Dynamic Networks

Davide A. Cucci Jan Skaloud

Geodetic Engineering Laboratory
Ecole Polytechnique Fédérale de Lausanne
Switzerland

davide.cucci@gmail.com, jan.skaloud@epfl.ch

June 4th, 2022

Outline

1. 09.05 Intro - “la raison d’être”, theory & challenges (Ismael)
2. 09.30 DN in navigation (Davide)
3. 09.45 Example 1-Nav (Jan)
4. 10.30 !!! COFFEE BREAK !!!
5. 11.00 DN with optical constrains (Davide)
6. 11.20 Example 2-Nav/lidar (Jan)
7. 11.35 DN in photogrammetry (Davide)
8. 11.50 Example 3-Nav/photo
9. 12.15 Example 4-UAV (Jan)

Dynamic Networks?

An extension to **Geodetic Networks**

Differences:

1. One body which *moves*.
2. Points are the position of that body as a function of *time*.
3. Sensor measurements have to do with the *dynamic* properties of the body (e.g., acceleration, velocity, ...).

Figure: The Swiss gravity network.

Literature

The literature is huge, I give the *oldest* papers that I know of.

1. **Time dependent networks**

   Colomina, I., and M. Blázquez. "A unified approach to static and dynamic modelling in photogrammetry and remote sensing." *Proceedings of the XXth ISPRS Congress, Istanbul, Turkey*. 2004.

2. In **terrestrial mobile mapping**

   Rouzaud, D., and Skaloud, J. "Rigorous integration of inertial navigation with optical sensors by dynamic networks." *Navigation* 58.2 (2011): 141-152.

In robotics

The same concept is referred as *factor-graphs*, nowadays standard in Simultaneous Localization and Mapping (SLAM).

N.B. A DN is a factor-graph, a factor-graph is *sometimes* a DN

1. **Least-squares network optimization**

   Kümmerle, R., et al. "g^{2}o: A general framework for graph optimization." *2011 IEEE International Conference on Robotics and Automation. IEEE*, 2011.

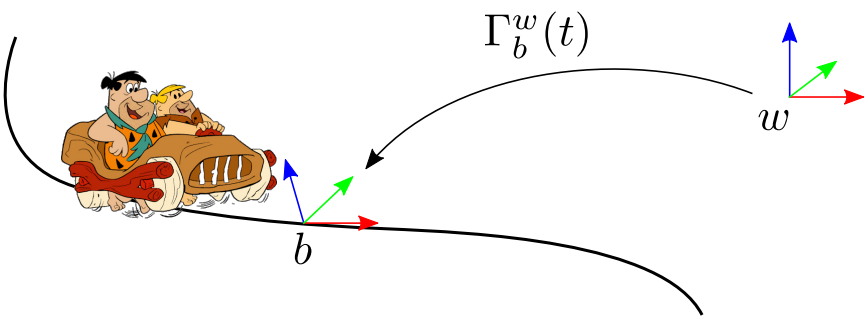
2. **Georgia Tech Smoothing and Mapping (GTSAM)**

   Kaess, M., et al. "iSAM2: Incremental smoothing and mapping using the Bayes tree." *The International Journal of Robotics Research* 31.2 (2012): 216-235.

3. **Ceres Solver (Google)**

To understand DNs we need to start from **navigation**.

Navigation in a nutshell



1. Determine Fred's and Barney's **pose** $\Gamma_b^w(t) = [r^w(t), R_b^w(t)]$ from sensor measurements
2. (side-effect) determine other **parameters**, e.g., inertial sensor biases $bf(t)$ and $b\omega(t)$

Ok, how do we do?

1. We mount a set of sensors on b: $\{s_1, s_2, \ldots s_n\}$,
2. These sensors provide measurements, e.g., $z_{s_1,t}$,
3. These measurements are **noisy**,
4. We come up with a deterministic and stochastic model for z in the form

$$z_{s_j,t} = f_j\Big(r^w(t), R_b^w(t), \underbrace{\dot{r}^w(t), \dot{R}_b^w(t)}_{\textit{derivatives!}}, \ \ldots\ , \underbrace{\xi_j(t)}_{\textit{noise}}\Big),$$

5. This gives $P(z_{s_j,t}|r^w(t), R_b^w(t), \ldots)$,
6. We use this to determine $r^w(t), R_b^w(t)$, see later on.

The "state" (... steals money from people with taxes)

The **state**, $\mathbb{X}$, is the set of quantities that we need to *predict* the available sensor measurements (**Complete state assumption, or Markov assumption**).

What does it mean to "*predict*"?

$$\hat{z}_{s_j,t} = \mathbb{E}\left[z_{s_j,t}|\mathbb{X}\right]$$

N.B. If $\xi_j(t) \sim \mathcal{N}(0, \Sigma_j)$ and additive, i.e., $f_j\Big(\mathbb{X}, \xi_j(t)\Big) = f_j(\mathbb{X}) + \xi_j$

$$\hat{z}_{s_j,t} = f_j(\mathbb{X})$$

Estimation

1. $\mathbb{X}$ is what we can *hope* to estimate from the measurements that we have.
2. How can we formulate an estimator for $\mathbb{X}$?

Unordered list of problems that you encounter in navigation:

1. defining f_j is already tricky in some cases,
2. f_j is typically non-linear in $\mathbb{X}$,
3. ξ_j is not always Gaussian
4. $\mathbb{X}$ is a non-Euclidean manifold, typically includes elements of SO(3) or SE(3)

Quite useful: Hertzberg, C., et al. "Integrating generic sensor fusion algorithms with sound state representations through encapsulation of manifolds." *Information Fusion* 14.1 (2013): 57-77.

Recursive Bayesian Filters

EPFL

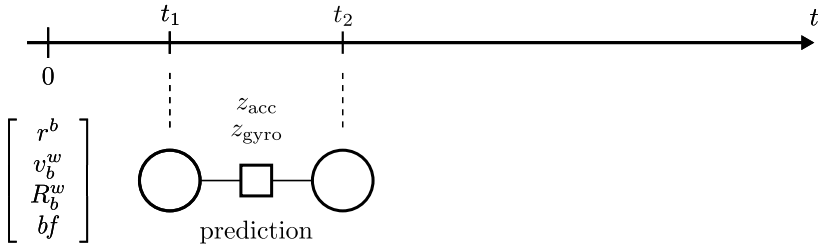
Measurements come one after the other, right?

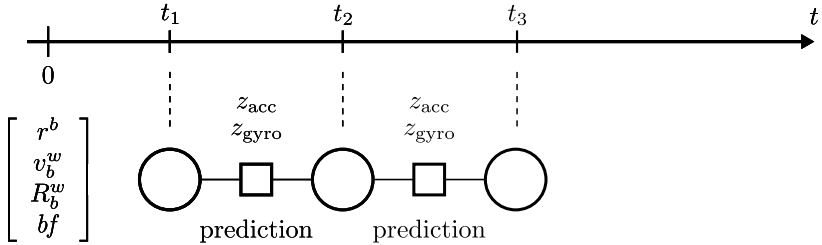
1. We remember the Markov assumption,
2. We have the state at time $t-1$, $\mathbb{X}_{t-1}$,
3. We have a notion of the distribution of $\mathbb{X}_{t-1}$, $P(\mathbb{X}_{t-1})$, a **belief** in robotics,
4. We have a control u_t (or other known forcing input, e.g. from IMU) [optional],
5. We have a way to tell $P(\mathbb{X}_t|u_t, \mathbb{X}_{t-1})$, (**prediction step**)
6. We can do Bayes: $P(\mathbb{X}_t) = \lambda P(z_t|\mathbb{X}_t) \int P(\mathbb{X}_t|u_t, \mathbb{X}_{t-1}) P(\mathbb{X}_{t-1}) d\mathbb{X}$ (**update step**)

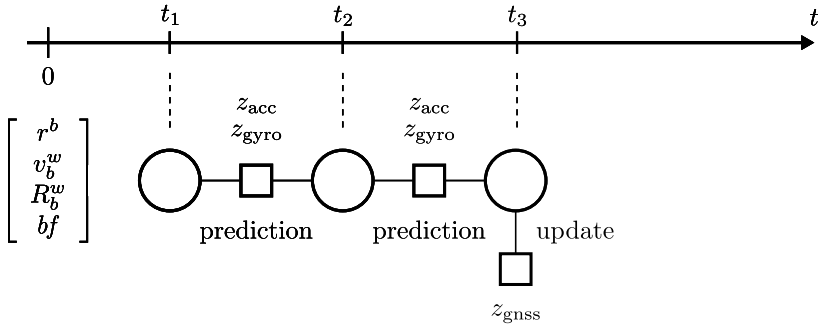
This leads to the **Kalman filter**, which is very good.

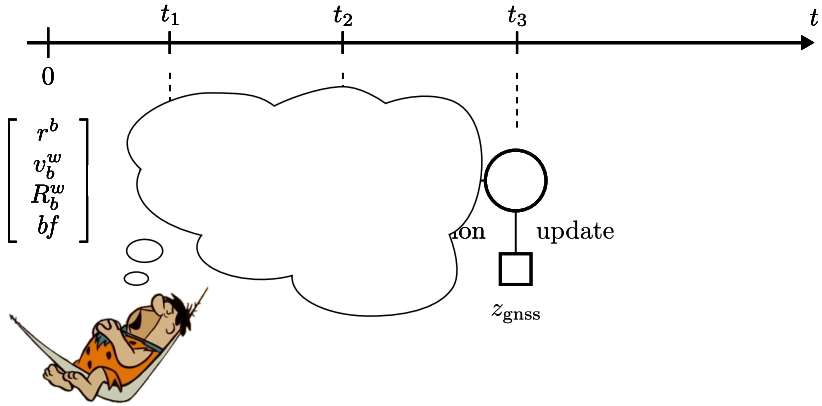
It is better to look at Bar-Shalom, Yaakov, at al. *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons, 2004.

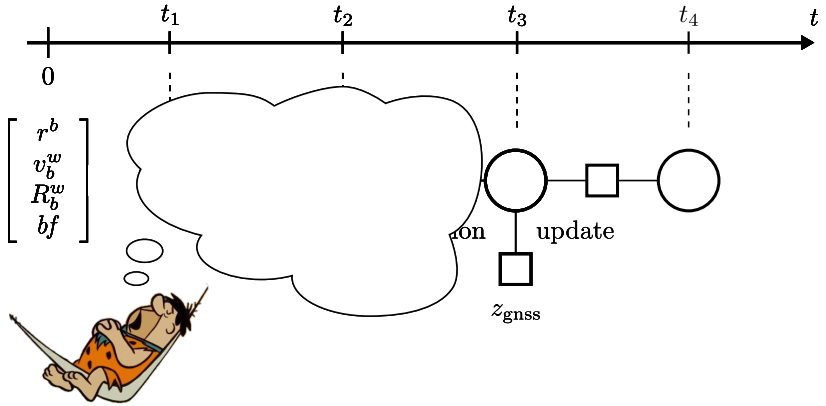

t_1
t
0
r^b
v_b^w
R_b^w
bf


t_1
t_2
t
0
z_{acc}
z_{gyro}
r^b
v_b^w
R_b^w
bf
prediction

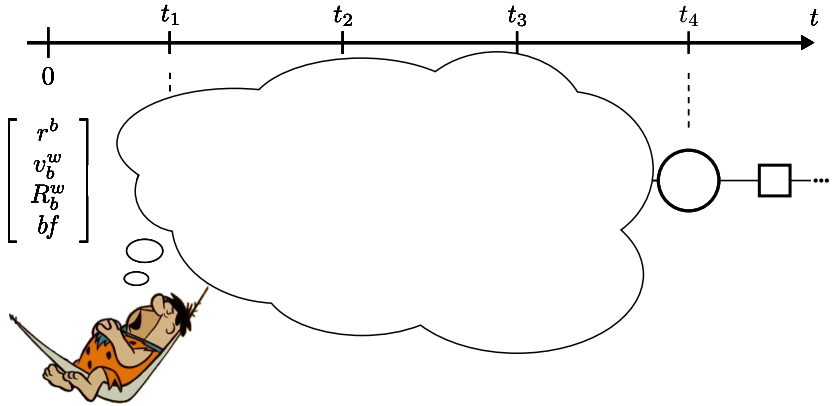

t_1
t_2
t_3
t
0
z_{acc}
z_{gyro}
z_{acc}
z_{gyro}
r^b
v_b^w
R_b^w
bf
prediction
prediction


t_1
t_2
t_3
t
0
z_{acc}
z_{gyro}
z_{acc}
z_{gyro}
r^b
v_b^w
R_b^w
bf
prediction
prediction
update
z_{gnss}


t_1
t_2
t_3
t
0
r^b
v_b^w
R_b^w
bf
on
update
z_{gnss}


t_1
t_2
t_3
t_4
t
0
r^b
v_b^w
R_b^w
bf
on
update
z_{gnss}

Inertial navigation and EKFs



Inertial navigation and EKFs

PROs:

1. Well established
2. Very low time-and-memory complexity

CONs:

1. New evidence can't correct past states (Markov!! )
2. High sensitivity to initial approximations (e.g., attitude initialization with low-cost IMUs)
3. Difficult rigorous formulation of spatial constraints
   e.g., *"I was already here before"*, as in airborne gravimetry (Skaloud et al, 2016)

We don't like *beliefs*.

Figure: The Nihilists (The Big Lebowski, 1998)

1. We consider all times and all measurements together
2. We know $P(z_j|\mathbb{X})$, then we define

$$\hat{\mathbb{X}} = \underset{\mathbb{X}}{\text{argmax}} \prod_j \mathcal{L}(\mathbb{X}; z_j)$$

3. If ξ_j is additive and $\xi_j \sim \mathcal{N}(0, \Sigma_j)$:

$$\hat{\mathbb{X}} = \underset{\mathbb{X}}{\text{argmin}} \sum_j \Big(z_j - f_j(\mathbb{X})\Big) \Sigma_j^{-1} \Big(z_j - f_j(\mathbb{X})\Big)^T \quad \text{(NL least-squares)}$$

4. We can do robust estimation if you so wish.

Inertial navigation in DNs

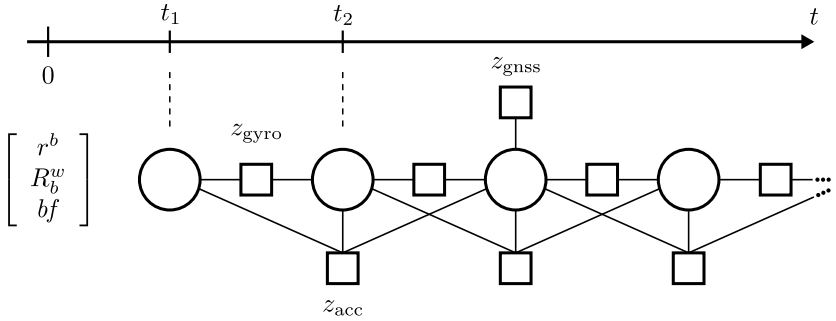


Figure: A simple DN for inertial navigation

Where did the velocities go?

EPFL

1. v_b^w and ω_{wb}^b are not estimated explicitly
2. ... but approximated via finite differences of poses

Example: Gyroscope observation model

$$z_{\omega,t} = \overbrace{\ldots}^{\text{Earth rotation. Yes, it turns..}} + \underbrace{\omega_{wb}^b(t)}_{\frac{\log\left(R_b^w(t_{i-1})^T R_b^w(t_i)\right)}{t_i - t_{i-1}}} + b\omega(t) + \xi_\omega(t).$$

For details see:

Cucci, D. A., Rehak, M., and Skaloud, J. "Bundle adjustment with raw inertial observations in UAV applications." *ISPRS Journal of Photogrammetry and Remote Sensing* 130 (2017): 1-12. [→ link]

Stochastic modeling of state components

1. Beliefs are hard to die, sometimes we need them.
2. Prior knowledge becomes *zero*-measurements, as if there was a sensor always measuring 0.
3. Example: auto-regressive 1st-order (AR1) process for the gyroscope bias

$$z_{\mathrm{AR1},t} = 0 = \underbrace{b\omega(t) - \phi b\omega(t-1)}_{f(\mathbb{X})} + \xi$$

4. Many other types of prior knowledge can be formulated like this. ML becomes MAP.

For details see:

Cucci, D. A., et al. "A general approach to time-varying parameters in pose-graph optimization." *2017 European Navigation Conference (ENC). IEEE*, 2017 [→ link]

Inertial navigation in DNs

PROs:

1. Joint & optimal estimation, no Forward/Backward/Smoothing needed
2. Lower sensitivity to initialization approximations (e.g., without large-heading models)
3. Direct integration of spatial constraints (e.g., optical observations)
4. Better de-correlation between parameters
5. As in geodetic networks, we can evaluate of redundancy, controllability, sensitivity, ...

CONs:

1. Severe time-and-memory complexity (but we can handle it!)
2. Ideal for post-processing, tricks needed for real-time.

Example 1 - INS/GNSS navigation

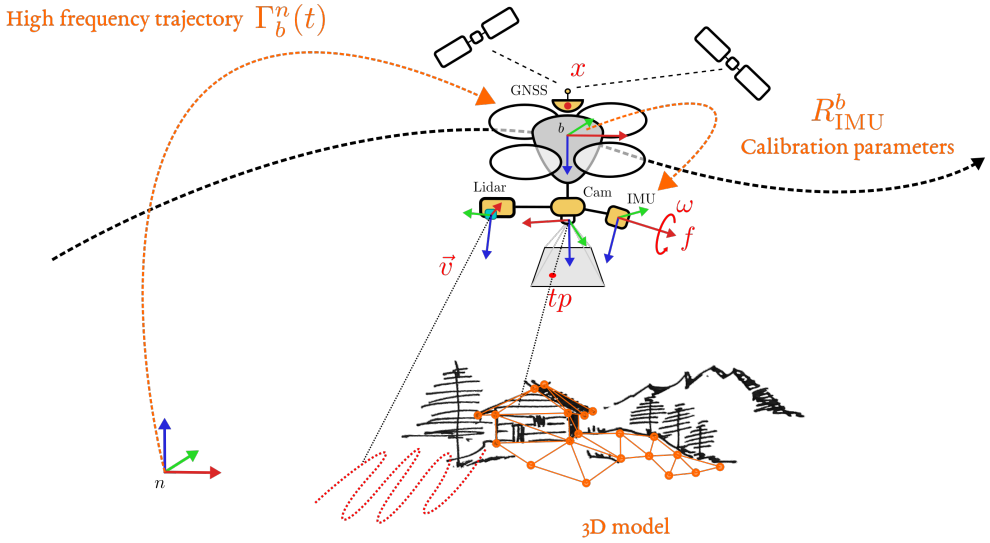
ODyN: an Online Dynamic Network Solver

EPFL

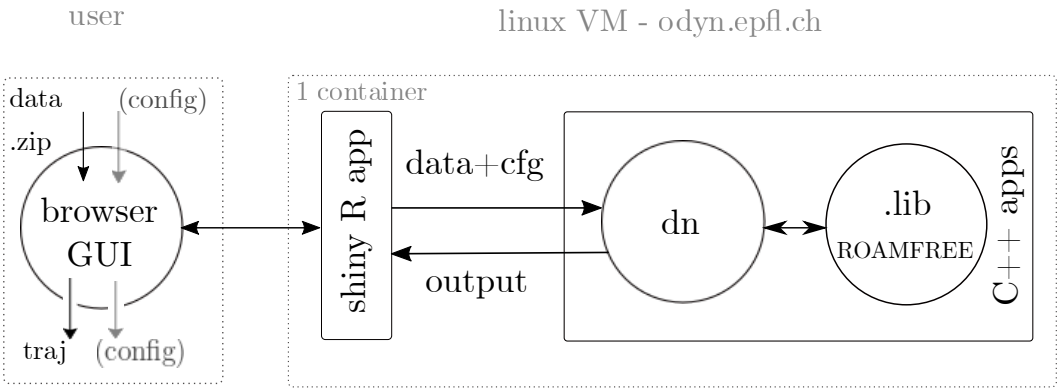
1. A solver for DN adjustment problems that run in the web browser
2. Based on the open-source ROAMFREE sensor fusion library
3. Available for free at `https://odyn.EPFL.ch`
4. The processing happens on cloud servers
5. Introduced at this congress, for more info please see

Cucci, D.A., "ODyN: An Online Dynamic Network Solver For Photogrammetry And Lidar Geo-referencing", *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, V-1-2022, 153–159. [→ link]

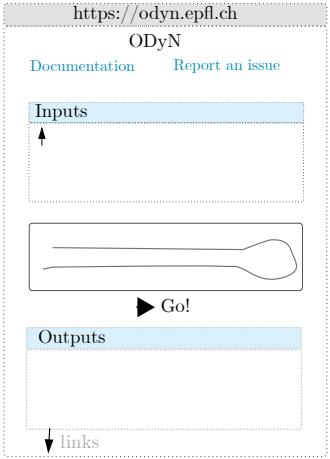
ODyN: an Online Dynamic Network Solver



ODyN: concept



ODyN: help?



Documentation (github)

- readme
- data

Inputs

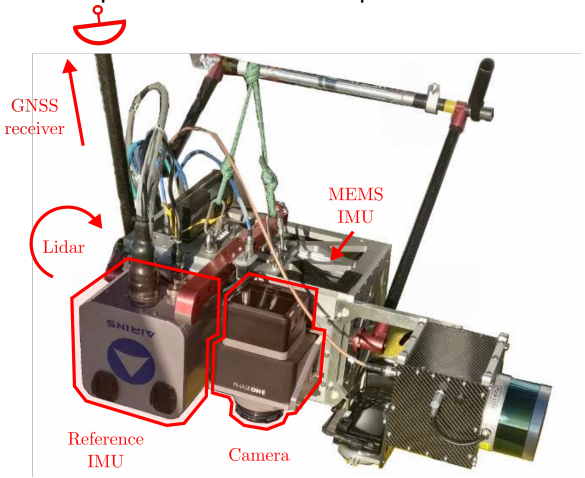
- data.zip (& config.)
- gui config (mounting, σ, etc.)

Outputs

- plots + stats
- estimated param.
- adjusted trajectory (link)

Example 1 - IMU + GNSS

Reference and UAV grade sensor mounted on the same platform on an helicopter.



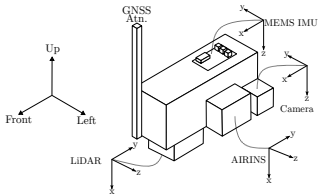
Example 1 - IMU + GNSS

Experiment (subset of):

Vallet et al. (2020), "Airborne And Mobile Lidar, Which Sensors For Which Application?" *ISPRS Archives* 43 (2020): 397-405.

DN - Processed:

1. Brun et al. (2022), "Lidar point–to–point correspondences for rigorous registration of kinematic scanning in dynamic networks", *J. ISPRS*, 89: 185-200 (open access).
2. Mouzakkidou et al. (2022), "On the benefit of concurrent adjustment of active and passive optical sensors with GNSS & raw inertial data", *ISPRS Annals* (Congress - Nice).



Sensors

- drone-like
- reference

Example 1 - IMU + GNSS

Goal:

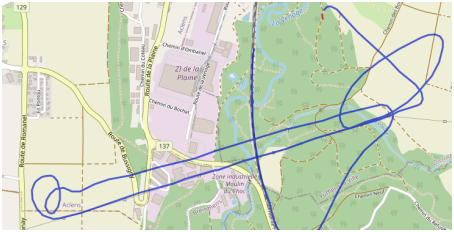
IMU(t, $\Delta\theta$, Δv)+GNSS (PPK) $\rightarrow$ PVA(t)

Needed:

1. `IMU.txt`
2. `GNSS.txt` (or `.cmb`)

Optional:

3. `initial_guess.txt` (or `.out`)
4. `reference.txt` (or `.out`)
5. `configuration.RData`



Dynamic network (online App)
odyn.epfl.ch

Example 1 - IMU + GNSS

Download data

- odyn.epfl.ch / Documentation (new TAB)
- /Examples/Configuration
- → data/configuration.RData
- /Examples/Inertial Navigation
- → data/INS.zip

Upload & process

- odyn.epfl.ch
- upload data [.zip]
- upload config [.Rdata]
- → Go!
- . . . analyze

Example 1 - INS/GNSS navigation
ODyN demo

More results on INS/GNSS with DNs

Cucci, D. A., et al. "A general approach to time-varying parameters in pose-graph optimization." *2017 European Navigation Conference (ENC). IEEE*, 2017. [→ link]

1. $\approx$ 20 min of car driving, 9.5 km in Vuarrens(CH)
2. tested: MEMS IMU + standalone GNSS.
3. reference: nav. grade IMU + carrier-phase differential GNSS.

We compare

1. DN with respect to a Kalman smoother (Stebler, 2013)
2. ... different inertial sensor biases sampling rates

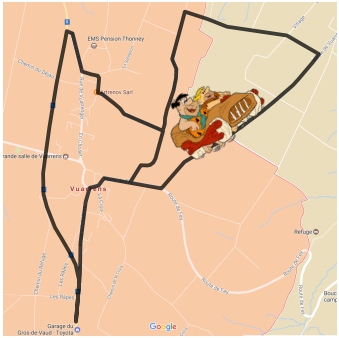


Figure: The trajectory.

Results

	ROAMFREE			NAVPROC
	5s-2.5s	30s-10s	90s-30s	
ENU [m]	0.97 (-5%)	0.97 (-5%)	0.95 (-7%)	1.02
	1.54 (-1%)	1.54 (-1%)	1.51 (-1%)	1.53
	2.96 (-2%)	2.91 (-5%)	2.45 (-15%)	2.89
RPH [deg]	0.09 (-52%)	0.09 (-51%)	0.10 (-42%)	0.18
	0.12 (-47%)	0.12 (-47%)	0.12 (-45%)	0.23
	1.44 (-29%)	1.42 (-30%)	1.52 (-25%)	2.02

Table: East-North-Up (ENU) and Roll-Pitch-Heading (RPH) error RMS with respect to reference.

A very recent work

Wen, W., et al. "Factor graph optimization for GNSS/INS integration: A comparison with the extended Kalman filter." *NAVIGATION, Journal of the Institute of Navigation* 68.2 (2021): 315-331.

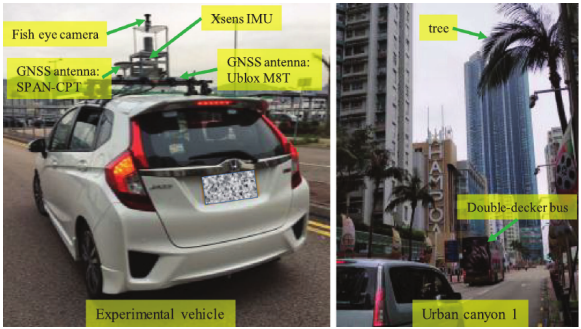


Figure: Experimental setup

A very recent work

Wen, W., et al. "Factor graph optimization for GNSS/INS integration: A comparison with the extended Kalman filter." *NAVIGATION, Journal of the Institute of Navigation* 68.2 (2021): 315-331.

1. DN (factor-graph) vs Kalman Filtering (not smoothing, sadly)
2. Loosely and **tightly** coupled GNSS
3. Urban canyons (they look only at position)

All data	Loosely coupled EKF	Tightly coupled EKF	Loosely coupled FGO	Tightly coupled FGO
Mean error	9.14 m	8.03 m	7.01 m	3.64 m
Std	7.60 m	7.15 m	6.41 m	2.84 m
Used Time	0.053 s	0.071 s	15.41 s	75.30 s

Figure: Positioning performance and computational load (time)

Optical constrains

Optical constraints

1. Photogrammetry: Image observations / tie-points / 2D - 2D correspondences
2. Laser scanning: LiDAR observations / 3D - 3D correspondences

Photogrammetry

Aerial Triangulation

1. Automatic identification of feature points in images
2. Bundle Adjustment $\rightarrow$ camera poses, features 3D position, calibration (optionally)
3. Absolute scale and geo-referencing via Ground Control Points (GCPs)

Issue: GCPs are time/cost intensive

Quasi-systematic errors

Poorly distributed / insufficient GCPs lead to large-scale deformations.

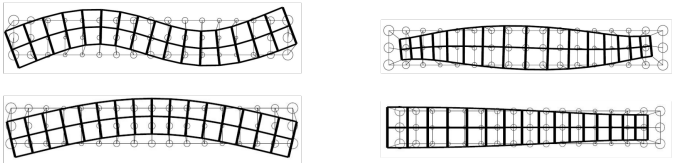


Figure: Typical corridor *quasi-systematic* errors.
From W. Förstner, UAV-g, Sept, 4th 2017.

Assisted Aerial Triangulation

GNSS/Inertial fusion gives priors for the camera position and orientation.

1. Global accuracy with less/no GCPs
2. Complex pipeline

Issues: initial alignment in KF, bias/boresight determination, prior uncertainties, etc.

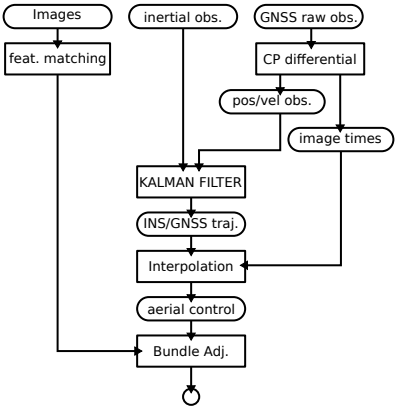


Figure: AAT pipeline

A single step, joint-adjustment solution.

Higher redundancy $\rightarrow$ stronger conditions on parameter estimation.

Issues: less to none :)

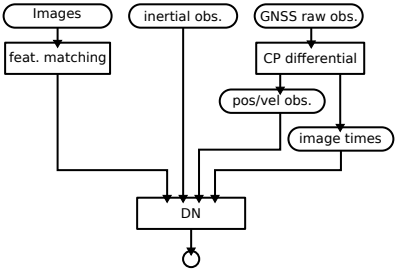
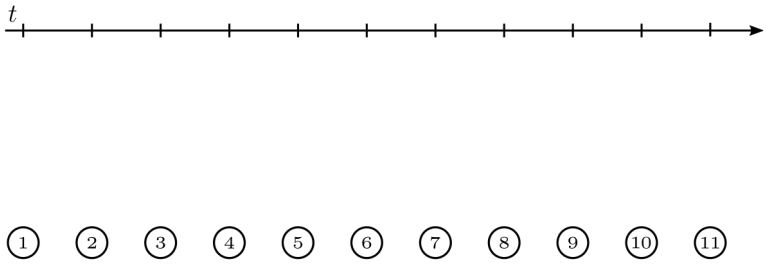
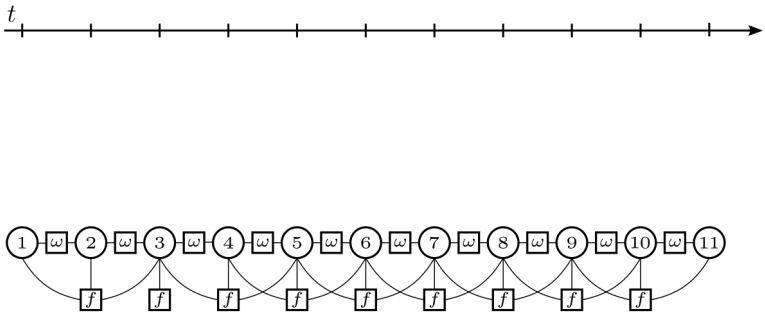
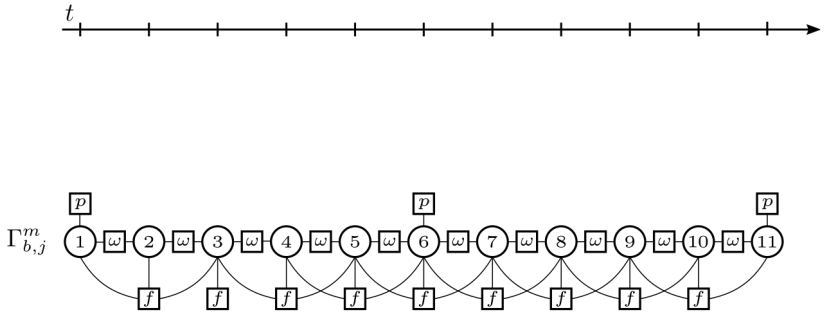


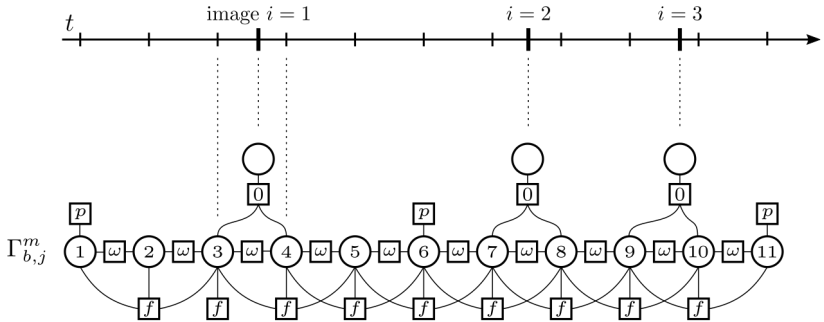
Figure: DN pipeline

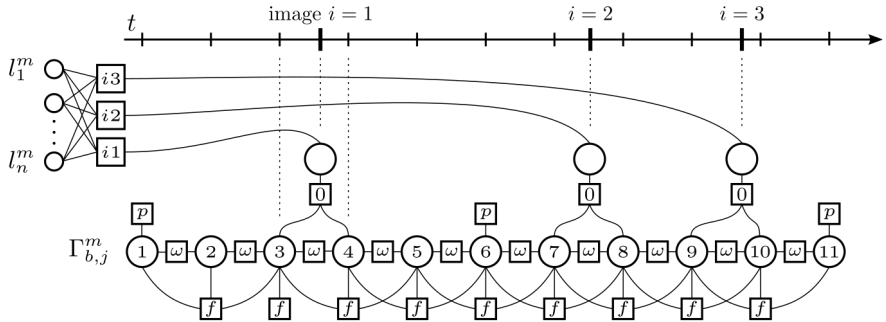

t
$\Gamma^m_{b,j}$
1
2
3
4
5
6
7
8
9
10
11


t
$\Gamma^m_{b,j}$
1 ω 2 ω 3 ω 4 ω 5 ω 6 ω 7 ω 8 ω 9 ω 10 ω 11
f f f f f f f f f

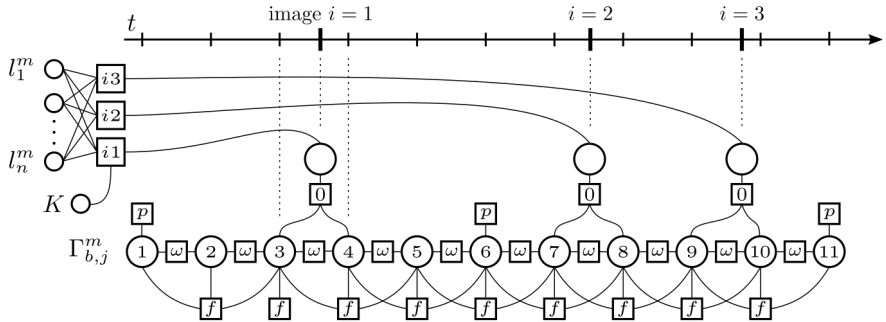

t
$\Gamma^m_{b,j}$
p
ω
f
1
2
3
4
5
6
7
8
9
10
11

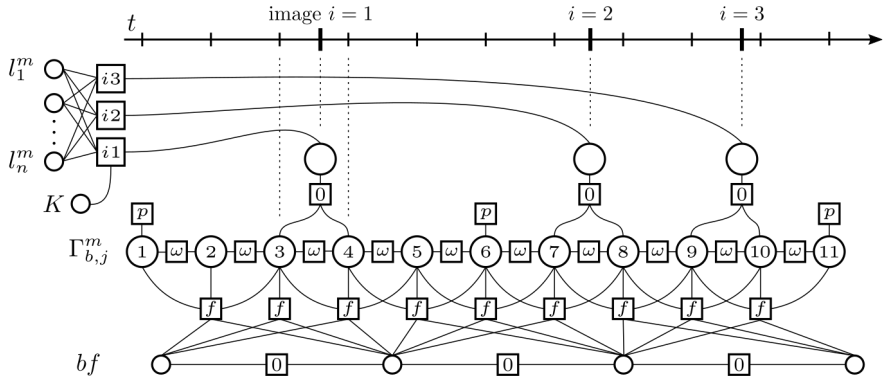
Dynamic Networks for Navigation and Mapping



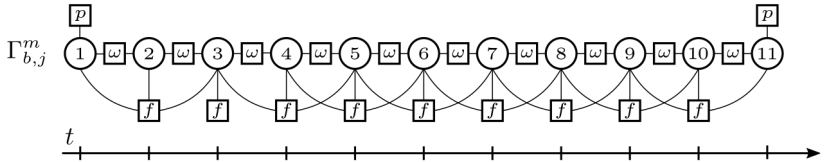

t
image $i = 1$
$i = 2$
$i = 3$
l_1^m
l_n^m
$i3$
$i2$
$i1$
0
p
$\Gamma_{b,j}^m$
1
ω
2
3
4
5
6
7
8
9
10
11
f

Dynamic Networks for Navigation and Mapping



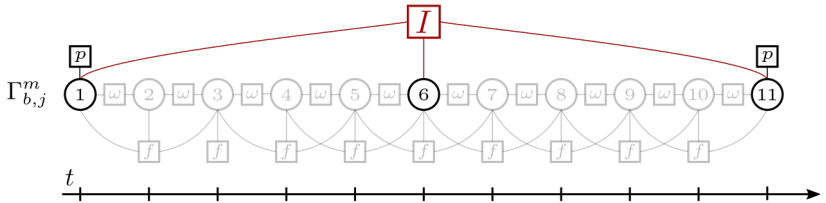

t
image $i = 1$
$i = 2$
$i = 3$
l_1^m
l_n^m
K
$i3$
$i2$
$i1$
0
p
$\Gamma_{b,j}^m$
1
ω
2
3
4
5
6
7
8
9
10
11
f
bf

High-rate IMU



1. Bloating number of unknowns (e.g.: 1 KHz IMU, minimum 6×1000 unknowns/s)
2. Ill-conditioning of the normal equations

IMU pre-integration



1. Reduces the number of measurement **without** loss of information (no downsampling)
2. Complications with IMU stochastic processes (not entirely understood)

IMU pre-integration

EPFL

1. Original publication:

   Lupton, T., and Sukkarieh, S. "Efficient integration of inertial observations into visual SLAM without initialization." *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE*, 2009.

2. All-in with Lie Algebra

   Barrau, A., and Bonnabel, S. "A mathematical framework for IMU error propagation with applications to preintegration." *2020 IEEE International Conference on Robotics and Automation (ICRA).* IEEE, 2020.

3. Our contribution: **gravity models** and **Earth rotation** in IMU-preintegration:

   Cucci, D. A., and Skaloud, J. "On Raw Inertial Measurements in Dynamic Networks." *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences* 4 (2019). [→ link]

Visual-inertial Odometry

This is very, very similar to what is called visual-inertial odometry today, e.g.,

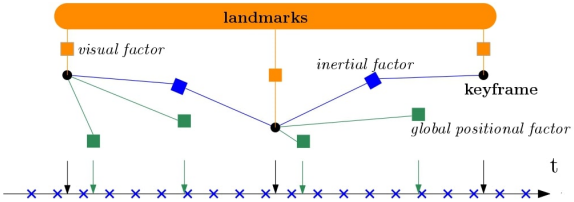


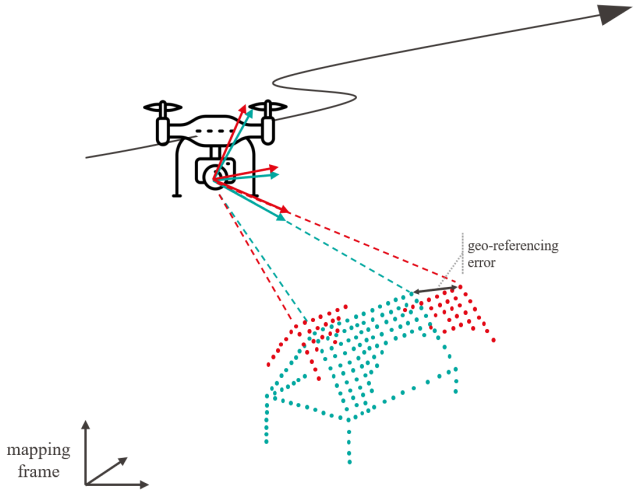
Figure: Representation of the proposed optimization-based multi-sensor fusion in [1]

[1] Cioffi, G., and Scaramuzza, D.. "Tightly-coupled fusion of global positional measurements in optimization-based visual-inertial odometry." *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE*, 2020.

Laser scanning

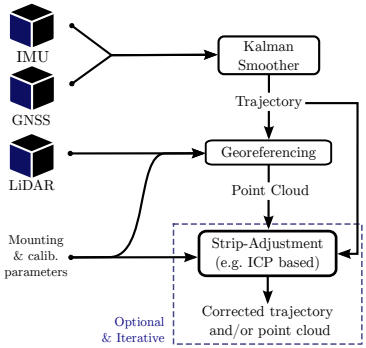
Laser scanning

1. In rotating scanners the pose of the sensor is different for every acquired point
2. We need a **high-frequency** trajectory to correctly geo-reference lidar observations
3. Today, this is obtainable only with **inertial sensors** $\rightarrow$ DNs :)



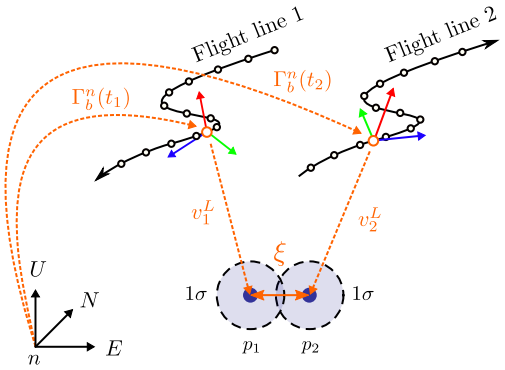
Conventional processing methodology:

1. INS/GNSS navigation to get a **high-frequency** trajectory
2. Direct geo-referencing of laser observations
3. (Optional) Corrections **at point-cloud level**



3D - 3D correspondences

Matching lidar observations corresponding to the same point on ground (up to the GSD).

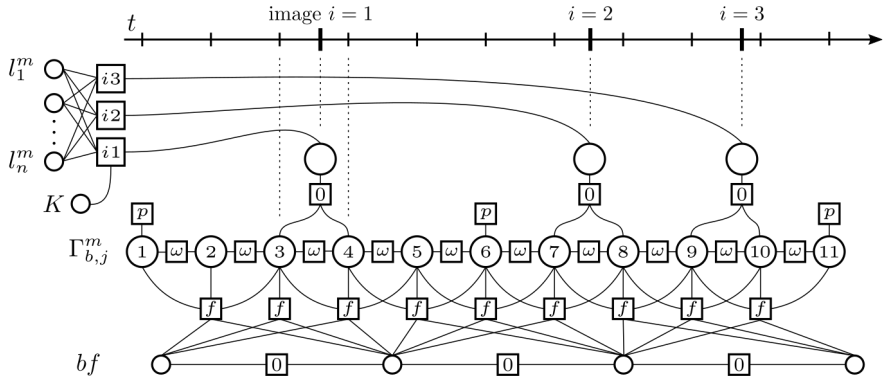


Which can be expressed as:

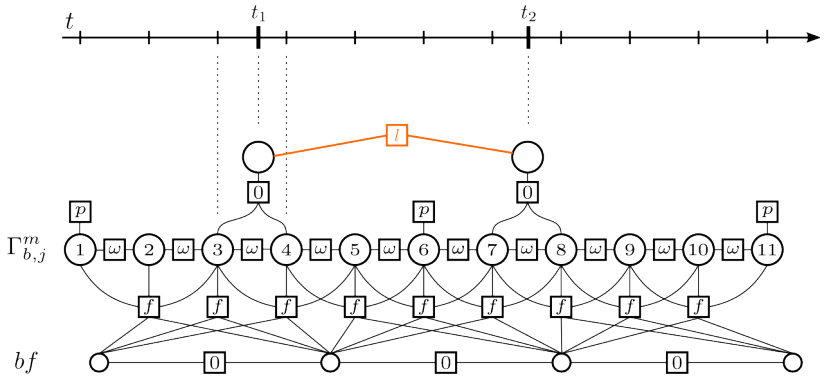
$$\tilde{\Gamma}^n_{b,t_1} R^b_L v^L_1 - \tilde{\Gamma}^n_{b,t_2} R^b_L v^L_2 =$$

$$R^b_L v^L_1 - \underbrace{\left[\tilde{\Gamma}^n_{b,t_1}\right]^{-1} \tilde{\Gamma}^n_{b,t_2}}_{\Gamma^{b,t_1}_{b,t_2}} R^b_L v^L_2 = 0 + \xi,$$

DN structure (photogrammetry)



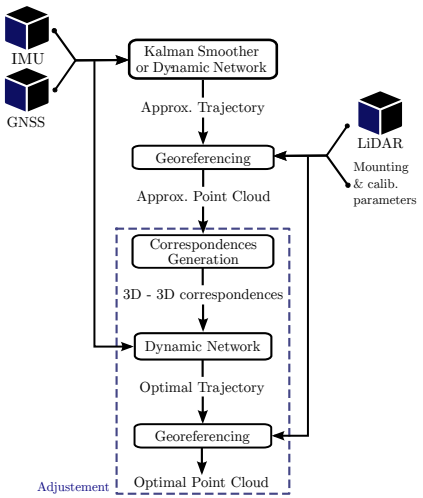
DN structure (laser scanning)



Laser scanning

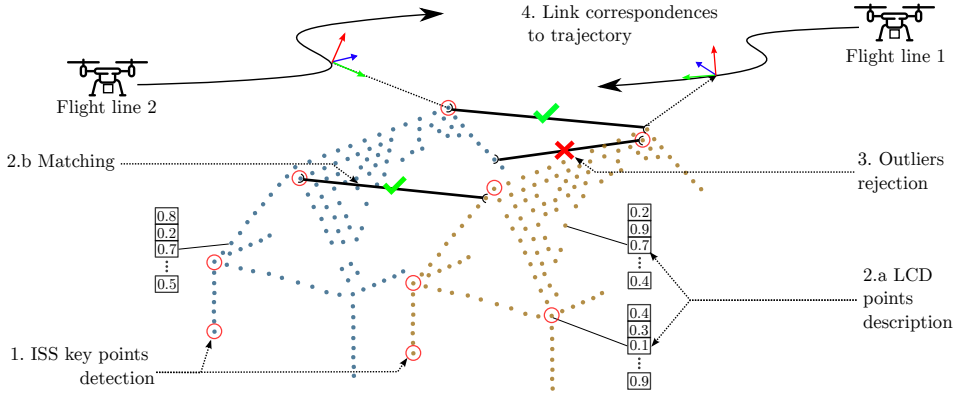
with DNs:

1. INS/GNSS navigation (possibly with DNs) to get an **approximated** trajectory
2. Identify 3D - 3D correspondences
3. Fuse everything with DNs to get a **corrected** trajectory
4. Direct geo-referencing of laser observations



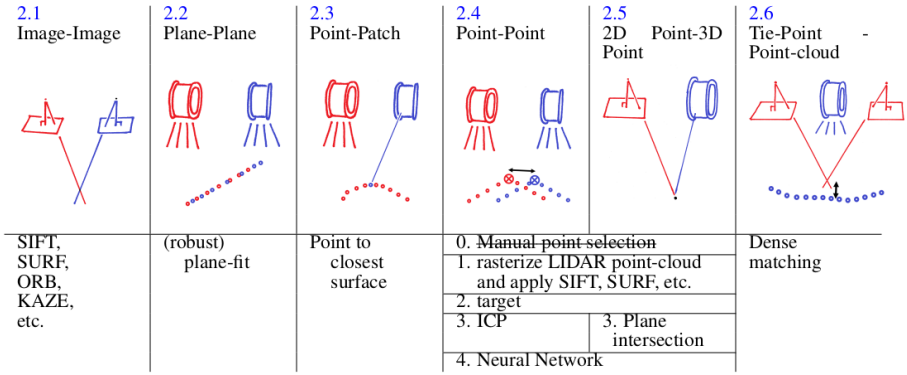
How do I obtain 3D - 3D correspondences?

Brun, A. **et us**, "LiDAR Point–to–point Correspondences for Rigorous Registration of Kinematic Scanning in Dynamic Networks." *ISPRS Journal of Photogrammetry and Remote Sensing* 189 (2022): 185-200. [→ link]



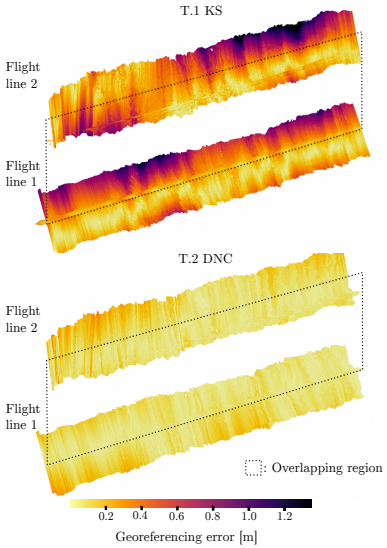
More correspondences?

Cledat, E., and J. Skaloud. "Fusion Of Photo With Airborne Laser Scanning" *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 1 (2020): 173-180.



Results

Better trajectory $\rightarrow$ Better point cloud.



Example 2 - INS/GNSS with 3D - 3D correspondences

ODyN demo

INS/GNSS with 3D – 3D correspondences

Input: `LiDAR_p2p.txt` see documentation

```
396765.649713,396660.533676,271.420000,-2.570000,18.060000,263.280000,-2.520000,-135.520000
396767.558842,396659.053466,263.810000,-2.520000,-0.480000,259.870000,-2.470000,-126.380000
```

Other files: as in `INS.zip`

Configuration

- 3 lever-arms (1 new)
- 2 boresight (1 new)

What we want?

- adjusted trajectory
- estimated parameters (e.g., boresight, IMU biases)

What we do after it?

- DiSO

Compare different processing pipelines in a challenging corridor mapping scenario

- AAT with absolute **position control** from GNSS [PC]
- AAT with absolute **orientation control** from previous INS/GNSS integration [AC]
- Dynamic Networks [DN]

A fixed wing UAV, 1.5m wingspan, 600 g payload including:

1. Custom 20 Mpx camera developed by IGN, France with Zeiss Biogon 35 mm lenses,
2. Gecko4Nav redundant IMU board with two Intersense NavChip MEMs IMUs,
3. Topcon B110 GPS/GLONASS L1/L2 receiver.

Figure: The payload

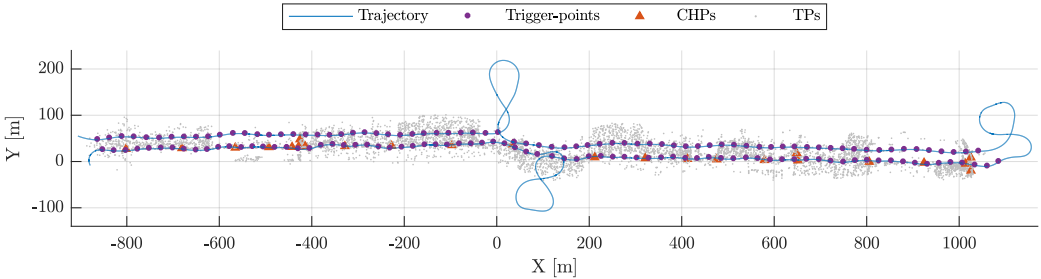
Software

1. [PC] GNSS processing with Novatel Grafnav, then Agisoft Photoscan
2. [AC] GNS/INS with Applanix Posproc (custom), then Agisoft Photoscan
3. [DN] Tiepoints from Photoscan, then joint DN adjustment with ROAMFREE

The flight

Two Km corridor mapping flight (released as open-data, see later on)

1. two flight lines at 100 m elevation (GSD $\sim$ 2 cm)
2. Alignement maneuvers
3. 4 GCPs, 20 checkpoints



	mean						std						RMS					
	E		N		U		E		N		U		E		N		U	
	mm	px	mm	px	mm	px	mm	px	mm	px	mm	px	mm	px	mm	px	mm	px
DN	**-1**	**-0.03**	**-3**	**-0.15**	**0**	**0.01**	**10**	**0.51**	13	0.64	28	1.41	**10**	**0.51**	13	0.66	28	1.41
AC	-13	-0.67	-8	-0.39	16	0.82	15	0.77	7	0.35	16	0.79	20	1.02	10	0.52	23	1.14
PC	-10	-0.48	-4	-0.18	11	0.53	12	0.60	**6**	**0.30**	**13**	**0.67**	15	0.76	**7**	**0.35**	**17**	**0.85**

Table: Statistic of the error in the estimated checkpoint position.

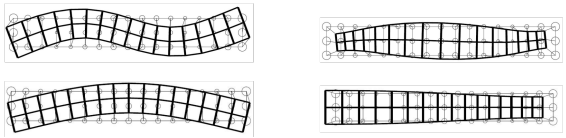
Interpretation:

1. DN is the best in terms of mean error
2. **BUT** it's more noisy (higher standard deviation)

Why?

1. The camera is too good for the IMU
2. The IMU introduces noise but helps fighting large-scale systematic errors.

Overall statistics hide spatially correlated errors, i.e., these effects:



Global deformations

Let's imagine the error on the ground is a function of the position, e.g.

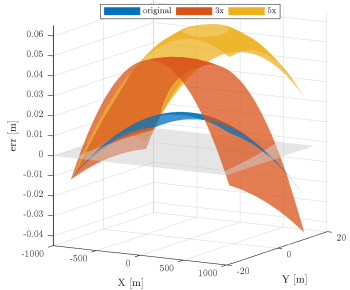
$$err = f(x, y, z) + \xi, \xi \sim \mathcal{N}(0, \Sigma) \tag{1}$$

The terrain is almost flat, we restrict to a second order polynomial function in x and y:

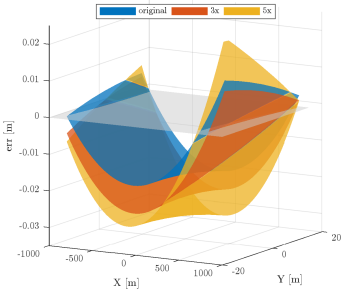
$$err = ax^2 + by^2 + cxy + dx + ey + f. \tag{2}$$

1. We downsample the images by a factor of 3x and 5x
2. We fit such model to the obtained residuals for [PC] and [DN]

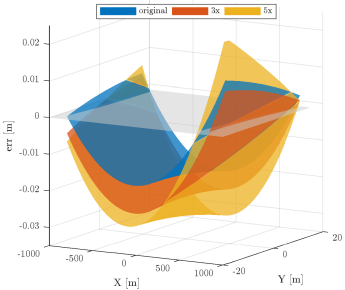
Global deformations [PC]



(a) err_x

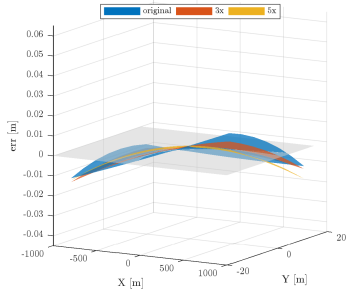


(b) err_y

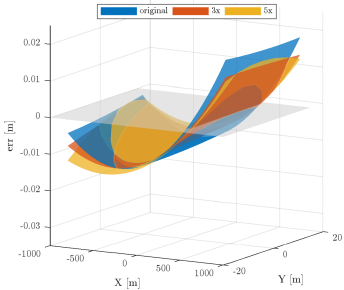


(c) err_z

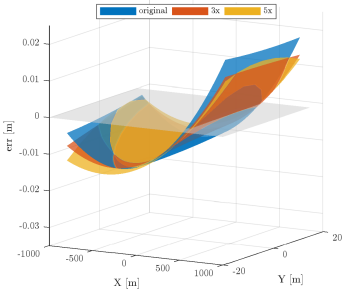
Global deformations [DN]



(d) err_x

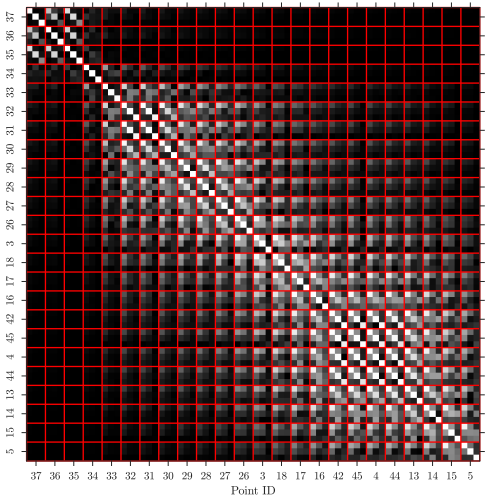


(e) err_y

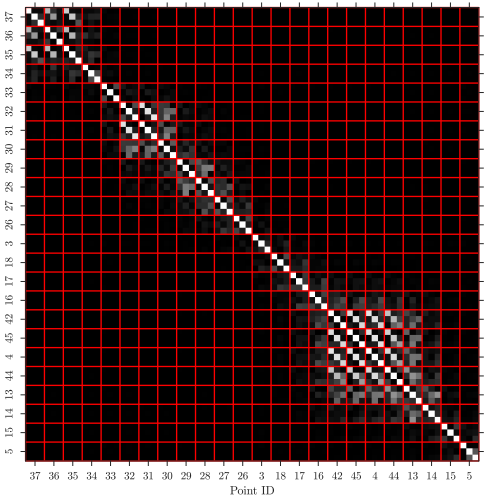


(f) err_z

Covariance [PC]



Covariance [DN]



Discussion

DN vs PC:

1. will have better decorrelation of the residuals.
2. will improve provided that the IMU is sufficiently good

DN vs AC:

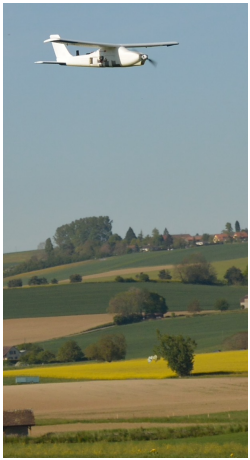
1. substantially simpler pipeline.
2. less experience-intensive process, easier to get better results
   (no alignment problems, better bias determination, boresight calibration)

Can I have a look?

The data for this flight (and similar ones) has been released as **open-data** (at last? ISPRS Congress).

1. Why: access to high quality data **with reference** is not obvious.
2. Goal: benchmarking traditional approaches and faster testing of new concepts.

Skaloud, J., Cucci, D. A., and Joseph Paul, K. "Fixed-wing Micro UAV Open Data With Digicam And Raw INS/GNSS" *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences* (2021). [→ link]



Example 3 - INS/GNSS with image observations
ODyN demo

Home work

- *Task.* Combine Example 2 with Example 3 into a common adjustment of all optical (lidar/photo) and navigation (GNSS/raw-IMU) data.
- *Prerequisite.* Create a new `.zip` and configuration (via GUI).
- *Analysis.* Is there an improvement?
- *Hint.* Look at the details, e.g.,→

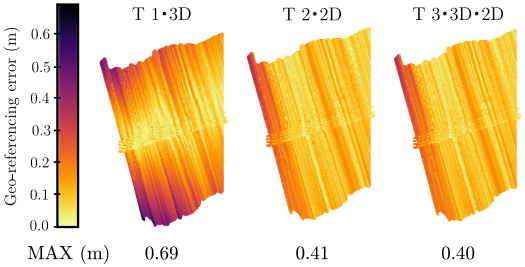


Figure: Extreme west part of the point cloud.

Mouzakkidou, K., Cucci, D.A., and Skaloud, J., "On the benefit of concurrent adjustment of active and passive optical sensors with GNSS & raw inertial data", *ISPRS Annals*, V-1-2022: 161-168. [→ link]

Another example

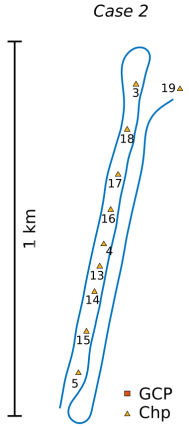
1. 1 Km corridor
2. Same IMU as before
3. Worse camera (Sony NEX-5R)
4. NO GCPs

We compare

1. AAT (Kalman smoother and then Bundle-adjustment)
2. DN

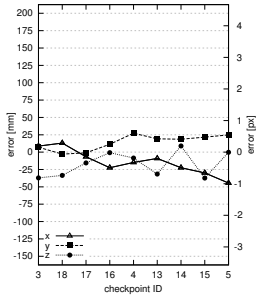
More cases in:

Cucci, D. A., Rehak, M., and Skaloud, J. "Bundle adjustment with raw inertial observations in UAV applications." *ISPRS J. of Photogrammetry and Remote Sensing* 130 (2017): 1-12. [→ link]



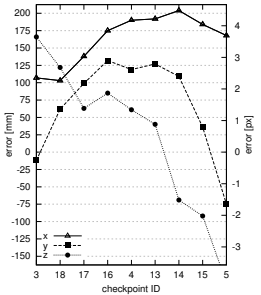
Results

DN



	E	N	U
mean (mm)	−14	+14	−17
RMS (mm)	22	17	24

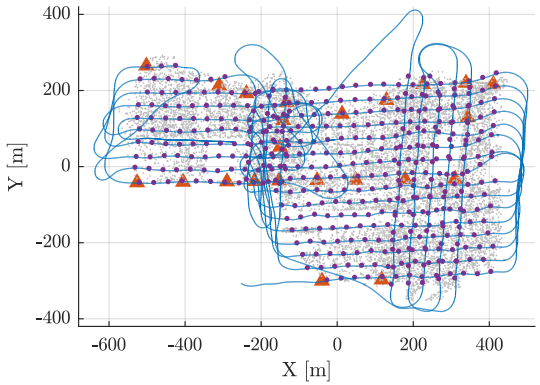
AAT



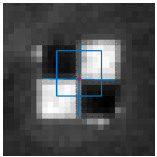
	E	N	U
mean (mm)	+160	+66	+12
RMS (mm)	164	95	108

Example 4 - INS/GNSS with image observations (UAV)
ODyN demo

Open UAV data



- 440 images, IGN-digicam/Biogon 35mm
- 26 lines, 2 levels, ∼35 min
- 65/45 % overlap, 1/5000 s
- 2× raw INS 400 Hz, GNSS 10 Hz
- 2-4 cm GSD, 21 GCPs at 0.1 pix



Skaloud, J., Cucci, D.A., and Paul, K.J. "Fixed-wing micro UAV open data with digicam and raw INS/GNSS". *ISPRS Annals* 51 (2021): 105-111. [→ link]

Conclusions

1. DNs are a very general sensor fusion framework. Many things can be added:
   - 1.1 more navigation sensors: magnetometers, barometers, ...
   - 1.2 other exteroceptive sensors, e.g., LiDARs,
   - 1.3 *robust* fusion, outlier rejection, fault detection and isolation, ...
2. The one-shot, all-together, tight/joint adjustment is better than multi-stage or cascade processing methods
3. Redundancy can be *quantified*
   - 3.1 e.g., redundancy numbers, parameters cross-correlation, ...
4. Redundancy allows for *calibration*
   - 4.1 lever arms, boresights, scale factors, magnetic distortion, camera calibration, etc.

Conclusions

1. *General.* DN is the rigorous problem formulation for the common adjustment of navigation/optical data. It improves modeling, error propagation & parameter estimation.
2. *On INS/GNSS redundancy*. Observability of (IMU time-correlated) errors is limited by trajectory dynamics. Inclusion of optical constrains mitigates this weakness.
3. *Weak geometry (e.g., corridor, intermittent GNSS, point/line scanners)*. One-step (DN), rather than sequential (KF + AT), adjustment is preferable for geometry weak scenarios as it limits the deformations in object space.
4. *Example drones – nominal GNSS / low-cost IMU / lidar.* The inclusion of (a subset of) point-to-point correspondences into DN effectively mitigated the "wavy" patterns in the point cloud registration caused by the attitude errors in the drone trajectory.

Questions?