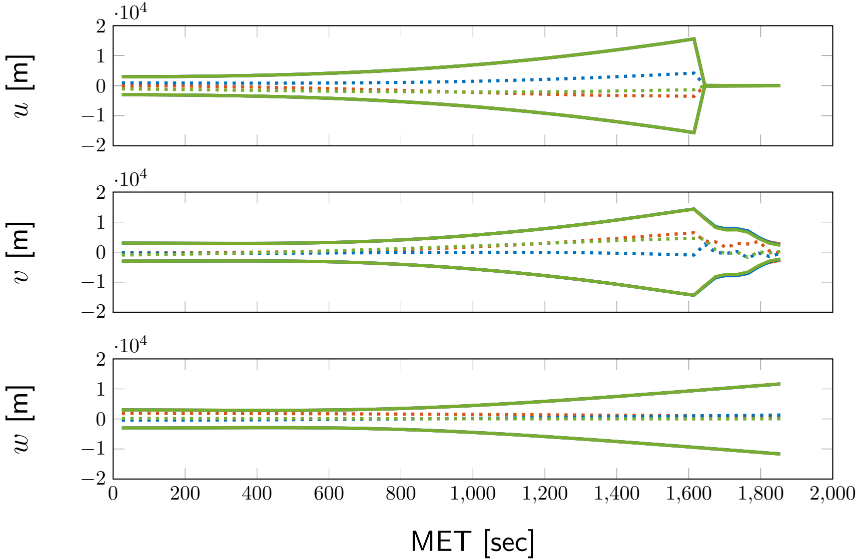
$$\bar{\boldsymbol{q}}_{\text{err},k} = \begin{bmatrix} \sin\left(\frac{1}{2}\|\boldsymbol{\theta}_{\text{err},k}\|\right) \frac{\boldsymbol{\theta}_{\text{err},k}}{\|\boldsymbol{\theta}_{\text{err},k}\|} \\ \cos\left(\frac{1}{2}\|\boldsymbol{\theta}_{\text{err},k}\|\right) \end{bmatrix} \quad \text{and} \quad \boldsymbol{\theta}_{\text{err},k} = \boldsymbol{b}_{sc} + \boldsymbol{v}_{sc,k}$$

- Sensor Specifications
  - Bias $(1\sigma) = 10''$
  - Noise $(1\sigma) = 30''$
  - Frequency $= 1$ Hz
  - Active: when <u>not</u> thrusting

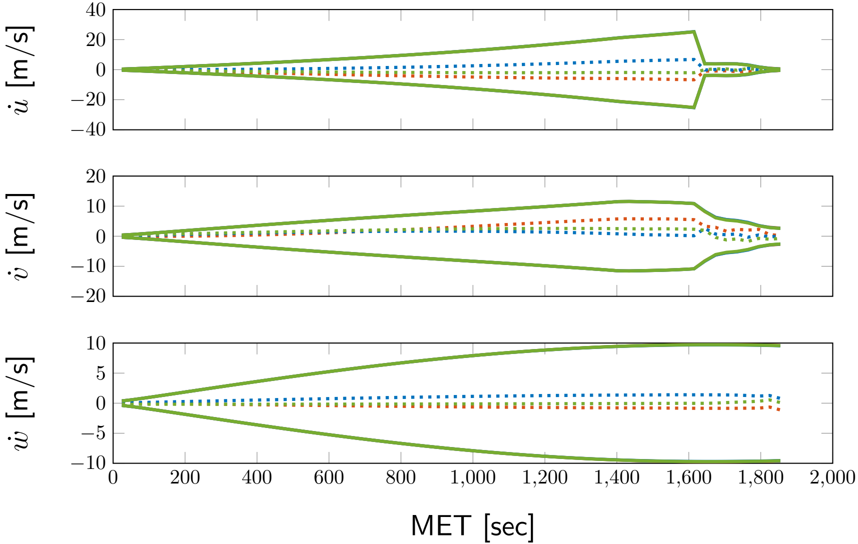
Monte Carlo Comparison

- Assess statistical consistency
    - 1000 Monte Carlo trials
    - Resample initial states and noises
    - Compute sample covariance
    - Compare to single run performance
    - Look at full covariance, UDU factorized, and Cholesky factorized filters

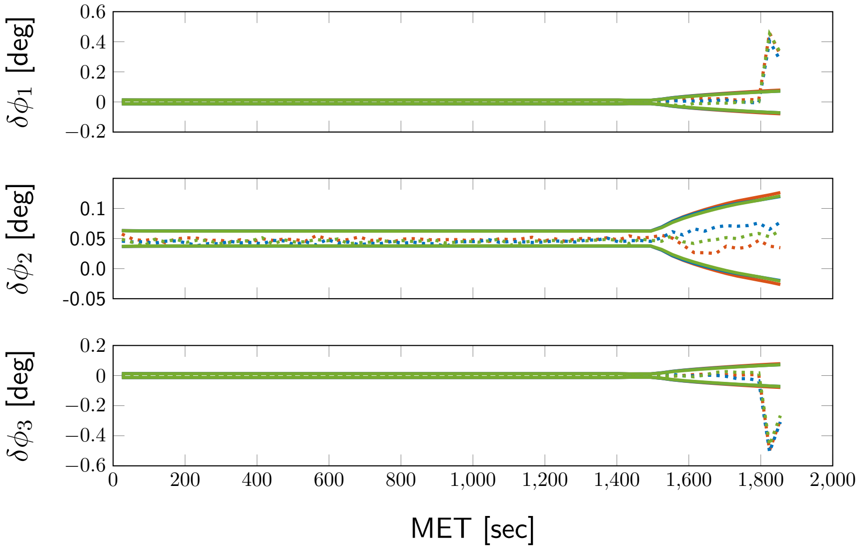
Monte Carlo: Position



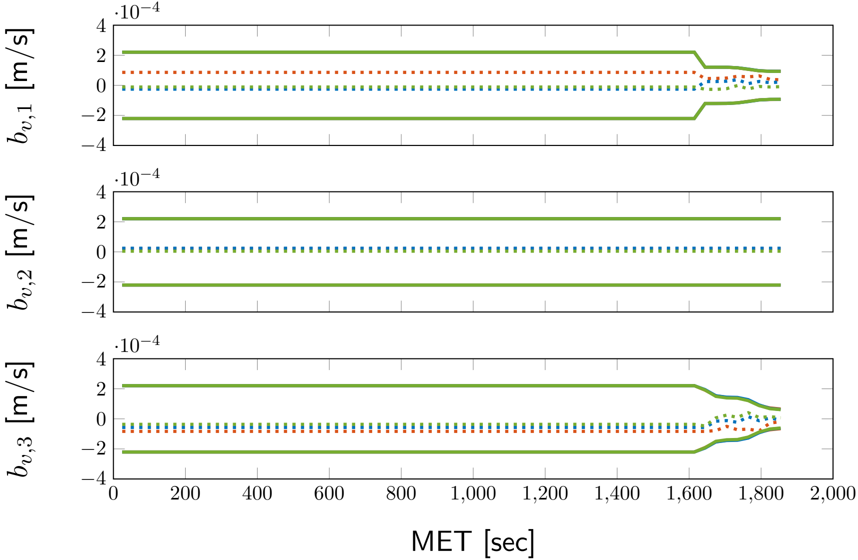
Monte Carlo: Velocity



Monte Carlo: Attitude



Monte Carlo: Accel. Bias



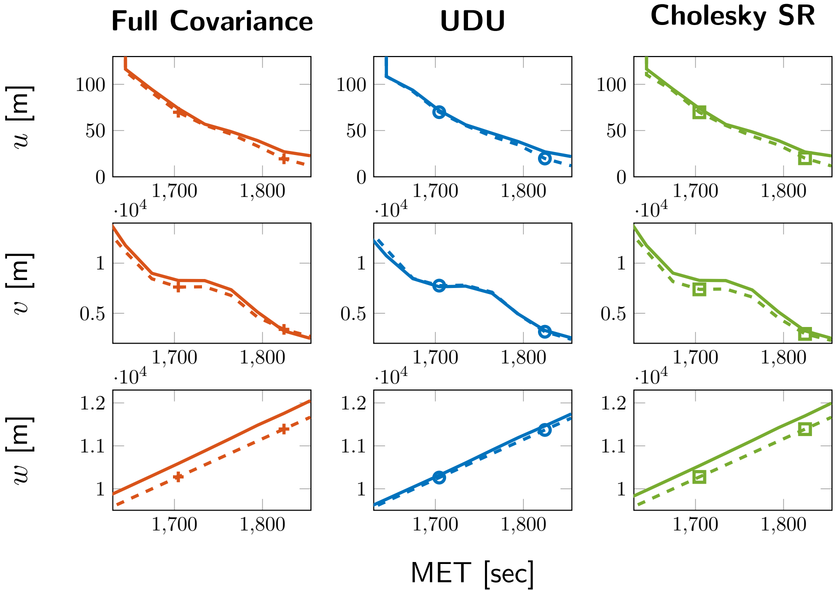
Monte Carlo Comparison

- Assess statistical consistency
  - 1000 Monte Carlo trials
  - Resample initial states and noises
  - Compute sample covariance
  - Compare to single run performance
  - Look at full covariance, UDU factorized, and Cholesky factorized filters
- Observations
  - Some full covariance trials failed
  - All UDU and Cholesky factorized trials successful
  - Translational uncertainty growth before altimeter turns on
  - Rotational uncertainty growth after star camera turns off
    - errors caused by sampling

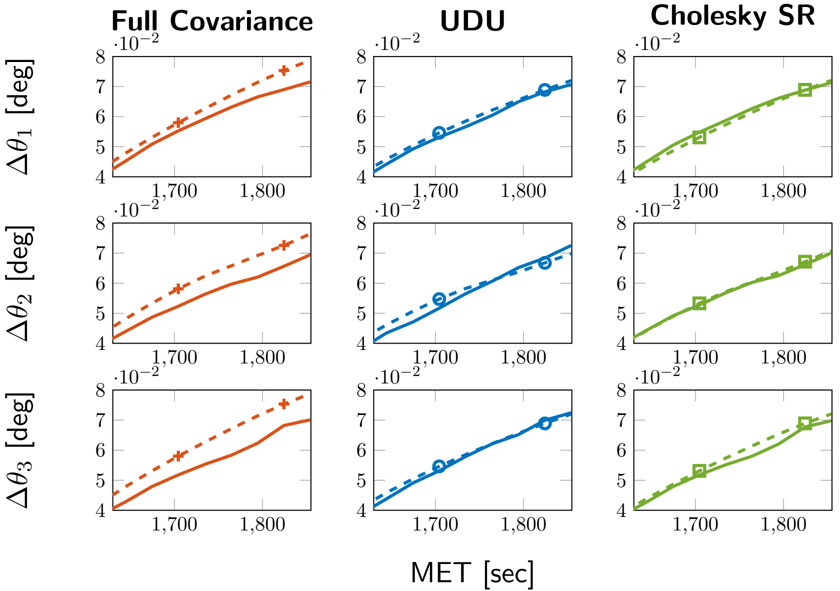
Terminal Descent Analysis

- More in-depth analysis during terminal descent
  - Same simulation, same configuration
  - Enhanced view in terminal descent

Grid Comparison: Position



Grid Comparison: Attitude



Terminal Descent Analysis

- More in-depth analysis during terminal descent
  - Same simulation, same configuration
  - Enhanced view in terminal descent
- Observations

Terminal Descent Analysis

- More in-depth analysis during terminal descent
  - Same simulation, same configuration
  - Enhanced view in terminal descent
- Observations
  - Full covariance
    - Conservative in position uncertainty
    - Overly confident in attitude uncertainty
    - Failures due to loss of positive definiteness

Terminal Descent Analysis

- More in-depth analysis during terminal descent
  - Same simulation, same configuration
  - Enhanced view in terminal descent
- Observations
  - Full covariance
    - Conservative in position uncertainty
    - Overly confident in attitude uncertainty
    - Failures due to loss of positive definiteness
  - UDU factorized
    - Back and forth in position uncertainty
    - Back and forth in attitude uncertainty
    - No failures due to loss of positive definiteness

Terminal Descent Analysis

- More in-depth analysis during terminal descent
  - Same simulation, same configuration
  - Enhanced view in terminal descent
- Observations
  - Full covariance
    - Conservative in position uncertainty
    - Overly confident in attitude uncertainty
    - Failures due to loss of positive definiteness
  - UDU factorized
    - Back and forth in position uncertainty
    - Back and forth in attitude uncertainty
    - No failures due to loss of positive definiteness
  - Cholesky factorized
    - Conservative in position uncertainty
    - Back and forth in attitude uncertainty
    - No failures due to loss of positive definiteness

Conclusions

- Comparison of different filtering approaches for descent navigation

Conclusions

- Comparison of different filtering approaches for descent navigation
  - Full covariance
    - brute-force symmetrization
    - no guarantee on positive definiteness

Conclusions

- Comparison of different filtering approaches for descent navigation
    - Full covariance
        - brute-force symmetrization
        - no guarantee on positive definiteness
    - UDU factorization
        - guaranteed symmetry
        - easy check for positive definiteness

Conclusions

- Comparison of different filtering approaches for descent navigation
  - Full covariance
    - brute-force symmetrization
    - no guarantee on positive definiteness
  - UDU factorization
    - guaranteed symmetry
    - easy check for positive definiteness
  - Cholesky factorization
    - guaranteed symmetry
    - can guarantee positive definiteness

Conclusions

- Comparison of different filtering approaches for descent navigation
  - Full covariance
    - brute-force symmetrization
    - no guarantee on positive definiteness
  - UDU factorization
    - guaranteed symmetry
    - easy check for positive definiteness
  - Cholesky factorization
    - guaranteed symmetry
    - can guarantee positive definiteness
- When processing IMU, altimeter, and star camera data
  - observed failures in full covariance filters
  - similar consistency performance in UDU and Cholesky

Conclusions

- Comparison of different filtering approaches for descent navigation
  - Full covariance
    - brute-force symmetrization
    - no guarantee on positive definiteness
  - UDU factorization
    - guaranteed symmetry
    - easy check for positive definiteness
  - Cholesky factorization
    - guaranteed symmetry
    - can guarantee positive definiteness
- When processing IMU, altimeter, and star camera data
  - observed failures in full covariance filters
  - similar consistency performance in UDU and Cholesky
- Which filter should you use?
  - vector vs. scalar processing of data
  - computational resources available

Acknowledgments

This work was partially supported by a NASA Space Technology Research Fellowship and through Grant NNX16AF11A.

The authors would also like to acknowledge the many helpful discussions with Drs. Chris D'Souza and Renato Zanetti of NASA Johnson Space Center.

Questions?