Absolute scale velocity determination combining visual and inertial measurements for micro aerial vehicles

Jacques Kaiser
Supervised by Dr. Agostino Martinelli

INRIA

June 24, 2015

Section 1

Sensor fusion

Micro aerial vehicles

Micro aerial vehicles

Micro aerial vehicles

A basic state vector: $X = \begin{bmatrix} \textit{position} \\ \textit{velocity} \\ \textit{orientation} \end{bmatrix}$

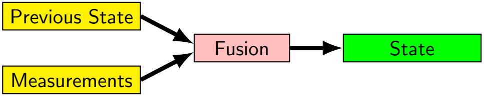
Micro aerial vehicles

A basic state vector: $X = \begin{bmatrix} \textit{position} \\ \textit{velocity} \\ \textit{orientation} \end{bmatrix}$

The goal of sensor fusion is to recover the state X

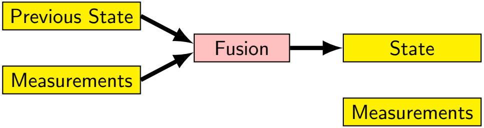
Visual-inertial sensor fusion

Filter based method



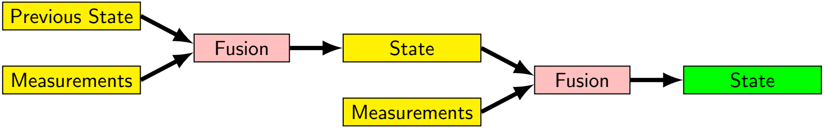
Visual-inertial sensor fusion

Filter based method



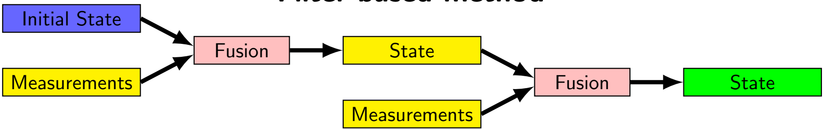
Visual-inertial sensor fusion

Filter based method



Visual-inertial sensor fusion

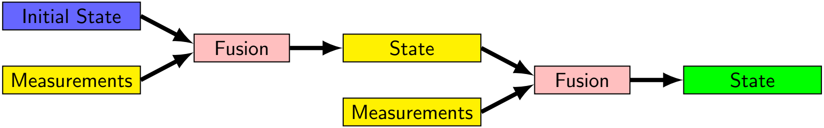
Filter based method



How to recover the **initial state**?

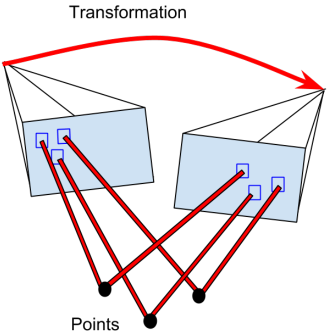
Visual-inertial sensor fusion

Filter based method

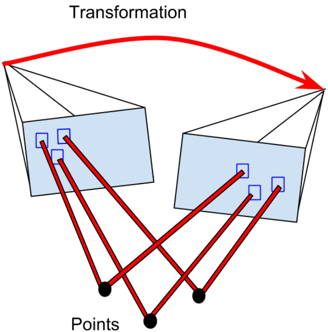


How to recover the **initial state**?
We need a **deterministic solution**

Deterministic solutions in Computer Vision



Deterministic solutions in Computer Vision



But the relative **translation** and **distance to features** are recovered only **up to scale**

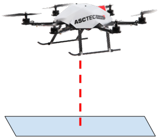
Absolute scale from visual measurements

How big is this building?

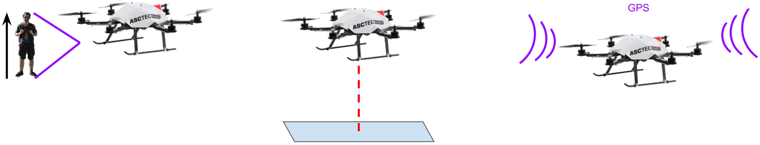
Absolute scale from visual measurements

Methods to recover the absolute scale

Methods to recover the absolute scale

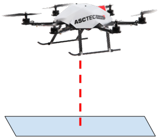


Methods to recover the absolute scale



Methods to recover the absolute scale

Not suited to unknown environments



Not precise, works only in hover

Does not work in GPS denied environments

Inertial Measurement Unit (IMU)

The IMU consists of two sensors providing **physical quantities**:

- Accelerometer: linear acceleration (and gravity) (m/s^2);
- Gyroscope: angular velocity (rad/s).

Back to the title

Absolute scale velocity determination combining visual and inertial measurements for micro aerial vehicles

Back to the title

Absolute scale velocity determination combining visual and inertial measurements for micro aerial vehicles

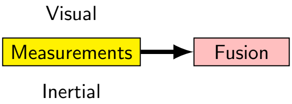
Visual

Measurements

Inertial

Back to the title

Absolute scale velocity determination combining visual and inertial measurements for micro aerial vehicles



Back to the title

Absolute scale velocity determination combining visual and inertial measurements for micro aerial vehicles

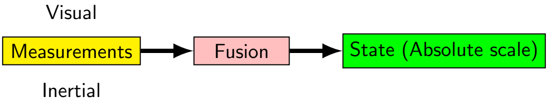


Table of Contents

Sensor fusion
- Filter based fusion
- Deterministic solution
- Absolute scale

The closed-form solution
- Linear system
- Identifying performance bottlenecks
- Estimating the gyroscope bias
- Validation

Conclusion

Section 2

The closed-form solution

The Closed-Form Solution - 2014

Requires:

- Calibrated camera;
- Inertial Measurement Unit (IMU);
- External Camera IMU transformation.

Output:

- Initial velocity;
- Distance to point-features;
- Attitude.

The Closed-Form Solution - 2014

Requires:

- Calibrated camera;
- Inertial Measurement Unit (IMU);
- External Camera IMU transformation.

Output:

- Initial velocity;
- Distance to point-features;
- Attitude.

$$S_j = \lambda_1^i \mu_1^i - V t_j - G \frac{t_j^2}{2} - \lambda_j^i \mu_j^i$$

The Closed-Form Solution - 2014

Requires:

- Calibrated camera;
- Inertial Measurement Unit (IMU);
- External Camera IMU transformation.

Output:

- Initial velocity;
- Distance to point-features;
- Attitude.

$$S_j = \lambda_1^i \mu_1^i - V t_j - G \frac{t_j^2}{2} - \lambda_j^i \mu_j^i$$

The Overconstrained Linear System

$$S_j = \lambda_1^i \mu_1^i - V t_j - G \frac{t_j^2}{2} - \lambda_j^i \mu_j^i$$

Valid for every point-features i at any time t_j:

The Overconstrained Linear System

$$S_j = \lambda_1^i \mu_1^i - V t_j - G \frac{t_j^2}{2} - \lambda_j^i \mu_j^i$$

Valid for every point-features i at any time t_j:

$$\left[\begin{array}{c} S_2 = \lambda_1^1 \mu_1^1 - V t_2 - G \frac{t_2^2}{2} - \lambda_2^1 \mu_2^1 \\ S_2 = \lambda_1^2 \mu_1^2 - V t_2 - G \frac{t_2^2}{2} - \lambda_2^2 \mu_2^2 \\ \vdots \\ S_3 = \lambda_1^1 \mu_1^1 - V t_3 - G \frac{t_3^2}{2} - \lambda_3^1 \mu_3^1 \\ \vdots \\ S_N = \lambda_1^{n_i} \mu_1^{n_i} - V t_N - G \frac{t_N^2}{2} - \lambda_N^{n_i} \mu_N^{n_i} \end{array}\right.$$

The Overconstrained Linear System

$$S_j = \lambda_1^i \mu_1^i - V t_j - G \frac{t_j^2}{2} - \lambda_j^i \mu_j^i$$

Valid for every point-features i at any time t_j:

$$\left[\begin{array}{c} S_2 = \lambda_1^1 \mu_1^1 - V t_2 - G \frac{t_2^2}{2} - \lambda_2^1 \mu_2^1 \\ S_2 = \lambda_1^2 \mu_1^2 - V t_2 - G \frac{t_2^2}{2} - \lambda_2^2 \mu_2^2 \\ \vdots \\ S_3 = \lambda_1^1 \mu_1^1 - V t_3 - G \frac{t_3^2}{2} - \lambda_3^1 \mu_3^1 \\ \vdots \\ S_N = \lambda_1^{n_i} \mu_1^{n_i} - V t_N - G \frac{t_N^2}{2} - \lambda_N^{n_i} \mu_N^{n_i} \end{array}\right.$$

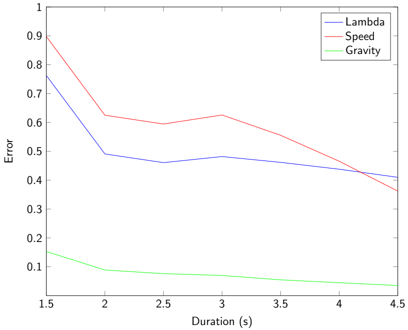
$$\Xi X = S$$

Problem: not robust in practice

> "**A closed-form solution** *for state estimation with a visual-inertial system that does not require initialization was presented. However, this approach is* **not suitable for systems that rely on noisy sensor data**"
>
> — Matthias Faessler, ICRA 2015

Problem: not robust in practice

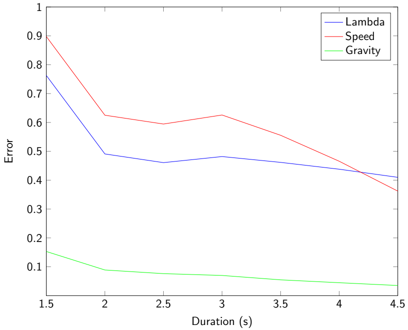
> ”**A closed-form solution** *for state estimation with a visual-inertial system that does not require initialization was presented. However, this approach is* **not suitable for systems that rely on noisy sensor data**”
>
> — Matthias Faessler, ICRA 2015



Problem: not robust in practice

"**A closed-form solution** *for state estimation with a visual-inertial system that does not require initialization was presented. However, this approach is* **not suitable for systems that rely on noisy sensor data**"

— Matthias Faessler, ICRA 2015



50% relative error on speed and distance estimation

Improving the performance

What makes the estimations so bad?

Possible bottlenecks

- Motion;

Improving the performance

What makes the estimations so bad?

Possible bottlenecks

- Motion;
- Observed features;

Improving the performance

What makes the estimations so bad?

Possible bottlenecks

- Motion;
- Observed features;
- Noisy sensors.

Improving the performance

What makes the estimations so bad?

Possible bottlenecks

- Motion;
- Observed features;
- Noisy sensors.

Sensors provide measurements affected by a Gaussian noise:

$$N(\mu + B, \sigma^2)$$

Improving the performance

What makes the estimations so bad?

Possible bottlenecks

- Motion;
- Observed features;
- Noisy sensors.

Sensors provide measurements affected by a Gaussian noise:

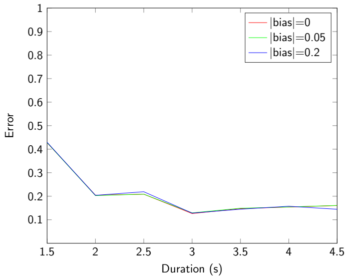
$$N(\mu + B, \sigma^2)$$

We considered:

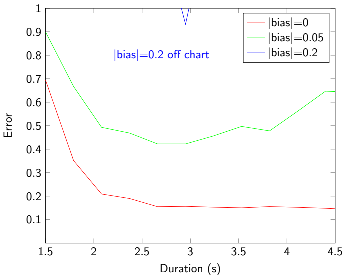
- Biased accelerometer;
- Biased gyroscope.

Biased inertial measurements

Speed estimation error



Varying accelerometer bias



Varying gyroscope bias

Estimating the gyroscope bias

- When solving $\Xi X = S$, we are solving $\operatorname{argmin}_X ||\Xi X - S||^2$;

Estimating the gyroscope bias

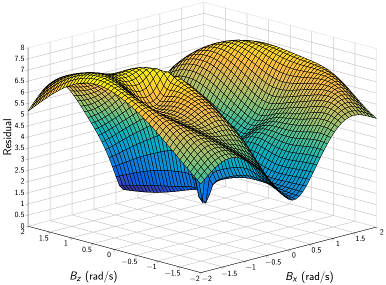
- When solving $\Xi X = S$, we are solving $\text{argmin}_X \, ||\Xi X - S||^2$;
- Unfortunately, we can not express the gyroscope bias linearly;

Estimating the gyroscope bias

- When solving $\Xi X = S$, we are solving $\text{argmin}_X \, ||\Xi X - S||^2$;
- Unfortunately, we can not express the gyroscope bias linearly;
- Alternative: non-linear minimization

$$\underset{{\color{red}B},X}{\text{argmin}} \, ||\Xi X - S||^2$$

With B the gyroscope bias, Ξ and S computed with respect to B

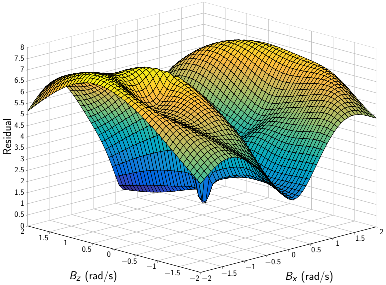


Estimating the gyroscope bias

- When solving $\Xi X = S$, we are solving $\text{argmin}_X \, ||\Xi X - S||^2$;
- Unfortunately, we can not express the gyroscope bias linearly;
- Alternative: non-linear minimization

$$\underset{B,X}{\text{argmin}} \, ||\Xi X - S||^2$$

With B the gyroscope bias, Ξ and S computed with respect to B



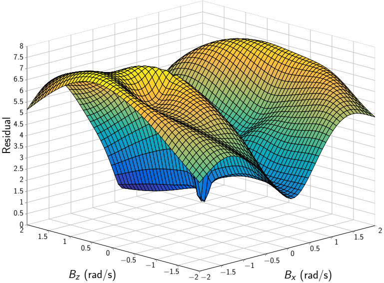
Symmetry induced by the strong weight of the gravity

Estimating the gyroscope bias

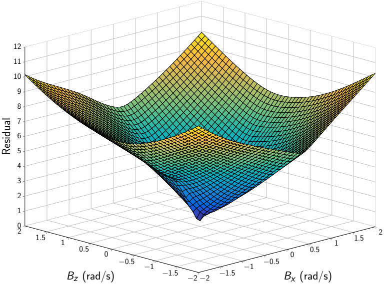
- When solving $\Xi X = S$, we are solving $\text{argmin}_X \, ||\Xi X - S||^2$;
- Unfortunately, we can not express the gyroscope bias linearly;
- Alternative: non-linear minimization

$$\underset{B,X}{\text{argmin}} \, ||\Xi X - S||^2 + \lambda \times |B|$$

With B the gyroscope bias, Ξ and S computed with respect to B

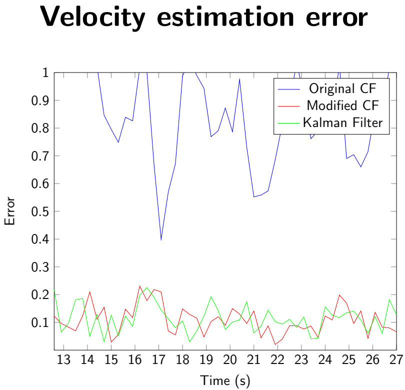


Symmetry induced by the strong weight of the gravity



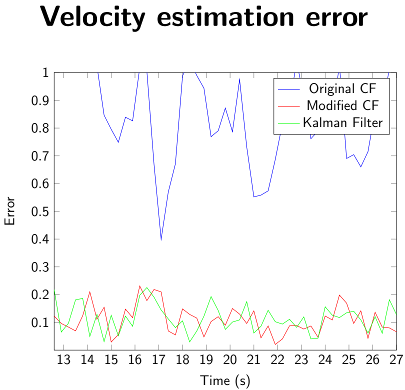
With regularization $\lambda = 3$

Results



Velocity estimation error
Error
Time (s)
Original CF
Modified CF
Kalman Filter
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
13 14 15 16 17 18 19 20 21 22 23 24 25 26 27

Results



Moreover, our technique also provides the gyroscope bias

Section 3

Conclusion

Conclusion

1. Closed-form solution for visual-inertial fusion;
   - No initialization required;
   - Recovers the absolute scale.

Conclusion

1. Closed-form solution for visual-inertial fusion;
   - No initialization required;
   - Recovers the absolute scale.
2. Does not work well in practice;

Conclusion

1. Closed-form solution for visual-inertial fusion;
   - No initialization required;
   - Recovers the absolute scale.
2. Does not work well in practice;
3. Gyroscope bias is a performance bottleneck;

Conclusion

1. Closed-form solution for visual-inertial fusion;
   - No initialization required;
   - Recovers the absolute scale.
2. Does not work well in practice;
3. Gyroscope bias is a performance bottleneck;
4. **Method to estimate this bias**;

Conclusion

1. Closed-form solution for visual-inertial fusion;
   - No initialization required;
   - Recovers the absolute scale.
2. Does not work well in practice;
3. Gyroscope bias is a performance bottleneck;
4. **Method to estimate this bias**;
5. Works well in practice;

Conclusion

1. Closed-form solution for visual-inertial fusion;
   - No initialization required;
   - Recovers the absolute scale.
2. Does not work well in practice;
3. Gyroscope bias is a performance bottleneck;
4. **Method to estimate this bias**;
5. Works well in practice;
6. Also provides the gyroscope bias.

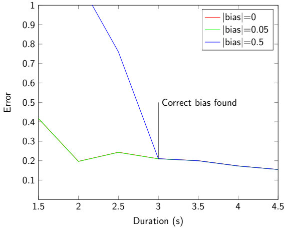
Potential PhD

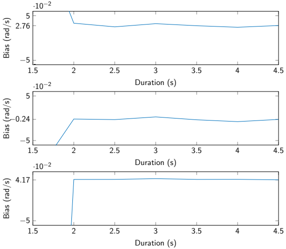
Implementation on a real platform

Thank you for your attention,
Any question?

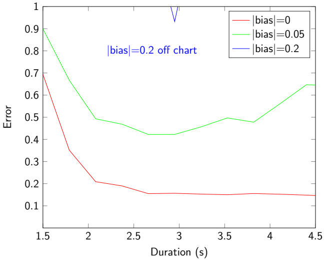
Gyroscope bias estimation

Optimized Closed-form solution





Original Closed-form solution



Numerical stability

$$\left[\begin{array}{lcl} S_j & = & \lambda_1^1\mu_1^1 - Vt_j - G\frac{t_j^2}{2} - \lambda_j^1\mu_j^1 \\ 0_3 & = & \lambda_1^1\mu_1^1 - \lambda_j^1\mu_j^1 - \lambda_1^i\mu_1^i + \lambda_j^i\mu_j^i \end{array}\right.$$

$$S_j = \lambda_1^i\mu_1^i - Vt_j - G\frac{t_j^2}{2} - \lambda_j^i\mu_j^i$$