University of Zurich[UZH] | ETH zürich

Institute of Informatics – Institute of Neuroinformatics

ROBOTICS & PERCEPTION GROUP

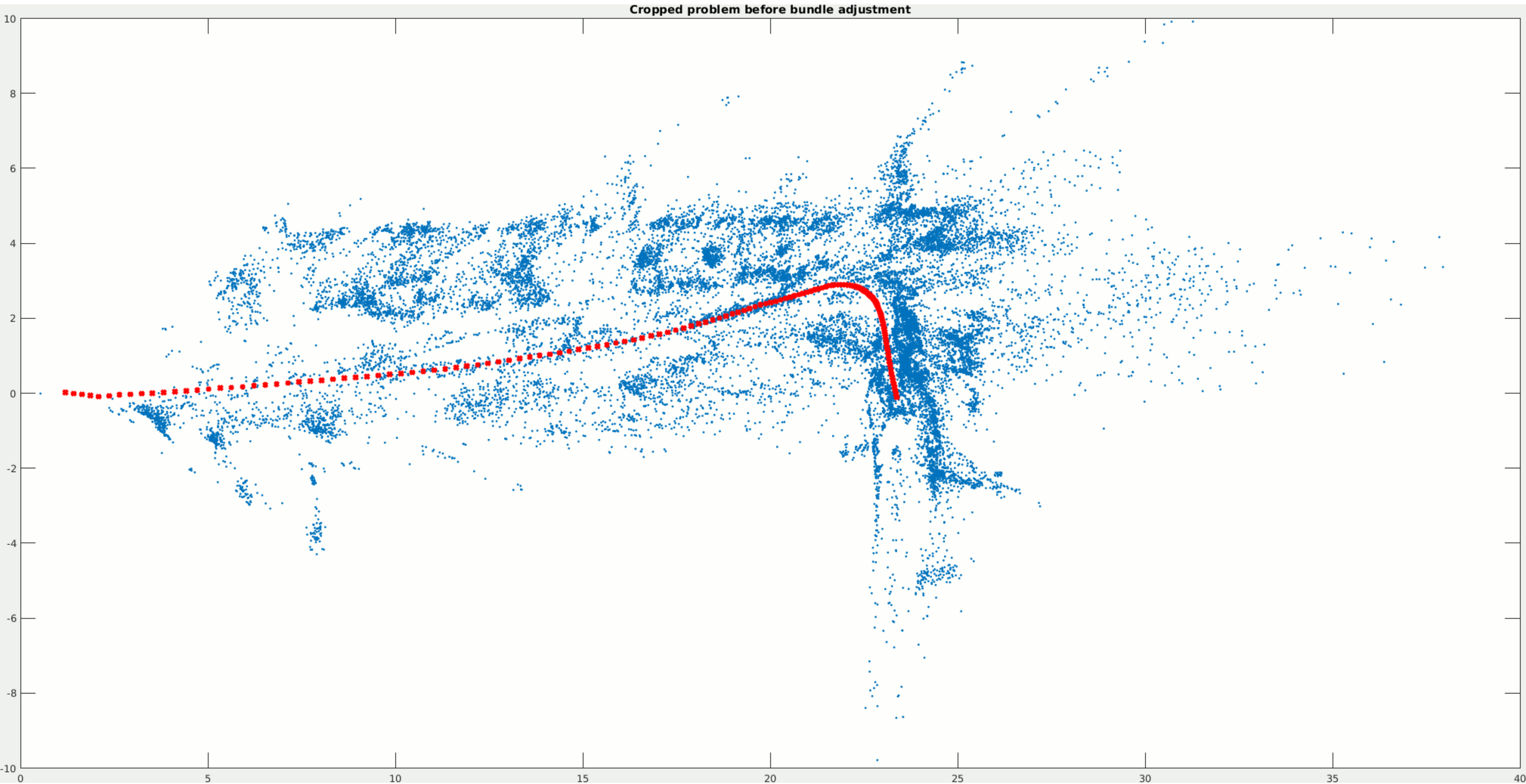
Lecture 13
Visual Inertial Fusion

Davide Scaramuzza

http://rpg.ifi.uzh.ch/

Lab Exercise 9 – Today afternoon

- Room ETH HG E 1.1 from 13:15 to 15:00
- Work description: Bundle Adjustment

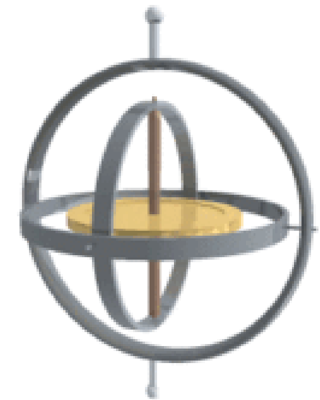


Outline

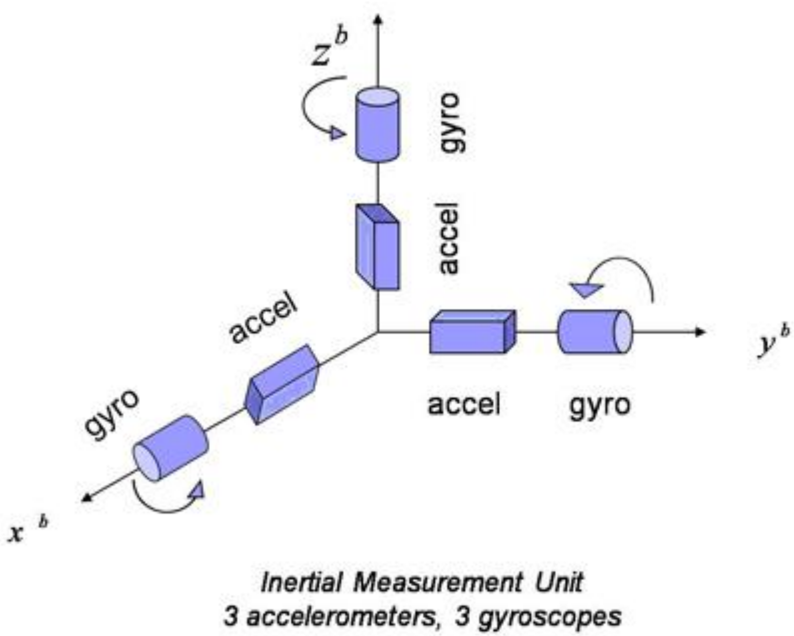
- Introduction
- IMU model and Camera-IMU system
- Different paradigms
  - Closed-form solution
  - Filtering approaches
  - Smoothing methods
    - Fixed-lag Smoothing (aka sliding window estimators)
    - Full smoothing methods
- Camera-IMU extrinsic calibration and Synchronization

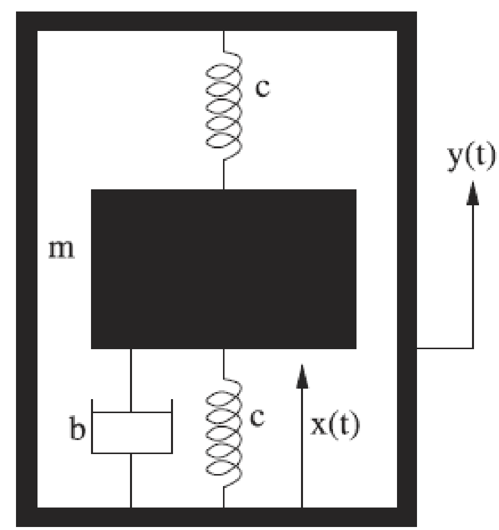
What is an IMU?

- **I**nertial **M**easurement **U**nit
  - Gyroscope: Angular velocity
  - Accelerometer: Linear Accelerations



Mechanical Gyroscope





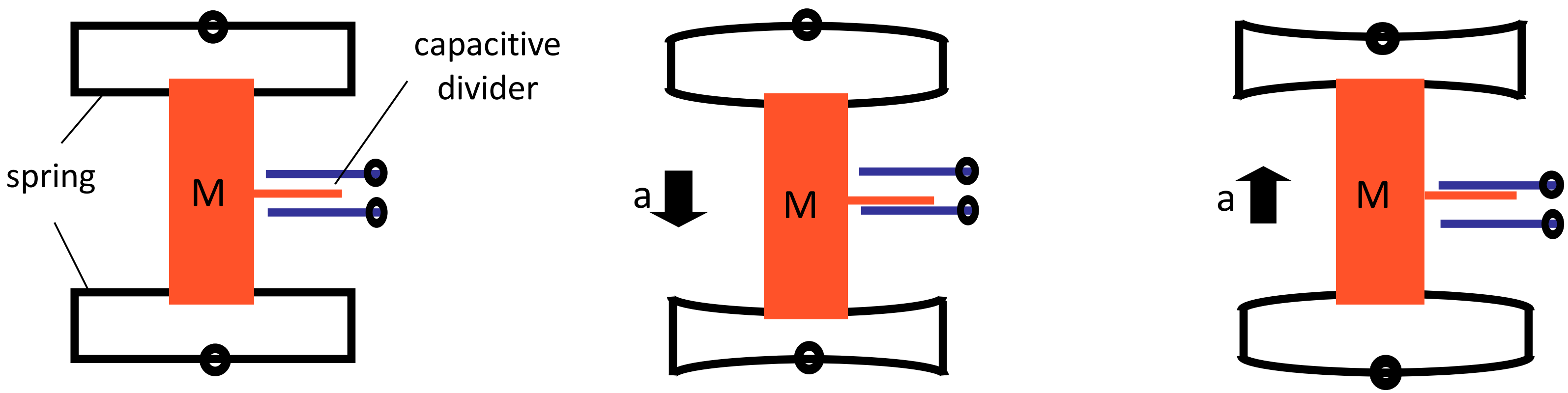
Mechanical Accelerometer

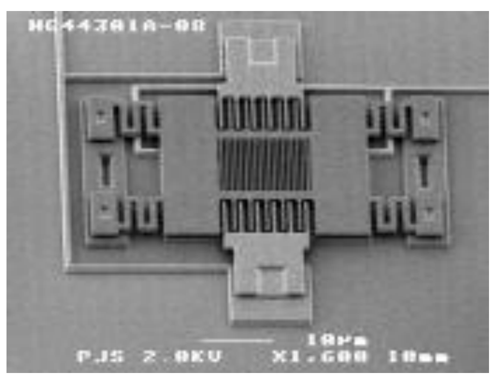
What is an IMU?

- Different categories
  - Mechanical ($100,000-1M)
  - Optical ($20,000-100k)
  - MEMS (from 1$ (phones) to 1,000$ (higher cost because they have a microchip running a Kalman filter))
- For small mobile robots & drones: MEMS IMU are mostly used
  - Cheap
  - Power efficient
  - Light weight and solid state

MEMS Accelerometer

A spring-like structure connects the device to a seismic mass vibrating in a capacitve divider. A capacitive divider converts the displacement of the seismic mass into an electric signal. Damping is created by the gas sealed in the device.





MEMS Gyroscopes

- MEMS gyroscopes measure the Coriolis forces acting on MEMS vibrating structures (tuning forks, vibrating wheels, or resonant solids)
- Their working principle is similar to the haltere of a fly
- Haltere are small structures of some two-winged insects, such as flies. They are flapped rapidly and function as gyroscopes, informing the insect about rotation of the body during flight.

Why IMU?

- ➢ Monocular vision is scale ambiguous.
- ➢ Pure vision is not robust enough
  - Low texture
  - High dynamic range
  - High speed motion

Robustness is a critical issue: Tesla accident

> *"The autopilot sensors on the Model S failed to distinguish a white tractor-trailer crossing the highway against a bright sky. " [The Guardian]*

Why vision?

- Pure IMU integration will lead to large drift (especially cheap IMUs)
  - Will see later mathematically
  - Intuition
    - Integration of angular velocity to get orientation: if there is a bias in angular velocity, the error is **proportional to t**
    - Double integration of acceleration to get position: if there is a bias in acceleration, the error of position is **proportional to t^2**
    - Worse, the actual position error also depends on the orientation error (see later).

	Accelerometer Bias Error		Horizontal Position Error [m]			
Grade	[mg]		1s	10s	60s	1hr
Navigation	0.025		0.13 mm	12 mm	0.44 m	1.6 km
Tactical	0.3		1.5 mm	150 mm	5.3 m	19 km
Industrial	3		15 mm	1.5 m	53 m	190 km
Automotive	125		620 mm	60 m	2.2 km	7900 km

Automotive, Smartphone, & Drone accelerometers

Errors computed assuming that the device at rest: http://www.vectornav.com/support/library/imu-and-ins

Why visual inertial fusion?

IMU and vision are complementary

Cameras

- ✓ Precise in slow motion
- ✓ Rich information for other purposes

- x Limited output rate (~100 Hz)
- x Scale ambiguity in monocular setup
- x Lack of robustness

IMU

- ✓ Robust
- ✓ High output rate (~1,000 Hz)
- ✓ Accurate at high acceleration

- x Large relative uncertainty when at low acceleration/angular velocity
- x Ambiguity in gravity / acceleration

What cameras and IMU have in common: both estimate the pose incrementally (known as dead-reckoning), which suffers from drifting over time. Solution: loop detection and loop closure

Outline

- Introduction
- IMU model and Camera-IMU system
- Different paradigms
  - Closed-form solution
  - Filtering approaches
  - Smoothing methods
    - Fixed-lag Smoothing (aka sliding window estimators)
    - Full smoothing methods
- Camera-IMU extrinsic calibration and Synchronization

IMU model: Measurement Model

➢ Measures angular velocity and acceleration in the body frame:

$$
{}_{\mathrm{B}}\tilde{\boldsymbol{\omega}}_{\mathrm{WB}}(t) = {}_{\mathrm{B}}\boldsymbol{\omega}_{\mathrm{WB}}(t) + \boxed{\mathbf{b}^{g}(t) + \mathbf{n}^{g}(t)}
$$

$$
{}_{\mathrm{B}}\tilde{\mathbf{a}}_{\mathrm{WB}}(t) = \mathbf{R}_{\mathrm{BW}}(t)\left({}_{\mathrm{W}}\mathbf{a}_{\mathrm{WB}}(t) - {}_{\mathrm{W}}\mathbf{g}\right) + \boxed{\mathbf{b}^{a}(t) + \mathbf{n}^{a}(t)}
$$

measurements

noise

where the superscript g stands for Gyroscope and a for Accelerometer

Notations:

- Left subscript: reference frame in which the quantity is expressed
- Right subscript {Q}{Frame1}{Frame2}: Q of Frame2 with respect to Frame1
- Noises are all in the body frame

IMU model: Noise Property

- Additive Gaussian white noise: $\mathbf{n}^g(t),\ \mathbf{n}^a(t)$

- Bias: $\mathbf{b}^g(t),\ \mathbf{b}^a(t)$

$$\dot{\mathbf{b}}(t) = \sigma_b \mathbf{w}(t) \qquad \mathbf{w}(\mathrm{t}) \sim \mathbf{N}(\mathrm{o}, 1)$$

i.e., the derivative of the bias is white Gaussian noise (so-called random walk)

Some facts about IMU biases:

- They can change due to temperature change, mechanical pressure, etc.
- They can change every time the IMU is started
- Good news: they can be estimated

Trawny, Nikolas, and Stergios I. Roumeliotis. "Indirect Kalman filter for 3D attitude estimation.“
https://github.com/ethz-asl/kalibr/wiki/IMU-Noise-Model

IMU model: Integration

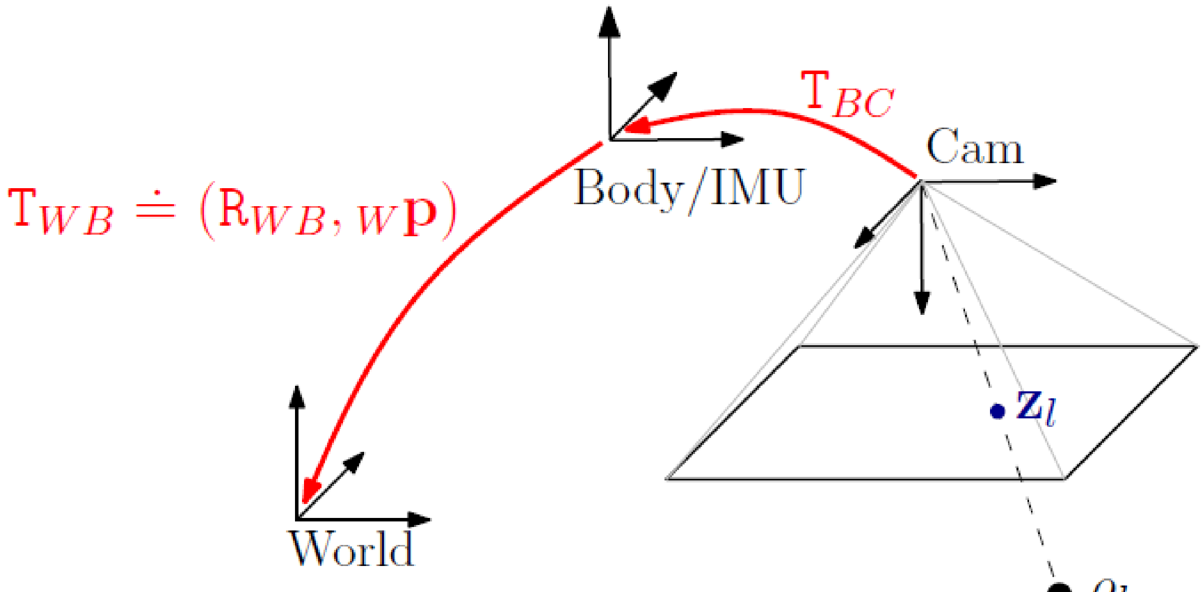
$$\mathbf{p}_{\mathrm{Wt}_2} = \boxed{\mathbf{p}_{\mathrm{Wt}_1}} + (t_2 - t_1)\boxed{\mathbf{v}_{\mathrm{Wt}_1}} + \int_{t_1}^{t_2}\!\!\int \left(\boxed{\mathbf{R}_{\mathrm{Wt}}}(t)\left(\tilde{\mathbf{a}}(t) - \mathbf{b}^a(t)\right) + {}_{\mathrm{w}}\mathbf{g} \right) \mathrm{d}t^2$$

per component: $\{t\}$ stands for {B}ody frame at time t

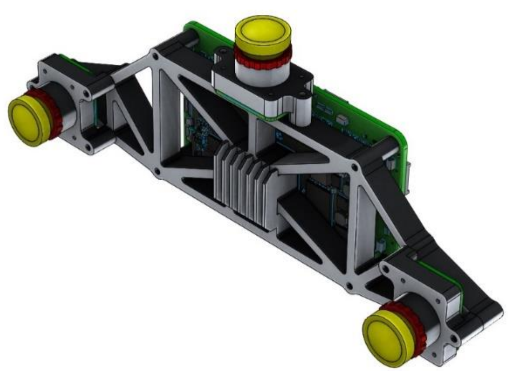
- Depends on initial position and velocity
- The rotation R(t) is computed from the gyroscope

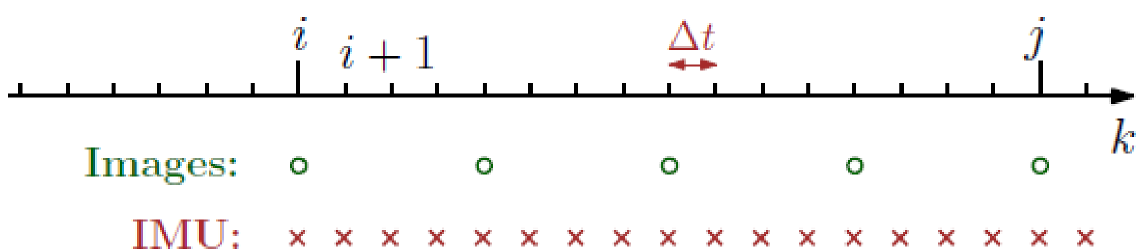
Trawny, Nikolas, and Stergios I. Roumeliotis. "Indirect Kalman filter for 3D attitude estimation."

Camera-IMU System



There can be multiple cameras.





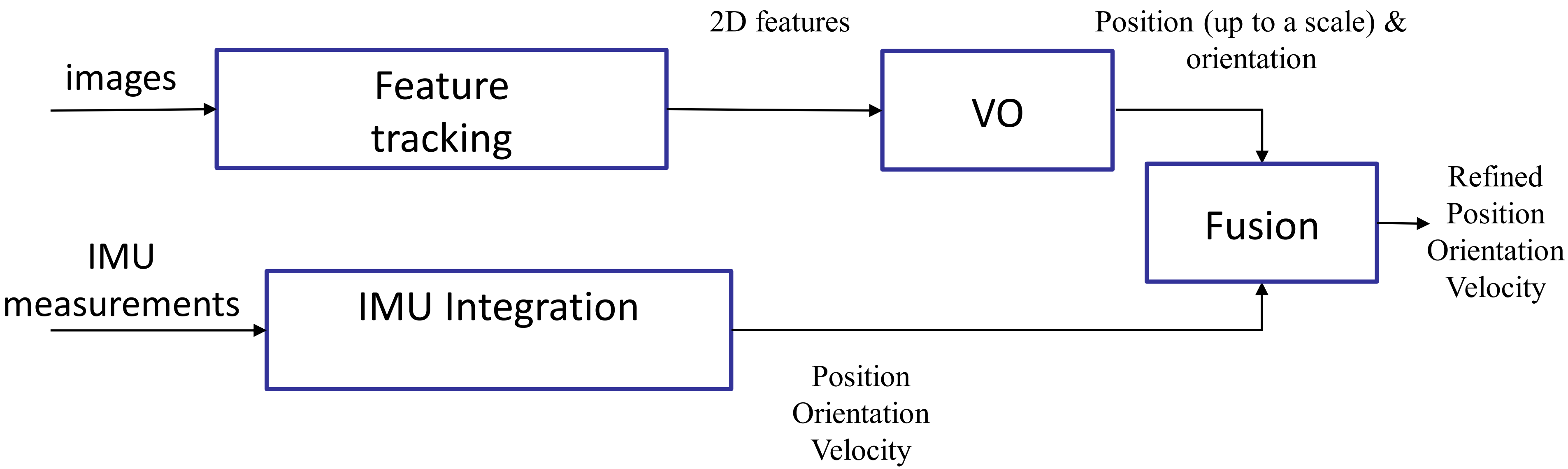
Outline

- Introduction
- IMU model and Camera-IMU system
- Different paradigms
  - Closed-form solution
  - Filtering approaches
  - Smoothing methods
    - Fixed-lag Smoothing (aka sliding window estimators)
    - Full smoothing methods
- Camera-IMU extrinsic calibration and Synchronization

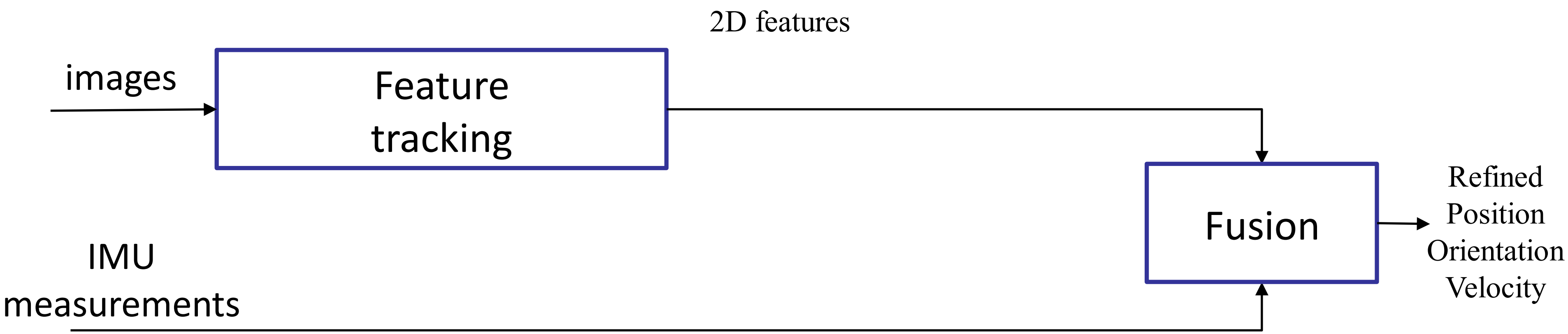
Different paradigms

- **Loosely coupled**:
  - Treats VO and IMU as two separate (not coupled) black boxes
    - Each black box estimates pose and velocity from visual (up to a scale) and inertial data (absolute scale)
- **Tightly coupled**:
  - Makes use of the raw sensors’ measurements:
    - 2D features
    - IMU readings
    - More accurate
    - More implementation effort
  - In the following slides, we will only see **tightly coupled approaches**

The Loosely Coupled Approach



The Tightly Coupled Approach



Filtering: Visual Inertial Formulation

System states:

Tightly coupled: $\mathbf{X} = \left[{}_{\mathrm{W}}\mathbf{p}(t);\ \mathbf{q}_{\mathrm{WB}}(t);\ {}_{\mathrm{W}}\mathbf{v}(t); \mathbf{b}^{a}(t); \mathbf{b}^{g}(t);\ \boxed{{}_{\mathrm{W}}\mathbf{L}_{1};\ {}_{\mathrm{W}}\mathbf{L}_{2}; \ldots ; {}_{\mathrm{W}}\mathbf{L}_{K}} \right]$

Loosely coupled: $\mathbf{X} = \left[{}_{\mathrm{W}}\mathbf{p}(t);\ \mathbf{q}_{\mathrm{WB}}(t);\ {}_{\mathrm{W}}\mathbf{v}(t); \mathbf{b}^{a}(t); \mathbf{b}^{g}(t) \right]$

Corke, An Introduction to Inertial and Visual Sensing, IJRR’07

Outline

- Introduction
- IMU model and Camera-IMU system
- Different paradigms
  - Closed-form solution
  - Filtering approaches
  - Smoothing methods
    - Fixed-lag Smoothing (aka sliding window estimators)
    - Full smoothing methods
- Camera-IMU extrinsic calibration and Synchronization

Closed-form Solution (1D case)

- The absolute pose x is known up to a scale s, thus

$$x = s\tilde{x}$$

- From the IMU

$$x = x_0 + v_0(t_1 - t_0) + \iint_{t_0}^{t_1} a(t)dt$$

- By equating them

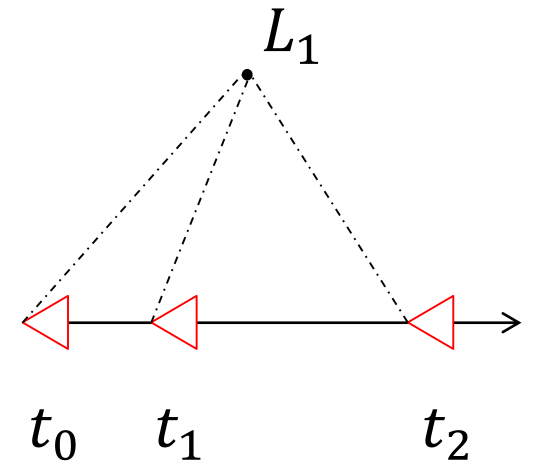
$$s\tilde{x} = x_0 + v_0(t_1 - t_0) + \iint_{t_0}^{t_1} a(t)dt$$

- As shown in [Martinelli'14], for 6DOF, both s and v_0 can be determined in closed form from a **single feature observation and 3 views**. x_0 can be set to 0.

Martinelli, Closed-form solution of visual-inertial structure from motion, International Journal of Computer Vision, 2014

Closed-form Solution (1D case)

$$\begin{cases} s\widetilde{x_1} = v_0(t_1 - t_0) + \iint_{t_0}^{t_1} a(t)dt \\ s\widetilde{x_2} = v_0(t_2 - t_0) + \iint_{t_0}^{t_2} a(t)dt \end{cases}$$



$$\begin{bmatrix} \widetilde{x_1} & (t_0 - t_1) \\ \widetilde{x_2} & (t_0 - t_2) \end{bmatrix} \begin{bmatrix} s \\ v_0 \end{bmatrix} = \begin{bmatrix} \iint_{t_0}^{t_1} a(t)dt \\ \iint_{t_0}^{2} a(t)dt \end{bmatrix}$$

Martinelli, Closed-form solution of visual-inertial structure from motion, International Journal of Computer Vision, 2014

Closed-form Solution (general case)

- Considers N feature observations and 6DOF case
- Can be used to initialize filters and smoothers (which always need an initialization point)
- More complex to derive than the 1D case. But it also reaches a linear system of equations that can be solved using the pseudoinverse:

$$A\boldsymbol{X} = \boldsymbol{S}$$

$\boldsymbol{X}$ is the vector of unknowns:

- 3D Point distances (wrt the first camera)
- Absolute scale,
- Initial velocity,
- Gravity vector,
- Biases

$\boldsymbol{A}$ and $\boldsymbol{S}$ contain 2D feature coordinates, acceleration, and angular velocity measurements

$$A = \left[\begin{array}{c|c|c|c|c|c|c|c|c|c|c|c}
T_2 & S_2 & \Gamma_2 & \boldsymbol{\mu}_1^1 & 0_3 & 0_3 & -\boldsymbol{\mu}_2^1 & 0_3 & 0_3 & 0_3 & 0_3 & 0_3 \\
0_{33} & 0_{33} & 0_{33} & \boldsymbol{\mu}_1^1 & -\boldsymbol{\mu}_1^2 & 0_3 & -\boldsymbol{\mu}_2^1 & \boldsymbol{\mu}_2^2 & 0_3 & 0_3 & 0_3 & 0_3 \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
0_{33} & 0_{33} & 0_{33} & \boldsymbol{\mu}_1^1 & 0_3 & -\boldsymbol{\mu}_1^N & -\boldsymbol{\mu}_2^1 & 0_3 & \boldsymbol{\mu}_2^N & 0_3 & 0_3 & 0_3 \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
T_{n_i} & S_{n_i} & \Gamma_{n_i} & \boldsymbol{\mu}_1^1 & 0_3 & 0_3 & 0_3 & 0_3 & 0_3 & -\boldsymbol{\mu}_{n_i}^1 & 0_3 & 0_3 \\
0_{33} & 0_{33} & 0_{33} & \boldsymbol{\mu}_1^1 & -\boldsymbol{\mu}_1^2 & 0_3 & 0_3 & 0_3 & 0_3 & -\boldsymbol{\mu}_{n_i}^1 & \boldsymbol{\mu}_{n_i}^2 & 0_3 \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
0_{33} & 0_{33} & 0_{33} & \boldsymbol{\mu}_1^1 & 0_3 & -\boldsymbol{\mu}_1^N & 0_3 & 0_3 & 0_3 & -\boldsymbol{\mu}_{n_i}^1 & 0_3 & \boldsymbol{\mu}_{n_i}^N
\end{array}\right]$$

- Martinelli, Vision and IMU data fusion: Closed-form solutions for attitude, speed, absolute scale, and bias determination, TRO'12
- Martinelli, Closed-form solution of visual-inertial structure from motion, Int. Journal of Comp. Vision, JCV'14
- Kaiser, Martinelli, Fontana, Scaramuzza, Simultaneous state initialization and gyroscope bias calibration in visual inertial aided navigation, IEEE RAL'17

Outline

- Introduction
- IMU model and Camera-IMU system
- Different paradigms
  - Closed-form solution
  - Filtering approaches
  - Smoothing methods
    - Fixed-lag Smoothing (aka sliding window estimators)
    - Full smoothing methods
- Camera-IMU extrinsic calibration and Synchronization

Different paradigms

Filtering	**Fixed-lag Smoothing**	**Full smoothing**
Only updates the most recent state • (e.g., extended Kalman filter)	Optimizes window of states • Marginalization • Nonlinear least squares optimization	Optimize all states • Nonlinear Least squares optimization
×1 Linearization ×Accumulation of linearization errors ×Gaussian approximation of marginalized states ✓Fastest	✓Re-Linearize ×Accumulation of linearization errors ×Gaussian approximation of marginalized states ✓Fast	✓Re-Linearize ✓Sparse Matrices ✓Highest Accuracy ×Slow (but fast with GTSAM)

Filtering: Visual Inertial Formulation

System states:

Tightly coupled: $\mathbf{X} = \left[{}_{\mathrm{W}}\mathbf{p}(t);\ \mathbf{q}_{\mathrm{WB}}(t);\ {}_{\mathrm{W}}\mathbf{v}(t); \mathbf{b}^{a}(t); \mathbf{b}^{g}(t);\ \boxed{{}_{\mathrm{W}}\mathbf{L}_1;\ {}_{\mathrm{W}}\mathbf{L}_2; \ldots; {}_{\mathrm{W}}\mathbf{L}_K} \right]$

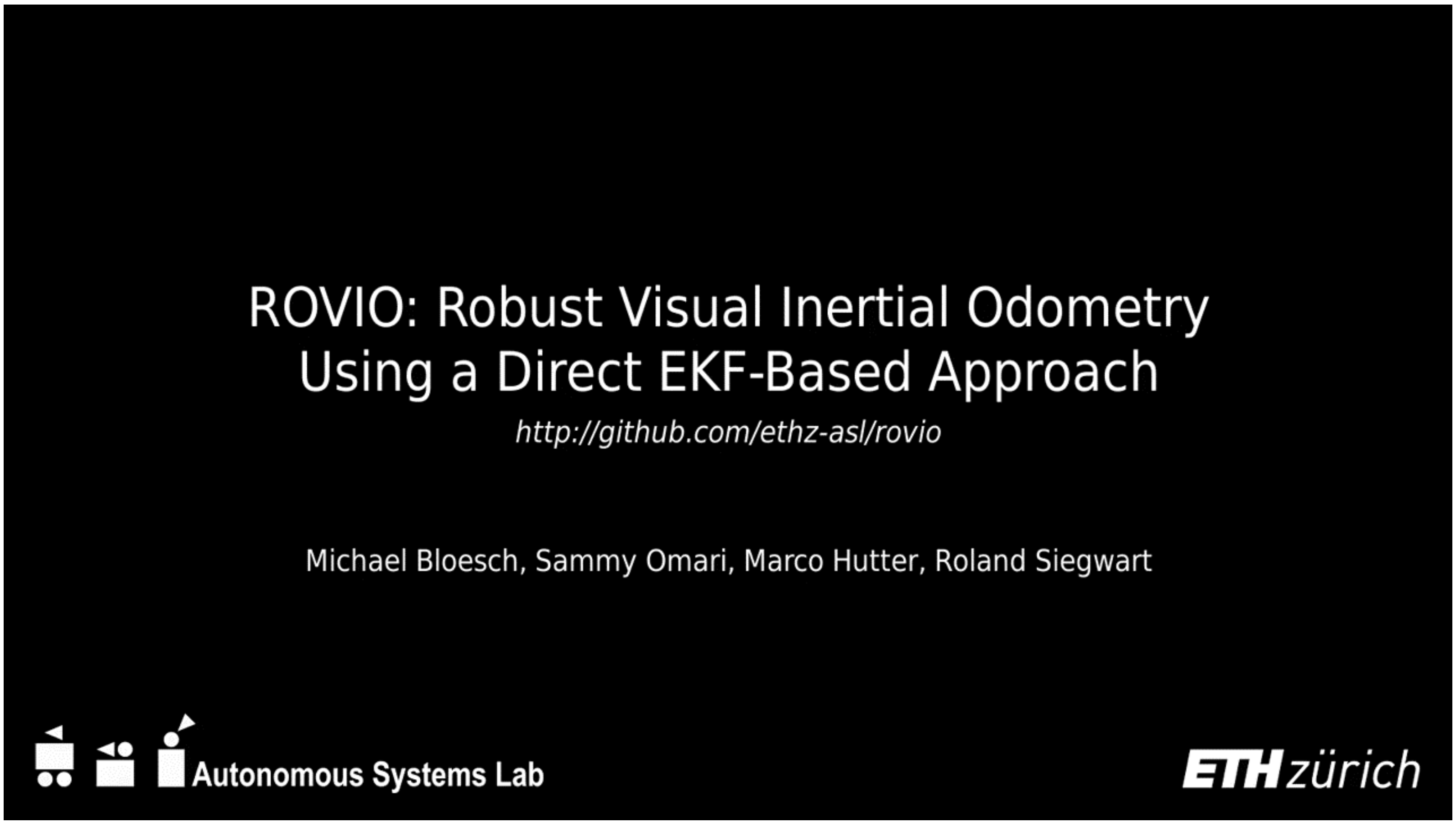
Loosely coupled: $\mathbf{X} = \left[{}_{\mathrm{W}}\mathbf{p}(t);\ \mathbf{q}_{\mathrm{WB}}(t);\ {}_{\mathrm{W}}\mathbf{v}(t); \mathbf{b}^{a}(t); \mathbf{b}^{g}(t) \right]$

Process Model: from IMU

- Integration of IMU states (rotation, position, velocity)
- Propagation of IMU noise
  - needed for calculating the Kalman Filter gain

Filtering: ROVIO

- EKF state: $\mathbf{X} = \left[\ {}_{\mathrm{W}}\mathbf{p}(t);\ \mathbf{q}_{\mathrm{WB}}(t);\ {}_{\mathrm{W}}\mathbf{v}(t);\mathbf{b}^{a}(t);\mathbf{b}^{g}(t);\ {}_{\mathrm{W}}\mathbf{L}_1;\ {}_{\mathrm{W}}\mathbf{L}_2;\ldots;{}_{\mathrm{W}}\mathbf{L}_K\ \right]$
- Minimizes the photometric error instead of the reprojection error



Bloesch, Michael, et al. "Iterated extended Kalman filter based visual-inertial odometry using direct photometric feedback“, IJRR’17

Filtering: Problems

- Wrong **linearization point**:
  - Linearization depends on the current estimates of states, which may be erroneous

- **Complexity** of the EKF grows **quadratically** in the number of estimated landmarks,

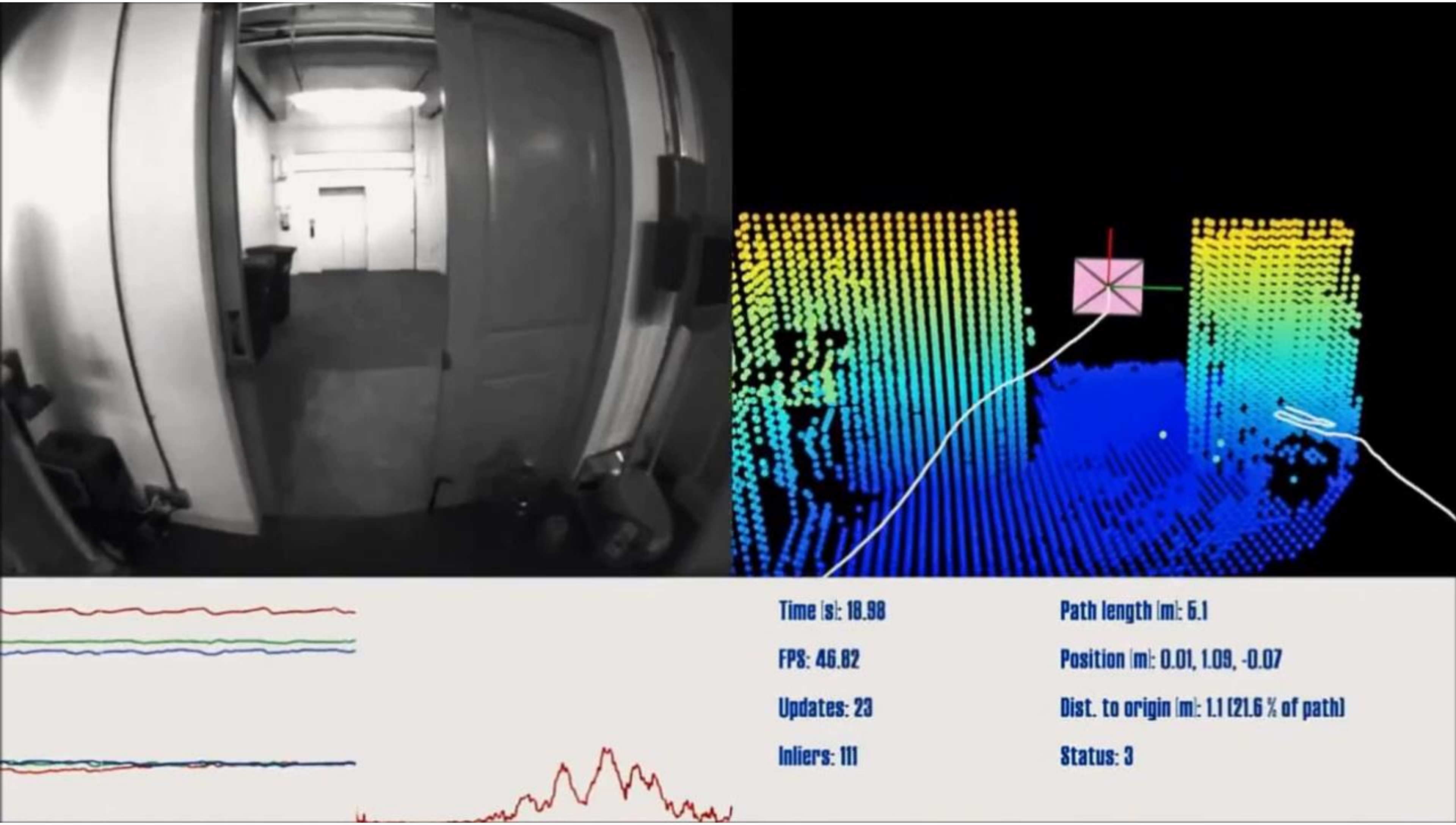
  → a **few landmarks** (~20) are typically tracked to allow real-time operation

- **Alternative: MSCKF** [Mourikis & Roumeliotis, ICRA’07]: used in Google ARCore
  - Keeps a window of recent states and updates them using EKF
  - incorporate visual observations without including point positions into the states

Mourikis & Roumeliotis, A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation, TRO’16

Li, Mingyang, and Anastasios I. Mourikis, High-precision, consistent EKF-based visual–inertial odometry, IJRR’13

Filtering: Google ARCore



Mourikis & Roumeliotis, A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation, TRO'16

Li, Mingyang, and Anastasios I. Mourikis, High-precision, consistent EKF-based visual–inertial odometry, IJRR'13

Outline

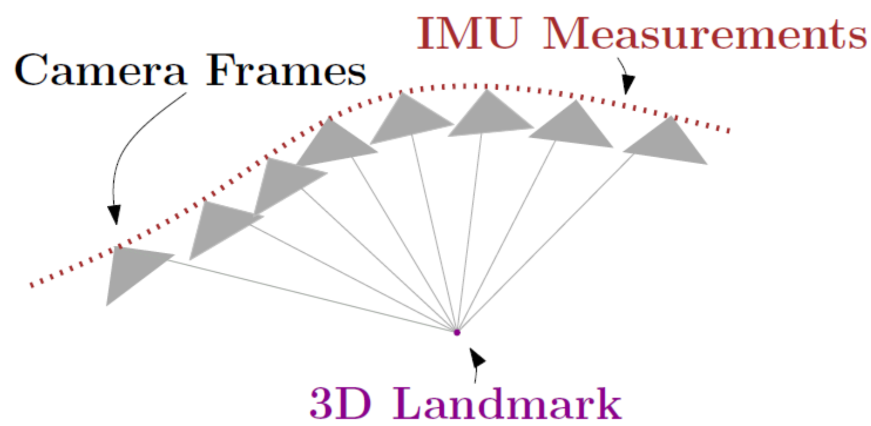
- Introduction
- IMU model and Camera-IMU system
- Different paradigms
  - Closed-form solution
  - Filtering approaches
  - Smoothing methods
    - Fixed-lag Smoothing (aka sliding window estimators)
    - Full smoothing methods
- Camera-IMU extrinsic calibration and Synchronization

Smoothing methods

VIO solved as a graph optimization problem over:

$X = \{x_1, \dots x_N\}$: Robot states (pose, velocity, acceleration)

$L = \{l_1, \dots, l_M\}$: 3D Landmarks



$x_k = f(x_{k-1}, u)$ is the state transition function; u is the set of IMU measurements

$z_{i_k} = \pi(x_k, l_i)$ is the reprojection of the landmark i in the camera frame k

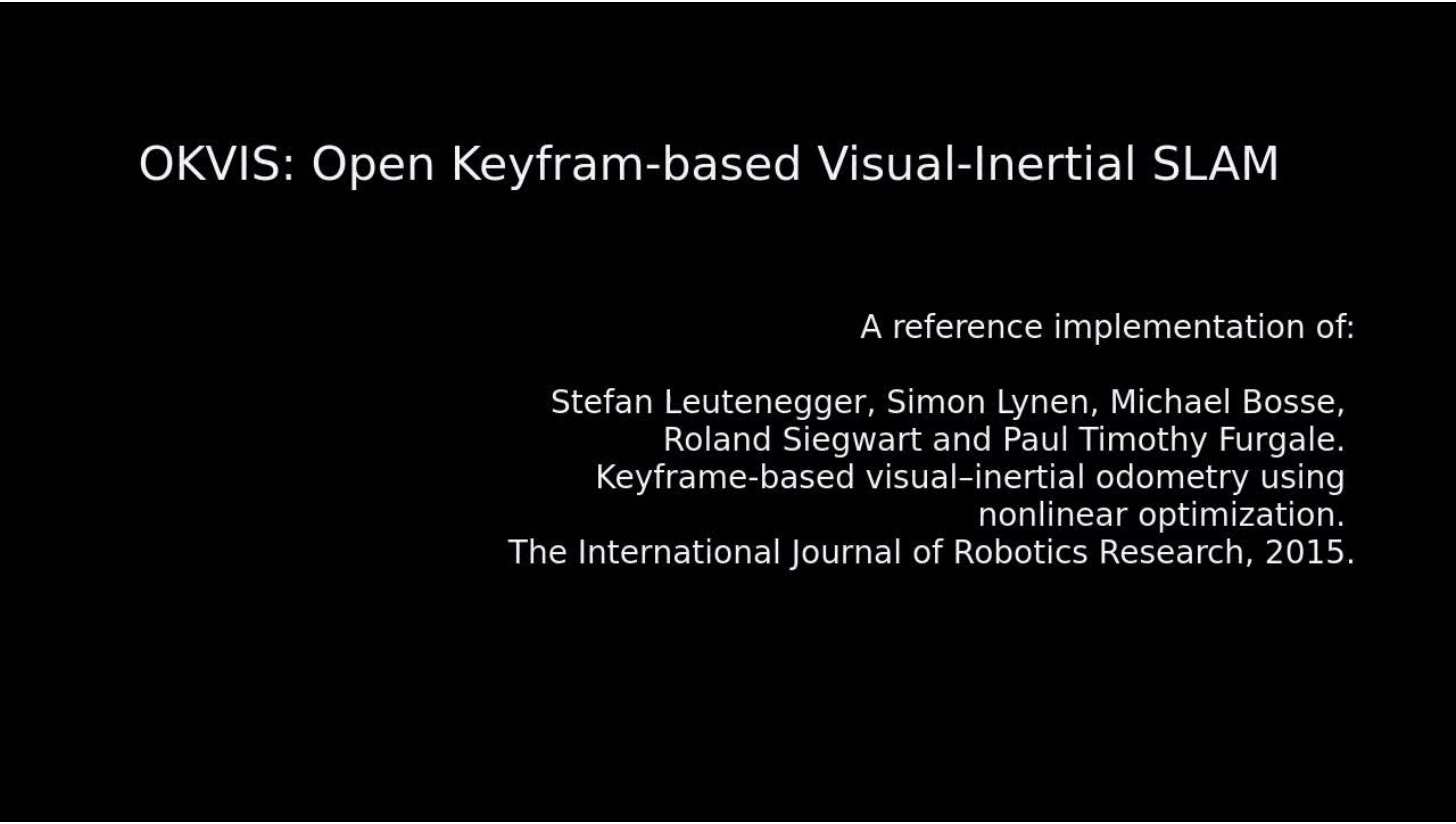
$$\{\mathrm{X, L}\} = argmin_{\{\mathrm{X, L}\}} \left\{ \underbrace{\sum_{k=1}^{N} \|f(x_{k-1}, u) - x_k\|^2_{\Lambda_k}}_{\textit{IMU residuals}} + \underbrace{\sum_{k=1}^{N} \sum_{i=1}^{M} \left\|\pi(x_k, l_i) - z_{i_k}\right\|^2_{\Sigma_{i_k}}}_{\textit{Reprojection residuals}} \right\}$$

Λ_k is the covariance from the IMU integration

Σ_{i_k} is the covariance from the noisy 2D feature measurements

[Jung, CVPR'01] [Sterlow'04] [Bryson, ICRA'09] [Indelman, RAS'13] [Patron-Perez, IJCV'15][Leutenegger, RSS'13-IJRR'15] [Forster, RSS'15, TRO'17]

Fixed-lag smoothing: OKVIS

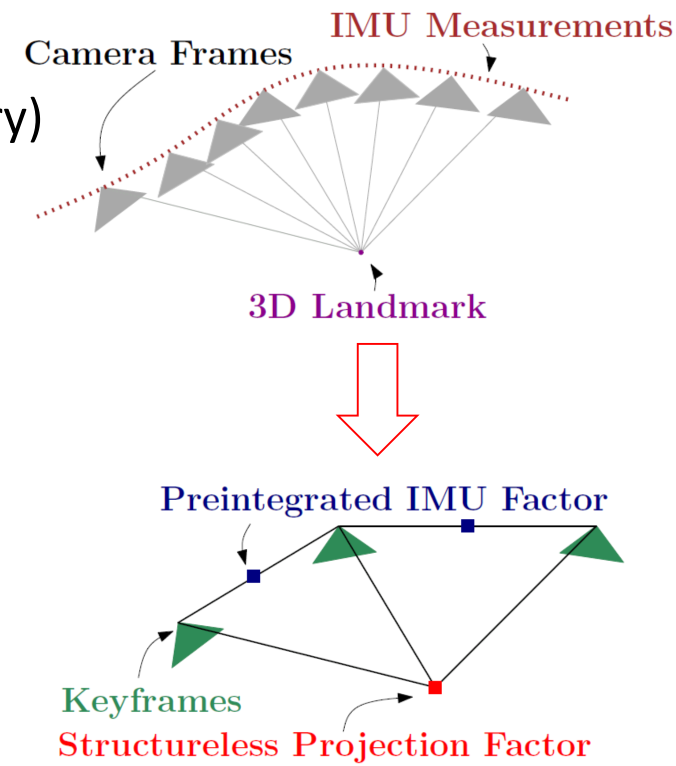


Leutenegger, OKVIS: Open Keyframe-based Visual-Inertial SLAM, IJRR'15

Full Smoothing: SVO+GTSAM & IMU Pre-integration

Solves the same optimization problem but:

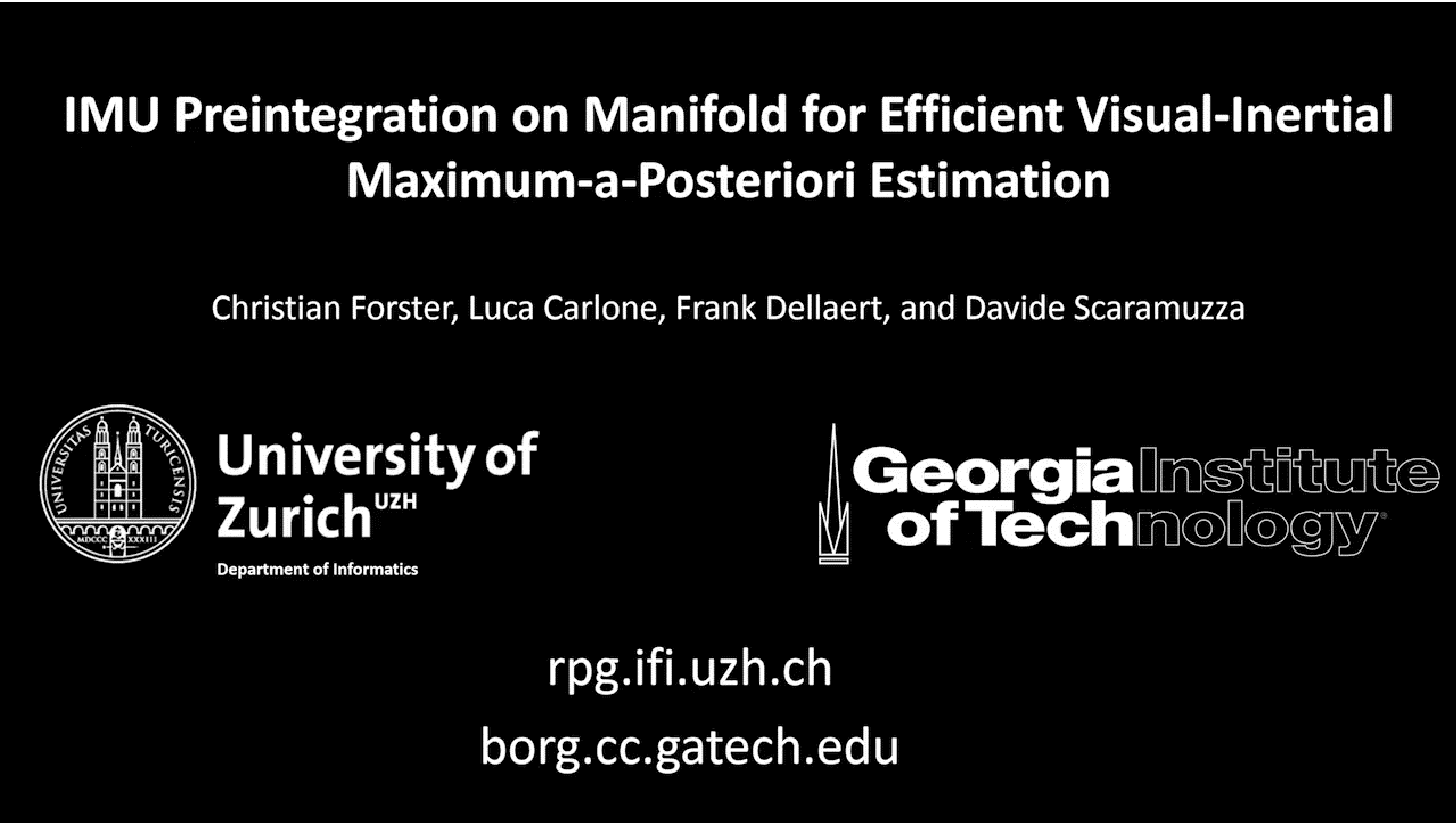
- ➢ Keeps all the frames (from the start of the trajectory)
- ➢ To make the optimization efficient
  - it makes the graph sparser using keyframes
  - pre-integrates the IMU data between keyframes
- ➢ Optimization salved using factor graphs (**GTSAM**)
  - Very fast because it only optimizes the poses which are affected by a new observation



$$\{\mathrm{X},\,\mathrm{L}\} = argmin_{\{\mathrm{X},\,\mathrm{L}\}} \left\{ \underbrace{\sum_{k=1}^{N} \| f(x_{k-1}, u) - x_k \|^2_{\Lambda_k}}_{\textit{IMU residuals}} + \underbrace{\sum_{k=1}^{N} \sum_{i=1}^{M} \left\| \pi(x_k, l_i) - z_{i_k} \right\|^2_{\Sigma_{i_k}}}_{\textit{Reprojection residuals}} \right\}$$

Forster, Carlone, Dellaert, Scaramuzza, On-Manifold Preintegration for Real-Time Visual-Inertial Odometry, IEEE Transactions on Robotics (TRO), Feb. 2017, **Best Paper Award 2018**.

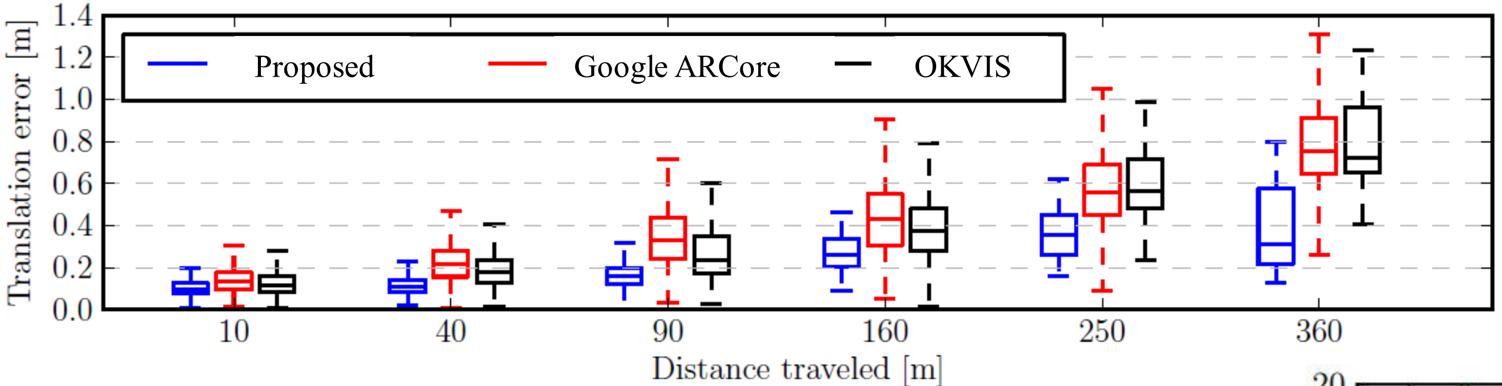
Full Smoothing: SVO+GTSAM & IMU Pre-integration

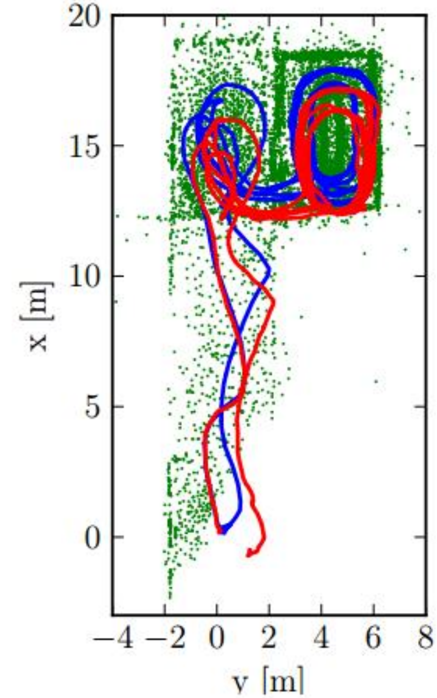


Forster, Carlone, Dellaert, Scaramuzza, On-Manifold Preintegration for Real-Time Visual-Inertial Odometry, IEEE Transactions on Robotics, Feb. 2017.

SVO + IMU Preintegration

Accuracy: 0.1% of the travel distance





Forster, Carlone, Dellaert, Scaramuzza, On-Manifold Preintegration for Real-Time Visual-Inertial Odometry, IEEE Transactions on Robotics, Feb. 2017.

Recap

- **Closed form solution**:
  - for 6DOF motion both s and v_0 can be determined **1 feature observation and at least 3 views** [Martinelli, TRO’12, IJCV’14, RAL’16]
  - Can be used to **initialize filters and smoothers**
- **Filters:** *update only last state → fast if number of features is low (~20)*
  - [Mourikis, ICRA’07, CVPR’08], [Jones, IJRR’11] [Kottas, ISER’12][Bloesch, IROS’15] [Wu et al., RSS’15], [Hesch, IJRR’14], [Weiss, JFR’13]
  - **Open source**: **ROVIO** [Bloesch, IROS’15, IJRR’17], **MSCKF** [Mourikis, ICRA’07] (i.e., Google ARCore)
- **Fixed-lag smoothers:** *update a window of states → slower but more accurate*
  - [Mourikis, CVPR’08] [Sibley, IJRR’10], [Dong, ICRA’11], [Leutenegger, RSS’13-IJRR’15]
  - **Open source: OKVIS** [Leutenegger, RSS’13-IJRR’15]
- **Full-smoothing methods:** *update entire history of states → slower but more accurate*
  - [Jung, CVPR’01] [Sterlow’04] [Bryson, ICRA’09] [Indelman, RAS’13] [Patron-Perez, IJCV’15] [Forster, RSS’15, TRO’16]
  - **Open source: SVO+IMU** [Forster, TRO’17]

Open Problem: consistency

- Filters
  - Linearization around different values of the same variable may lead to error
- Smoothing methods
  - May get stuck in local minima

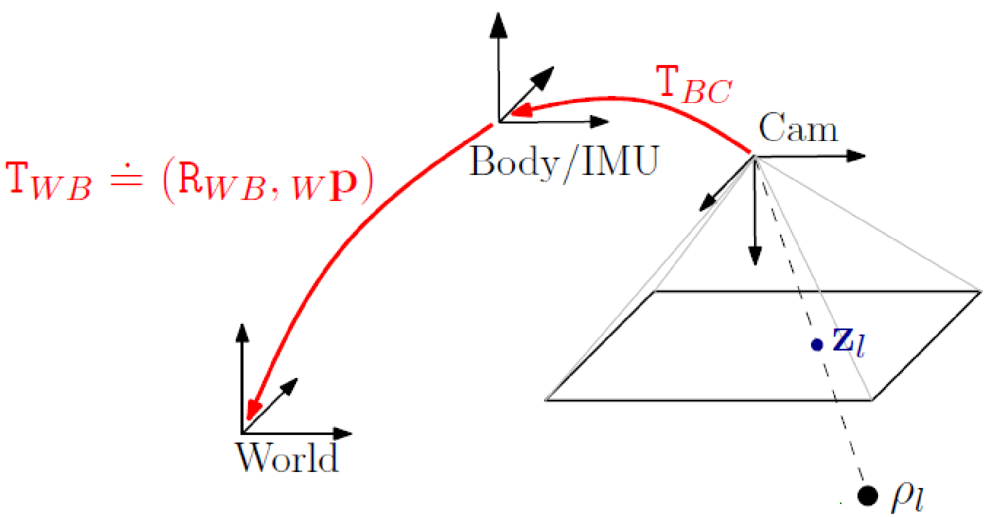
Outline

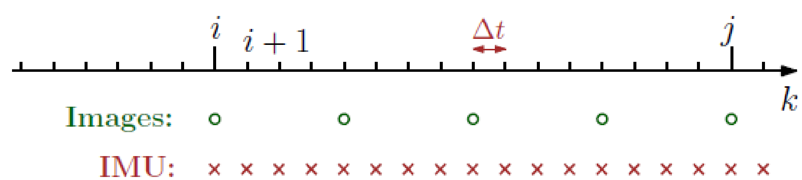
- ➢ Introduction
- ➢ IMU model and Camera-IMU system
- ➢ Different paradigms
  - Closed-form solution
  - Filtering approaches
  - Smoothing methods
    - Fixed-lag Smoothing (aka sliding window estimators)
    - Full smoothing methods
- ➢ Camera-IMU extrinsic calibration and Synchronization

Camera-IMU calibration

- **Goal**: estimate the rigid-body transformation $\boldsymbol{T_{BC}}$ and delay t_d between a camera and an IMU rigidly attached. Assume that the camera has already been intrinsically calibrated.

- **Data**:
  - Image points of detected calibration pattern (checkerboard).
  - IMU measurements: accelerometer $\{a_k\}$ and gyroscope $\{\omega_k\}$.

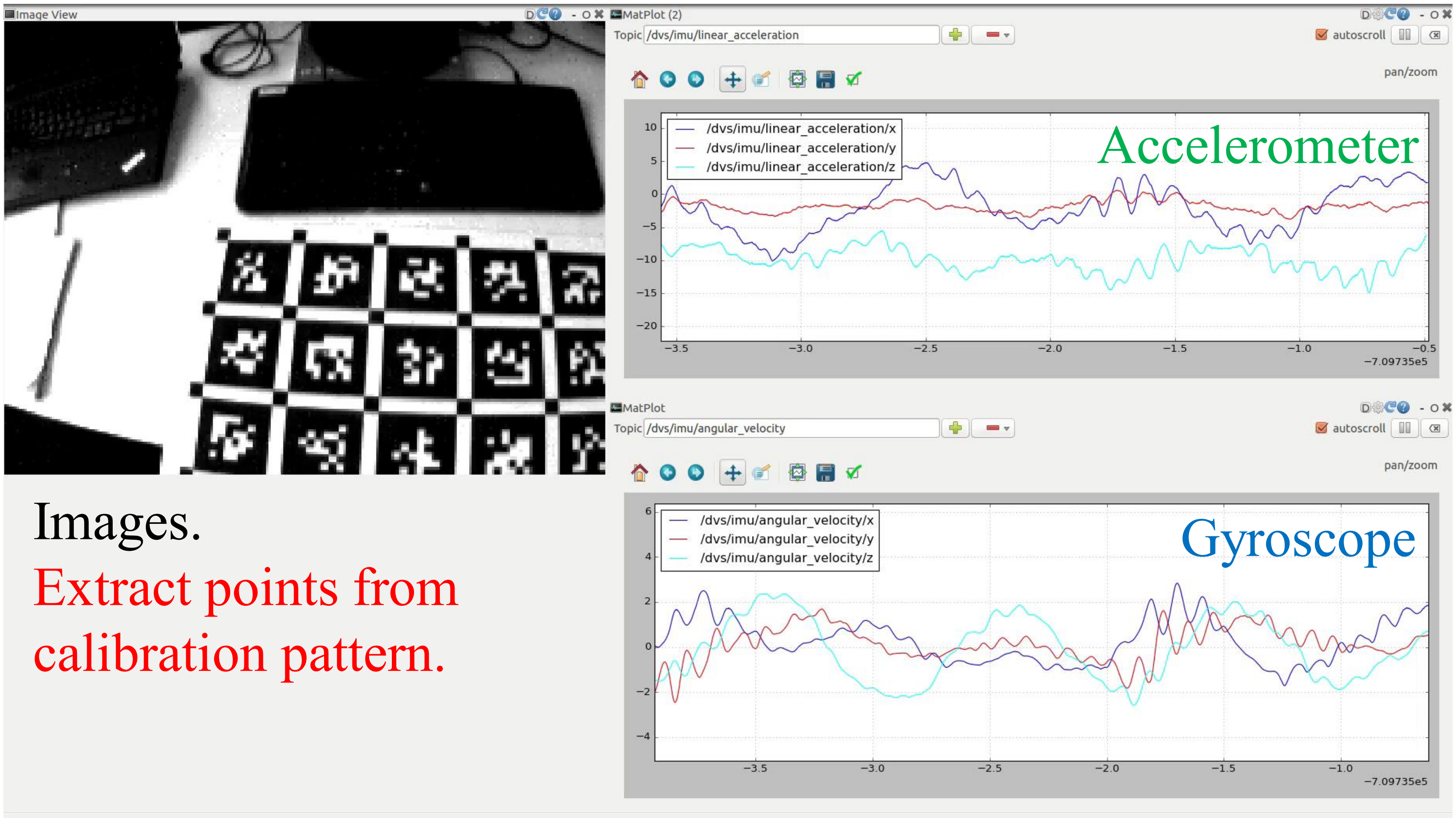




Furgale et al. "Unified Temporal and Spatial Calibration for Multi-Sensor Systems“, IROS’13.

Camera-IMU calibration - Example

- Data acquisition: Move the sensor in front of a static calibration pattern, exciting all degrees of freedom.



Camera-IMU calibration

- <u>Approach</u>: Minimize a cost function (Furgale'13):

  - $J(\theta) := J_{feat} + J_{acc} + J_{gyro} + J_{bias_{acc}} + J_{bias_{gyro}}$

    (Feature reprojection Error) $\sum_k \left(a_{IMU}(t_k - t_d) - a_{Cam}(t_k)\right)^2$ $\sum_k \left(\omega_{IMU}(t_k - t_d) - \omega_{Cam}(t_k)\right)^2$ $\int \left\| \frac{db_{acc}}{dt}(u) \right\|^2 du$ $\int \left\| \frac{db_{gyro}}{dt}(u) \right\|^2 du$

  - Unknowns: $T_{BC}, t_d, g_w, T_{WB}(t), b_{acc}(t), b_{gyro}(t)$
    - g_w = Gravity,
    - $T_{WB}(t)$ = 6-DOF trajectory of the IMU,
    - $b_{acc}(t)$, $b_{gyro}(t)$ = 3-DOF biases of the IMU

  - Continuous-time modelling using splines for $T_{WB}(t), b_{acc}(t), \ldots$

  - Numerical solver: Levenberg-Marquardt (i.e., Gauss-Newton).

Furgale et al. "Unified Temporal and Spatial Calibration for Multi-Sensor Systems", IROS'13.

Camera-IMU calibration - Example

- Software solution: Kalibr (Furgale'13).
  - Generates a report after optimizing the cost function.

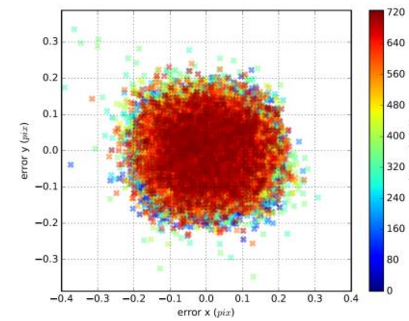
Residuals:
Reprojection error [px]: 0.0976 ± 0.051
Gyroscope error [rad/s]: 0.0167 ± 0.009
Accelerometer error [m/s^2]: 0.0595 ± 0.031

Transformation T_ci: (imu to cam):
[[0.99995526 -0.00934911 -0.00143776 0.00008436]
[0.00936458 0.99989388 0.01115983 0.00197427]
[0.00133327 -0.0111728 0.99993669 -0.05054946]
[0. 0. 0. 1.]]

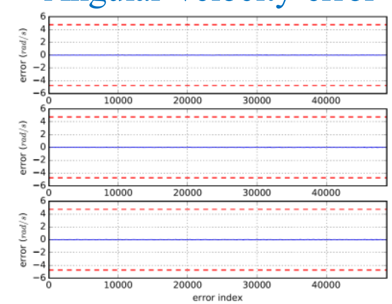
Time shift (delay *d*)
cam0 to imu0: [s] (t_imu = t_cam + shift)
0.00270636270255

Gravity vector in target coords: [m/s^2]
[0.04170719 -0.01000423 -9.80645621]

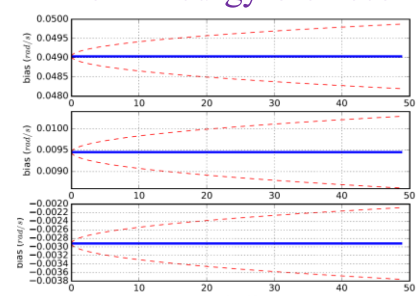
Reprojection error



Angular velocity error



Estimated gyro biases



Furgale et al. "Unified Temporal and Spatial Calibration for Multi-Sensor Systems", IROS'13.
https://github.com/ethz-asl/kalibr/wiki/camera-imu-calibration

Understanding Check

Are you able to answer the following questions?

- Why is it recommended to use an IMU for Visual Odometry?
- Why not just an IMU?
- How does a MEMS IMU work?
- What is the drift of an industrial IMU?
- What is the IMU measurement model?
- What causes the bias in an IMU?
- How do we model the bias?
- How do we integrate the acceleration to get the position formula?
- What is the definition of loosely coupled and tightly coupled visual inertial fusions?
- How can we use non-linear optimization-based approaches to solve for visual inertial fusion?