dyson Robotics Lab Imperial College London

Spatial Perception for Mobile Robots

Stefan Leutenegger

Implementing Machine Vision with FPGA and SoC Platforms 18/05/2017

Today's Mobile Robots – Some Examples

Space exploration
NASA Curiosity

Autonomous driving
Google Car

Robotic vacuum cleaning
Dyson 360 Eye

Aerial inspection
Ascending Technologies Firefly

About Us

Our Mission: Robot Vision

- In the real world
- In a real-time loop
- For a purpose

Principal Investigators:

Prof. Andrew Davison

Dr Stefan Leutenegger

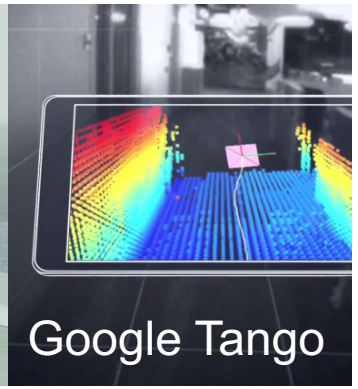
Lab Manager:

Iosifina Pournara

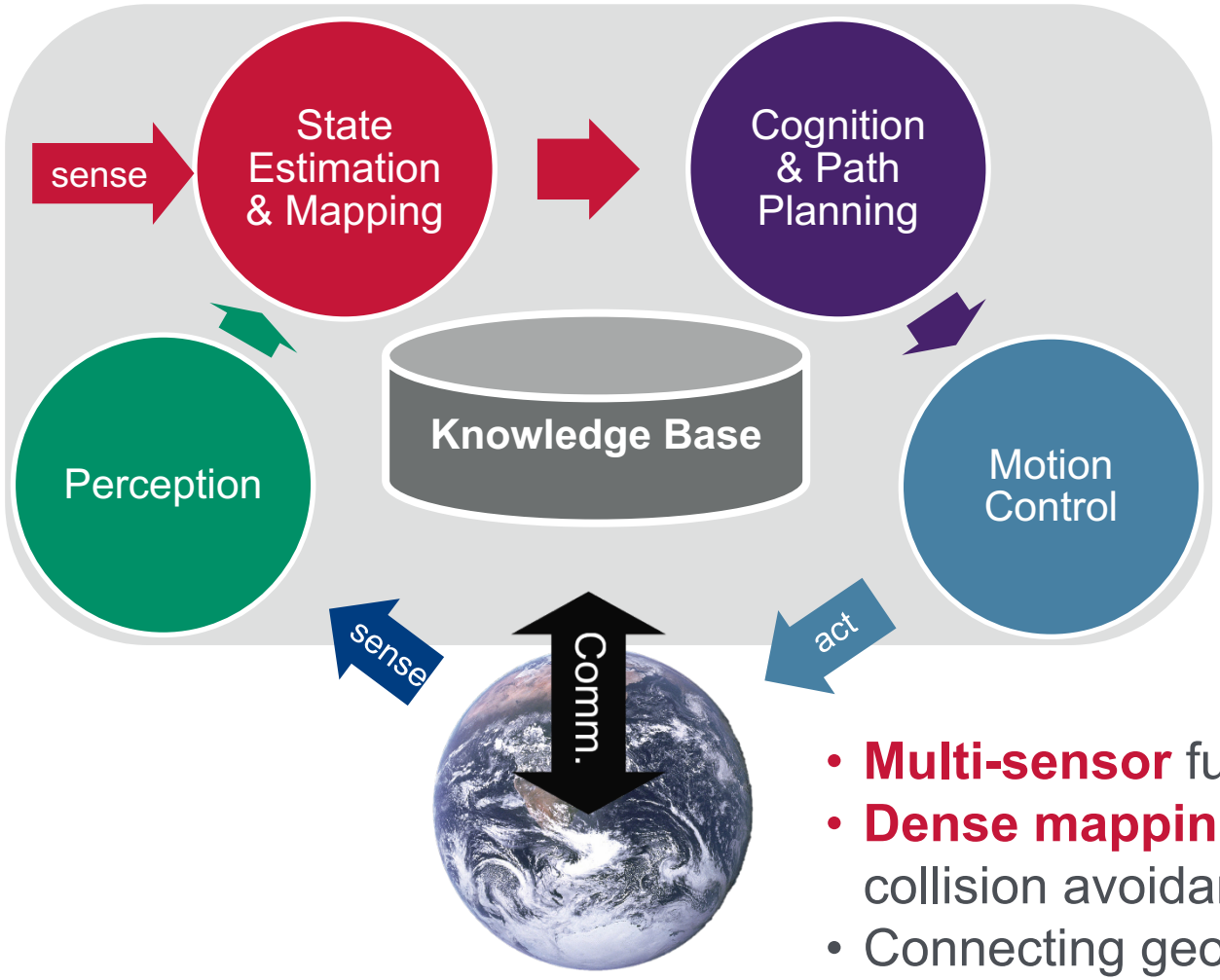
1 Research Engineer

4 Post-Docs

10 PhD-Students



The Perception-Action Cycle in Mobile Robotics



Requirements:
- robustness,
- efficiency,
- accuracy

of the co-design of
- sensor suite,
- processing hardware,
- algorithms,
- software.

- **Multi-sensor** fusion for robustness
- **Dense mapping** for planning, collision avoidance and interaction
- Connecting geometry to **semantics**

Probabilistic SLAM Formulation

Given

Measurements $\mathbf{z}$ are samples from a **distribution** $p(\mathbf{z}|\mathbf{x})$ given the variables $\mathbf{x}$ (robot states plus usually the map).

Find

- Values for variables $\mathbf{x}$ that best explain all the measurements (**maximum likelihood, ML**), or
- values for $\mathbf{x}$ that best explain all the measurements and a prior $p(\mathbf{x})$ (**maximum a posteriori, MAP**).

What our research consists of:

- **Decide** for a state and map representation $\mathbf{x}$.
- **Model** the likelihood function $p(\mathbf{z}|\mathbf{x})$ and prior $p(\mathbf{x})$ (e.g. regulariser).
- **Choose** approximations to solve the ML/MAP problem, e.g. marginalisation (filtering), or iterative minimisation.
- **Find** some way to bootstrap, associate data, and initialise.

Robot State

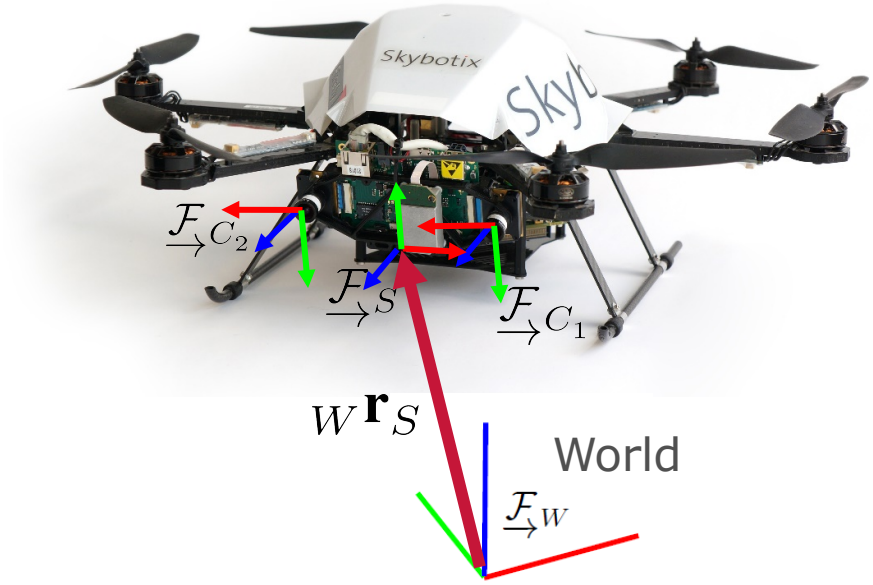
An example state vector:

$$\mathbf{x}_{\mathrm{R}}^{T} = \left[{}_{W}\mathbf{r}_{S}^{T}, \mathbf{q}_{WS}^{T}, {}_{W}\mathbf{v}^{T}, {}_{S}\mathbf{r}_{C_1}^{T}, \mathbf{q}_{SC_1}^{T}, {}_{S}\mathbf{r}_{C_2}^{T}, \mathbf{q}_{SC_2}^{T}, \mathbf{b}_{g}^{T}, \mathbf{b}_{a}^{T} \right]$$

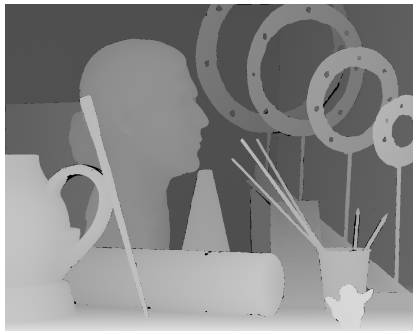
Position and orientation of frame S origin w.r.t. in W	Velocity of frame S origin expressed in W	Poses of cameras 1 and 2 relative to frame S	IMU gyro and accelerometer biases
Physical robot states		**Sensor extrinsics (here: camera)**	**Sensor intrinsics (here: IMU)**

Notes:

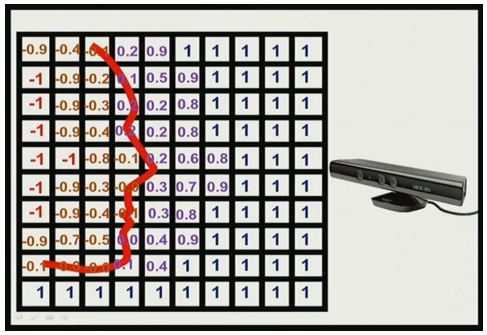
- This is just an example.
- We have to decide, **what** quantities should be estimated.
- We have to decide, how those quantities are **parameterised**.



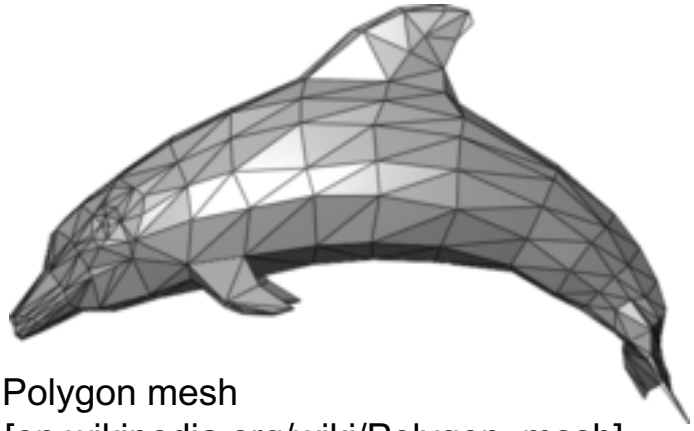
Map Representations



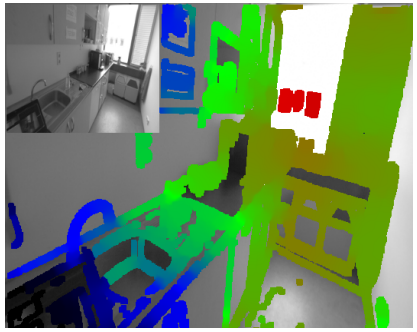
Depth maps
[vision.middlebury.edu]



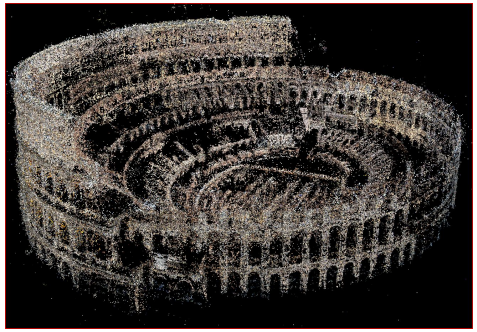
Truncated Signed Distance Function [pointclouds.org]



Polygon mesh
[en.wikipedia.org/wiki/Polygon_mesh]



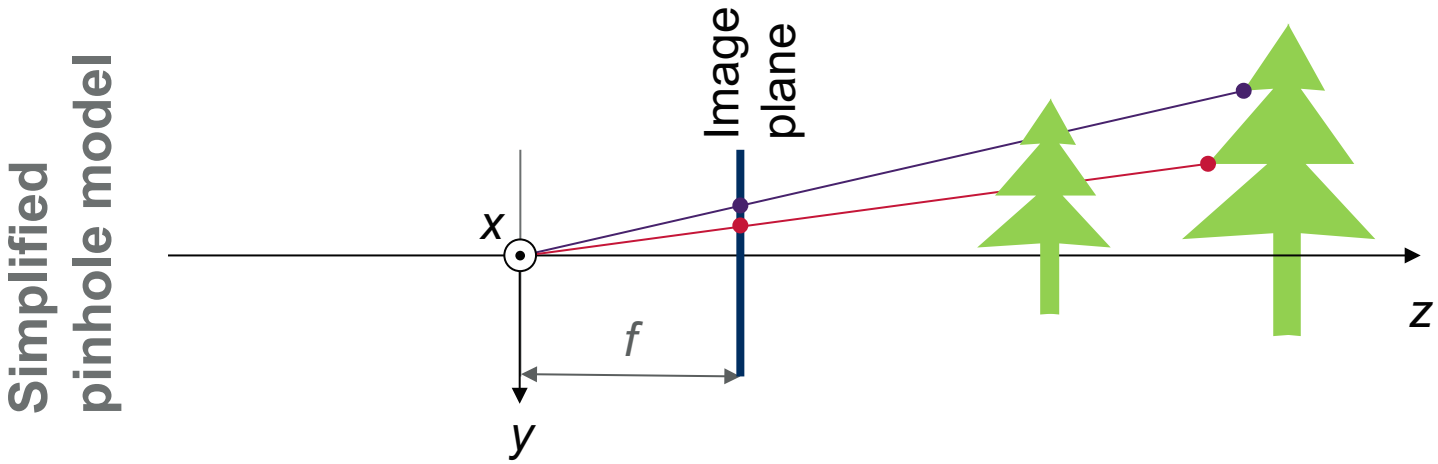
Semi-dense depth maps
[vision.in.tum.de]



Point clouds (here: sparse)
[grail.cs.washington.edu]

Surfel maps
[wp.doc.ic.ac.uk/thefutureofslam]

The Camera



Pinhole projection: $\begin{bmatrix} u \\ v \end{bmatrix} = \frac{f}{z} \begin{bmatrix} x \\ y \end{bmatrix}$

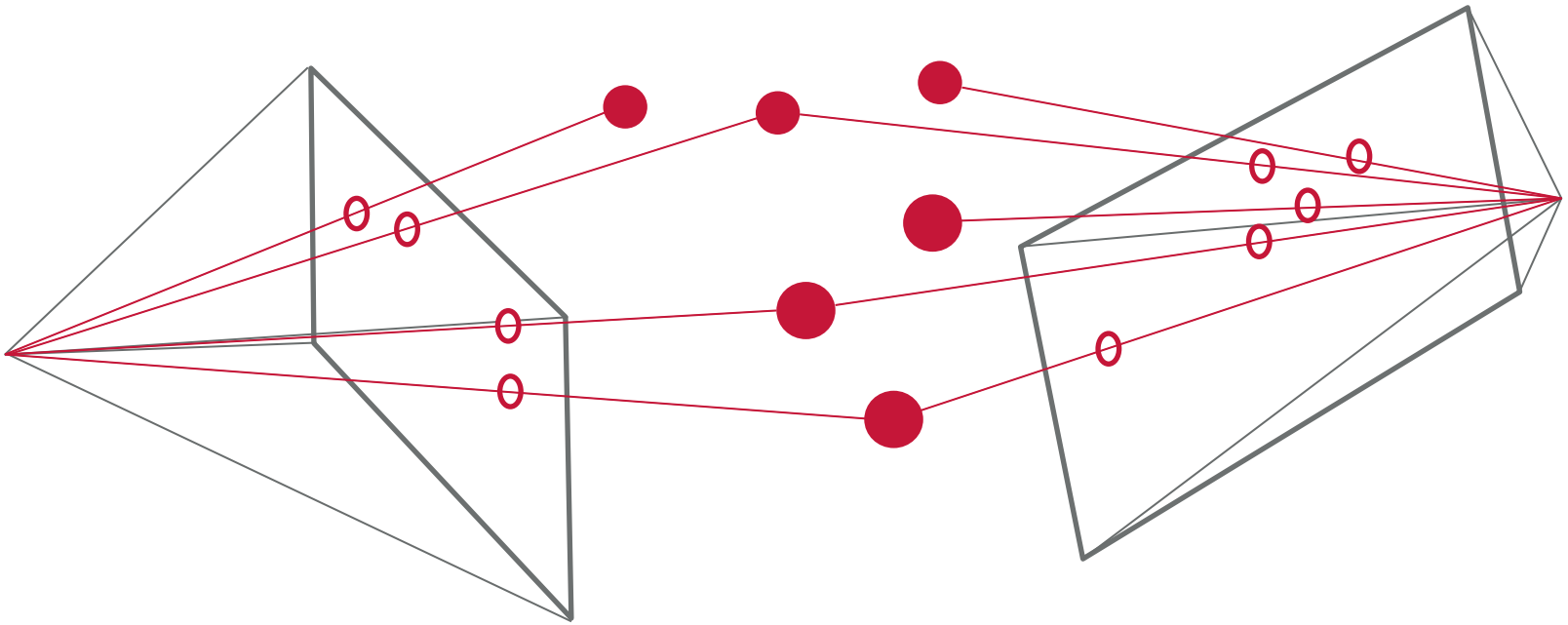
IMU Kinematics with Sensor Error Models

$$
\begin{aligned}
{}_W\dot{\mathbf{r}}_S &= {}_W\mathbf{v}\,, \\
\dot{\mathbf{q}}_{WS} &= \frac{1}{2}\begin{bmatrix} {}_S\tilde{\boldsymbol{\omega}} + \mathbf{w}_{\mathrm{g}} - \mathbf{b}_{\mathrm{g}} \\ 0 \end{bmatrix}^{\oplus} \mathbf{q}_{WS}\,, \\
{}_W\dot{\mathbf{v}} &= \mathbf{C}_{WS}\left({}_S\tilde{\mathbf{a}} + \mathbf{w}_{\mathrm{a}} - \mathbf{b}_{\mathrm{a}}\right) + {}_W\mathbf{g}, \\
\dot{\mathbf{b}}_{\mathrm{g}} &= \mathbf{w}_{b_{\mathrm{g}}}\,, \\
\dot{\mathbf{b}}_{\mathrm{a}} &= -\frac{1}{\tau}\mathbf{b}_{\mathrm{a}} + \mathbf{w}_{b_{\mathrm{a}}}\,.
\end{aligned}
$$

IMU biases

The Visual SLAM Problem

SLAM: Simultaneous Localisation and Mapping



Bundle Adjustment: Visual SLAM as an Optimisation Problem

Given: 2D detections $\tilde{\mathbf{u}}_{ij}$ of corresponding unknown 3D landmarks ${}_W\boldsymbol{l}_j$ in images $i = 1 \dots N$.

Find: 3D landmark positions ${}_W\boldsymbol{l}_j$ and 6D camera poses $[{}_W\mathbf{r}_{C_i}, \mathbf{q}_{WC_i}]^T$ where the images were taken.

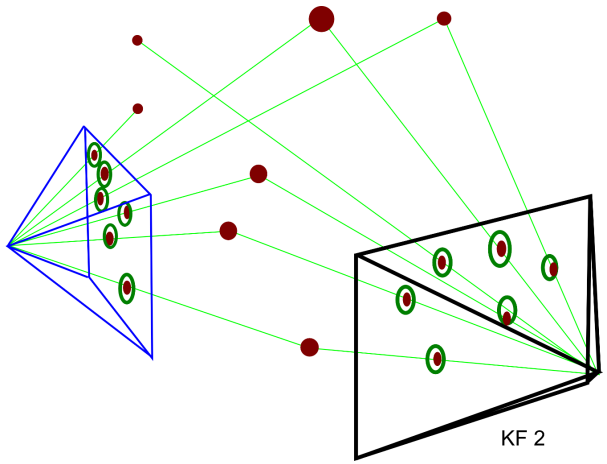
Solution: minimise the reprojection error

$$\mathbf{e}_{ij}({}_W\mathbf{r}_{C_i}, \mathbf{q}_{WC_i}, {}_W\boldsymbol{l}_j) = \tilde{\mathbf{u}} - \mathbf{u}(\boldsymbol{T}_{C_iW}\ {}_W\boldsymbol{l}_j),$$

over all images

$$\min \sum_{i=1}^{N} \sum_{j \in \mathcal{J}(i)} \|\mathbf{e}_{ij}\|^2,$$

Set of landmark indices as visible in camera *i*.



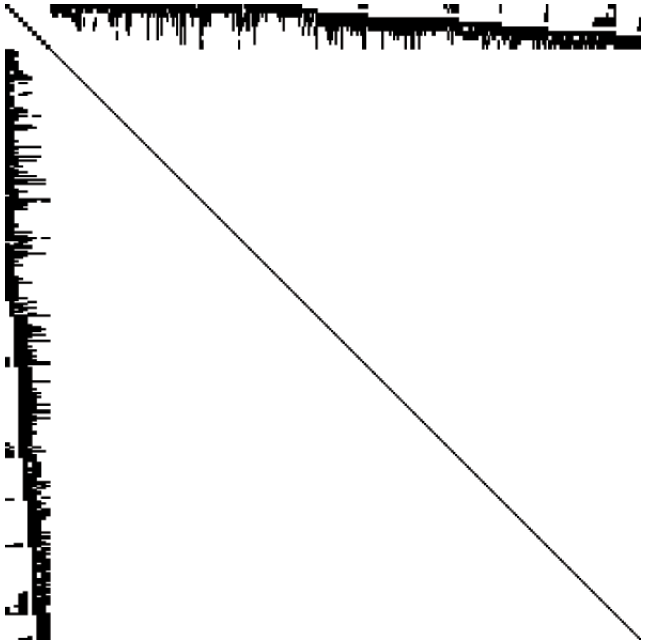
Applying Levenberg-Marquardt.

Note: you can use this exact same measurement term as an update in a filtering framework

Some Notes on Bundle Adjustment

- Typical problems contain many more points than poses
- Due to the sparsity of the linear equation system, it needs to be solved in a with some tricks, e.g.
  - Use Schur-Complement elimination of landmarks
  - Use sparse Cholesky decomposition
- In practise, **use an off-the-shelf solver** like
  - ceres: http://ceres-solver.org/
  - g2o: https://github.com/RainerKuemmerle/g2o

  This means, you will only have to implement the error function and Jacobians!



Example matrix A of a modestly sized problem.
https://en.wikipedia.org/wiki/Bundle_adjustment

Why Visual-Inertial

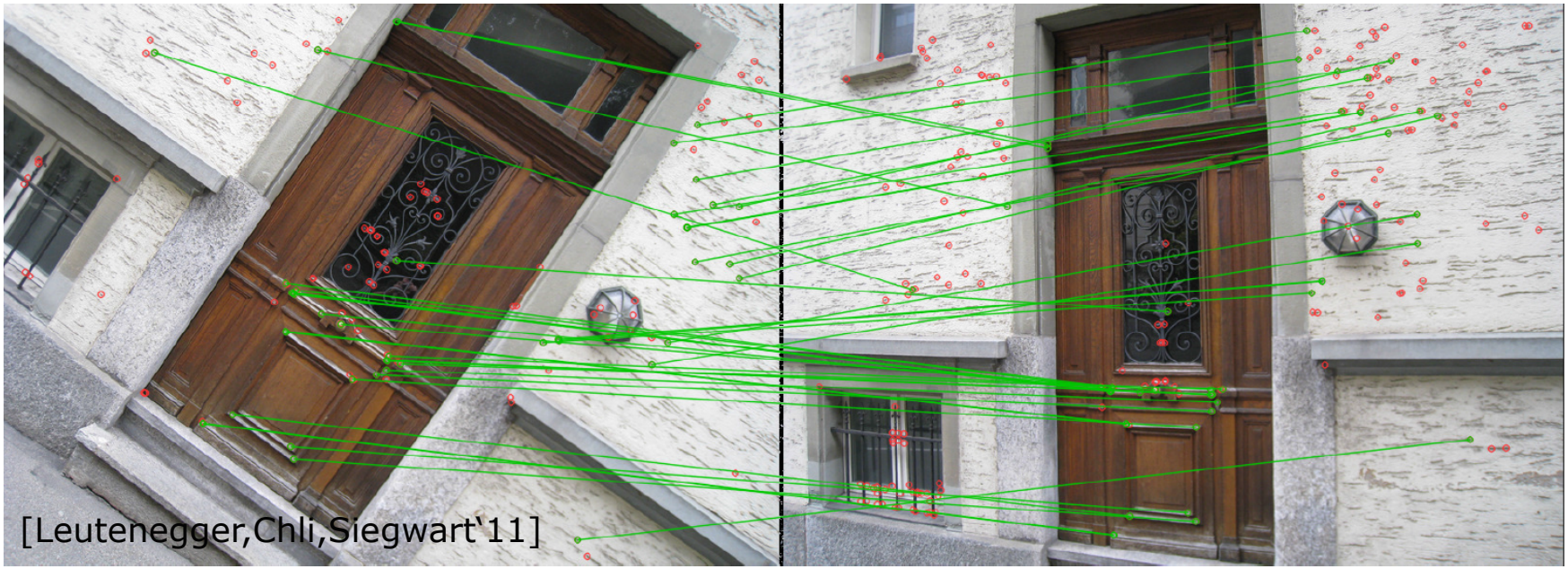
- Spatial relative pose constraints
- Information on structure

IDS uEye

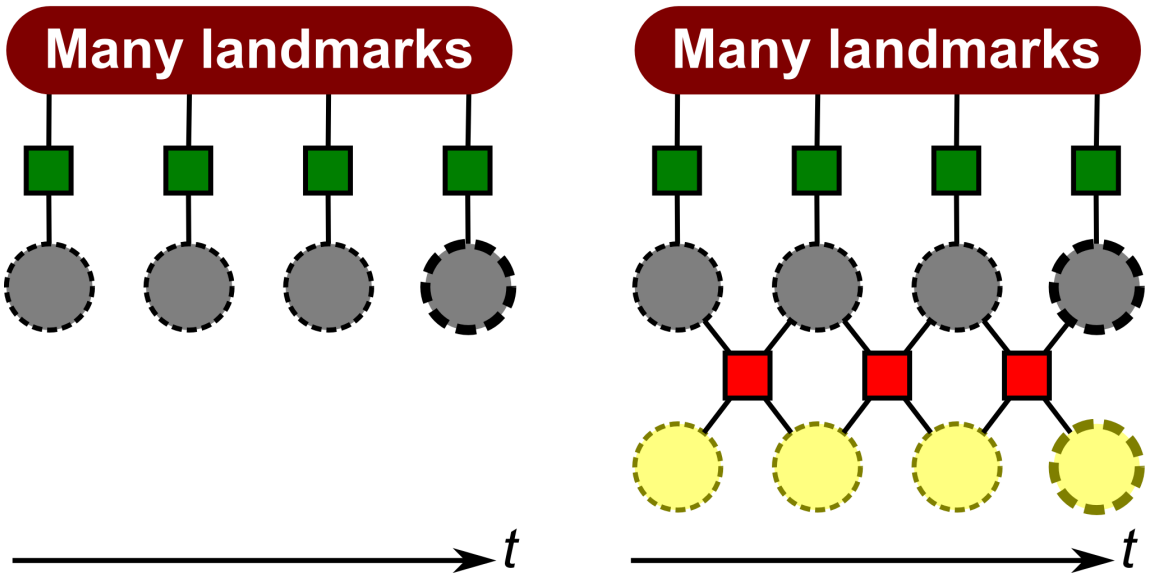
+

ADIS166448

- Strong short-term temporal pose constraints



Vision-Only vs. Visual-Inertial in Nonlinear Optimisation



- Pose
- Speed / IMU bias'
- Many keypoint measurements
- IMU measurements

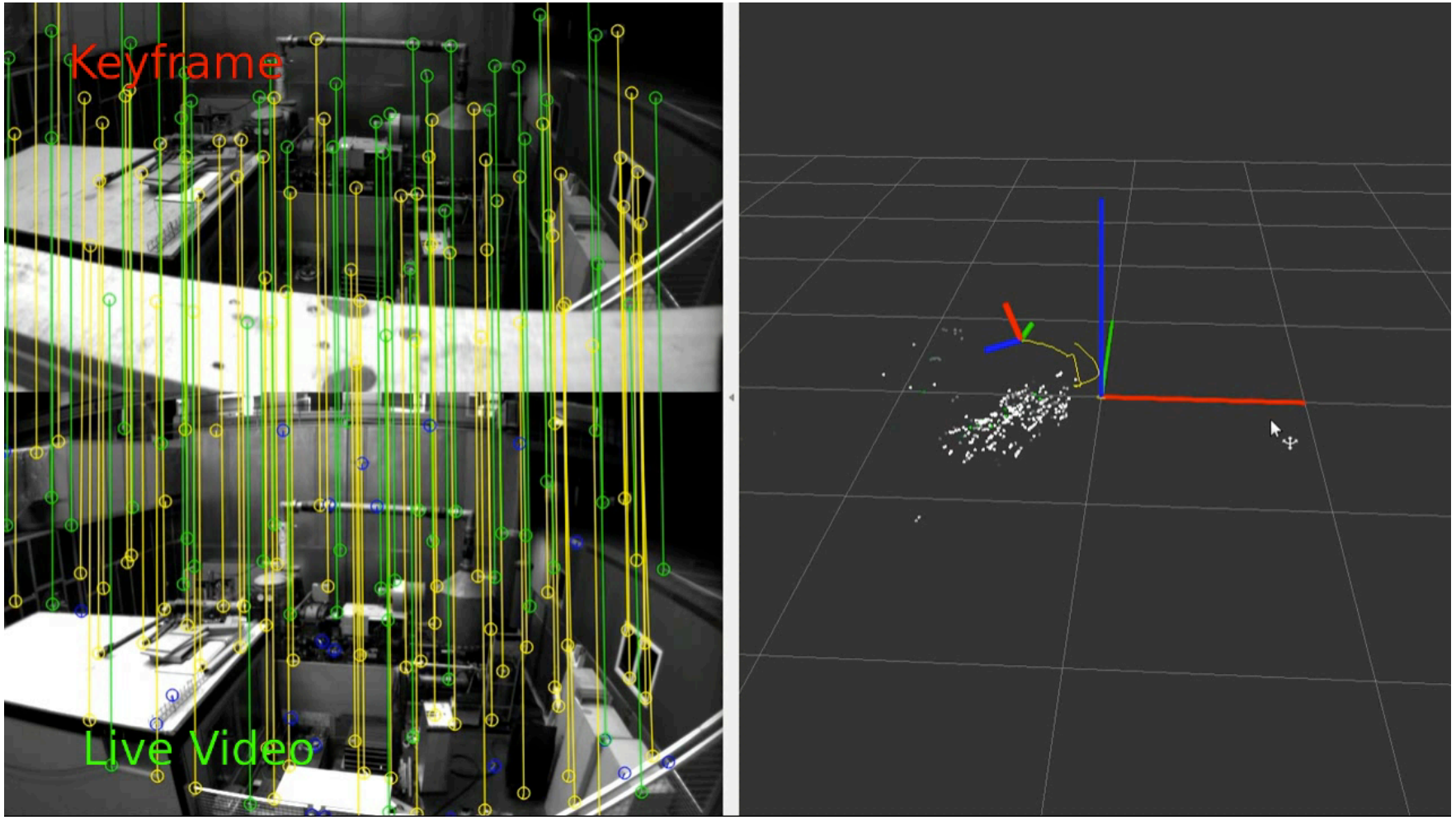
$$J(\mathbf{x}) := \sum_{i=1}^{I} \sum_{k=1}^{K} \sum_{j \in \mathcal{J}(i,k)} \mathbf{e}_{\mathrm{r}}^{i,j,k\,T} \mathbf{W}_{\mathrm{r}}^{i,j,k} \mathbf{e}_{\mathrm{r}}^{i,j,k} + \sum_{k=1}^{K-1} \mathbf{e}_{\mathrm{s}}^{k\,T} \mathbf{W}_{\mathrm{s}}^{k} \mathbf{e}_{\mathrm{s}}^{k}.$$

Cost

Reprojection errors:
Diff. between detected 2D keypoints and projected 3D landmarks

IMU terms:
Using the IMU kinematics model

OKVIS: Open Keyframe-based Visual-Inertial SLAM [IJRR'15]



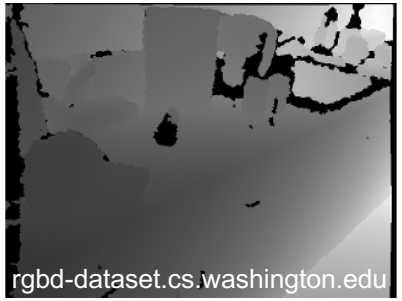
https://www.youtube.com/watch?v=TbKEPA2_-m4

OKVIS for Model-Predictive Control of MAV

Joint work with Dimos Tzoumanikas

Dense SLAM Overview

Depth-Cameras



Typical SLAM algorithm:

1. **Track**: align RGB (and D) image with map to compute 6D pose.
2. **Map**: fuse new RGB and D data into map.

Mono Camara

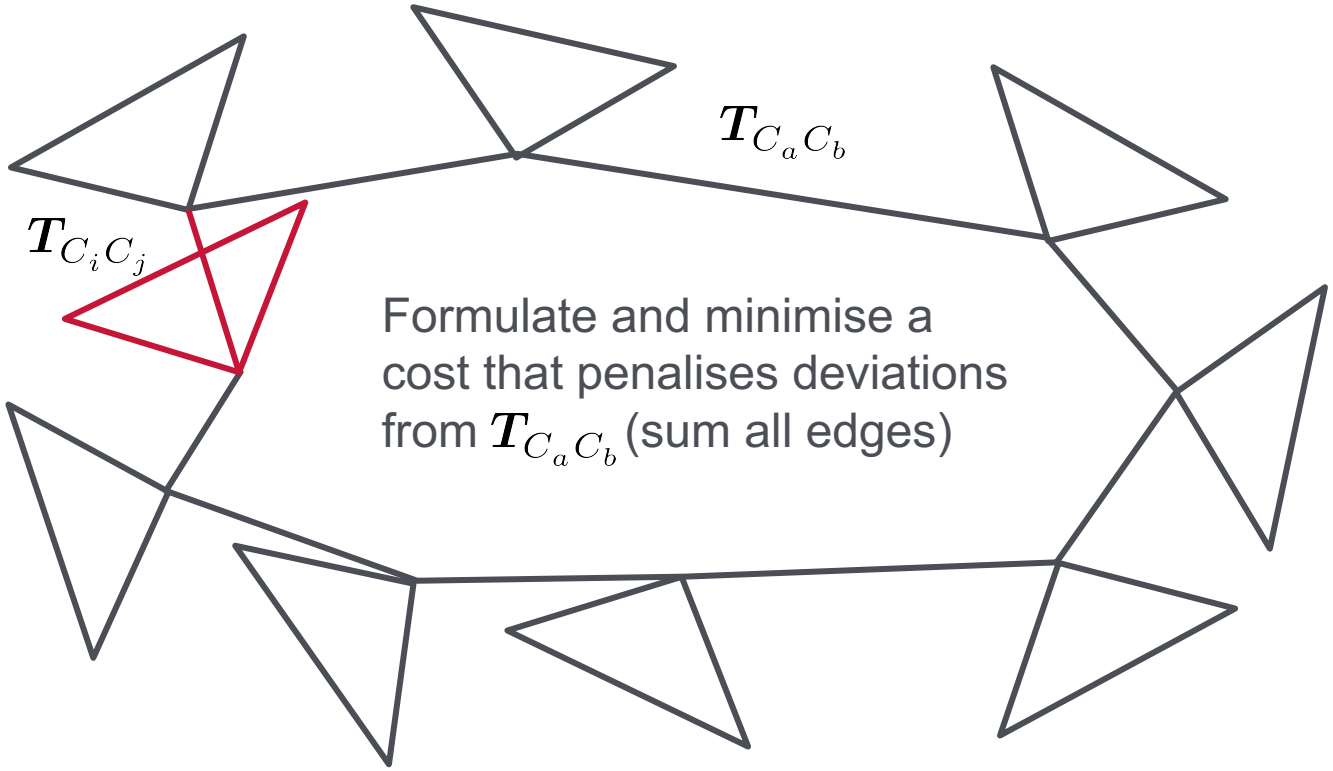
Typical SLAM algorithm:

1. **Track**: align image with map to compute 6D pose.
2. **Map**: fuse new data.
   Problem: we don't know depth…
   Options:
   - Compute depth maps from successive poses and then do the same as in RGB-D SLAM
   - Optimise the existing map directly to match the image(s)

Pose Graph Optimisation

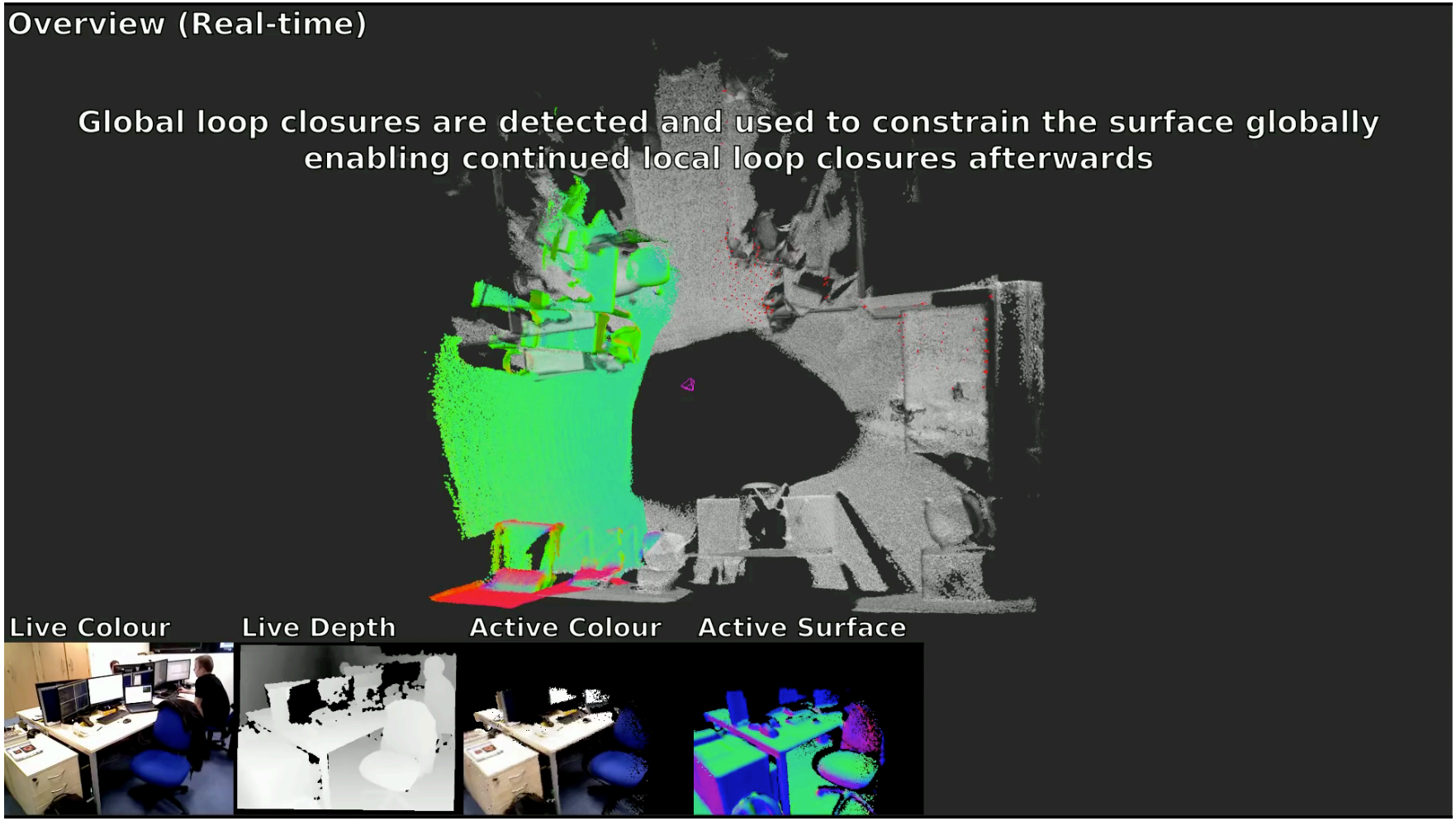
If trajectory / map needs to stay consistent after loop closure: **bend it into place**.

Example: Pose Graph Optimisation (Relaxation)



ElasticFusion [RSS'15, IJRR16]

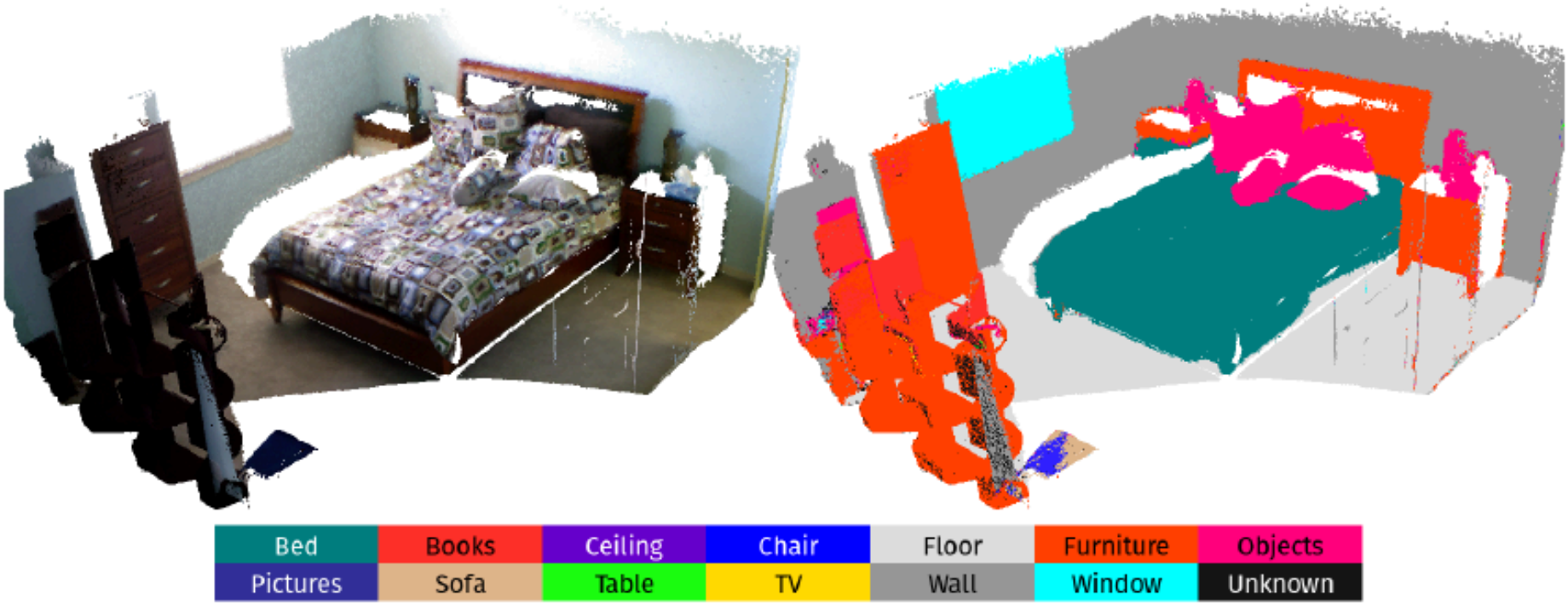
Joint work with Tom Whelan, Renato Salas-Moreno, Ben Glocker and Andy Davison



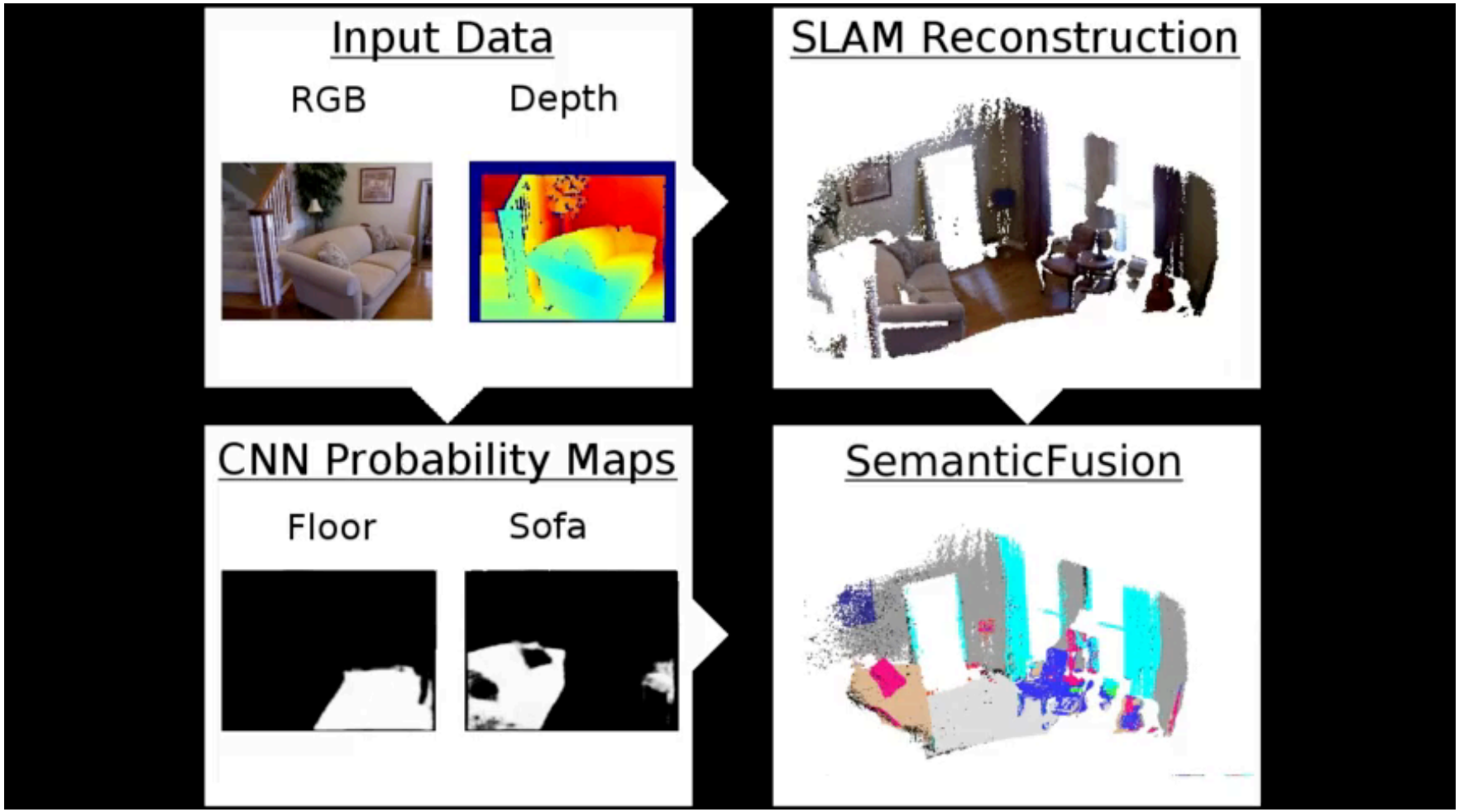
https://www.youtube.com/watch?v=XySrhZpODYs

From ElasticFusion to SemanticFusion [arXiv'16, ICRA'17]

Joint work with John McCormac, Ankur Handa, and Andrew Davison

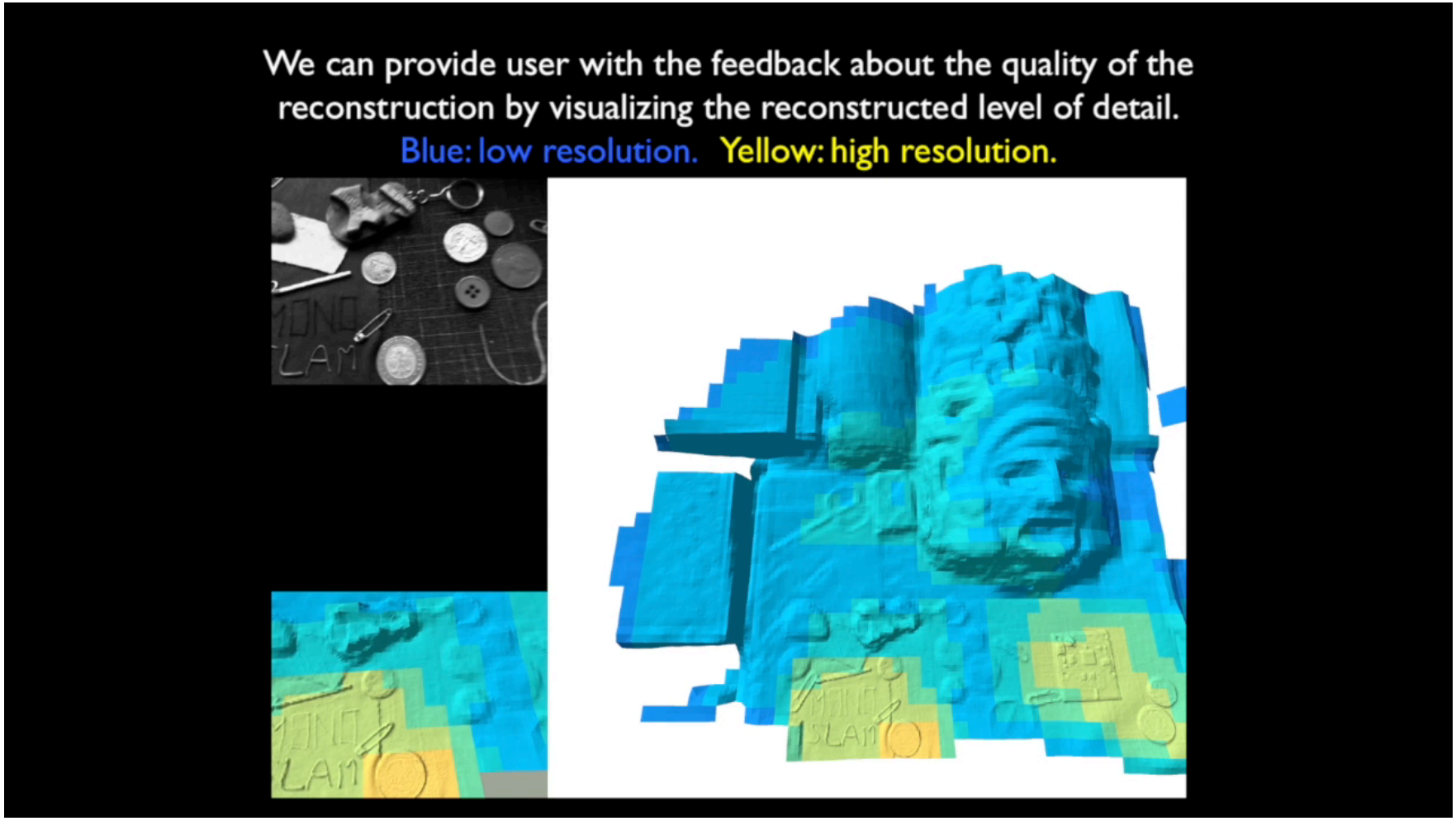


Fusing Semantics From CNN [arXiv'16]



https://www.youtube.com/watch?v=cGuoyNY54kU

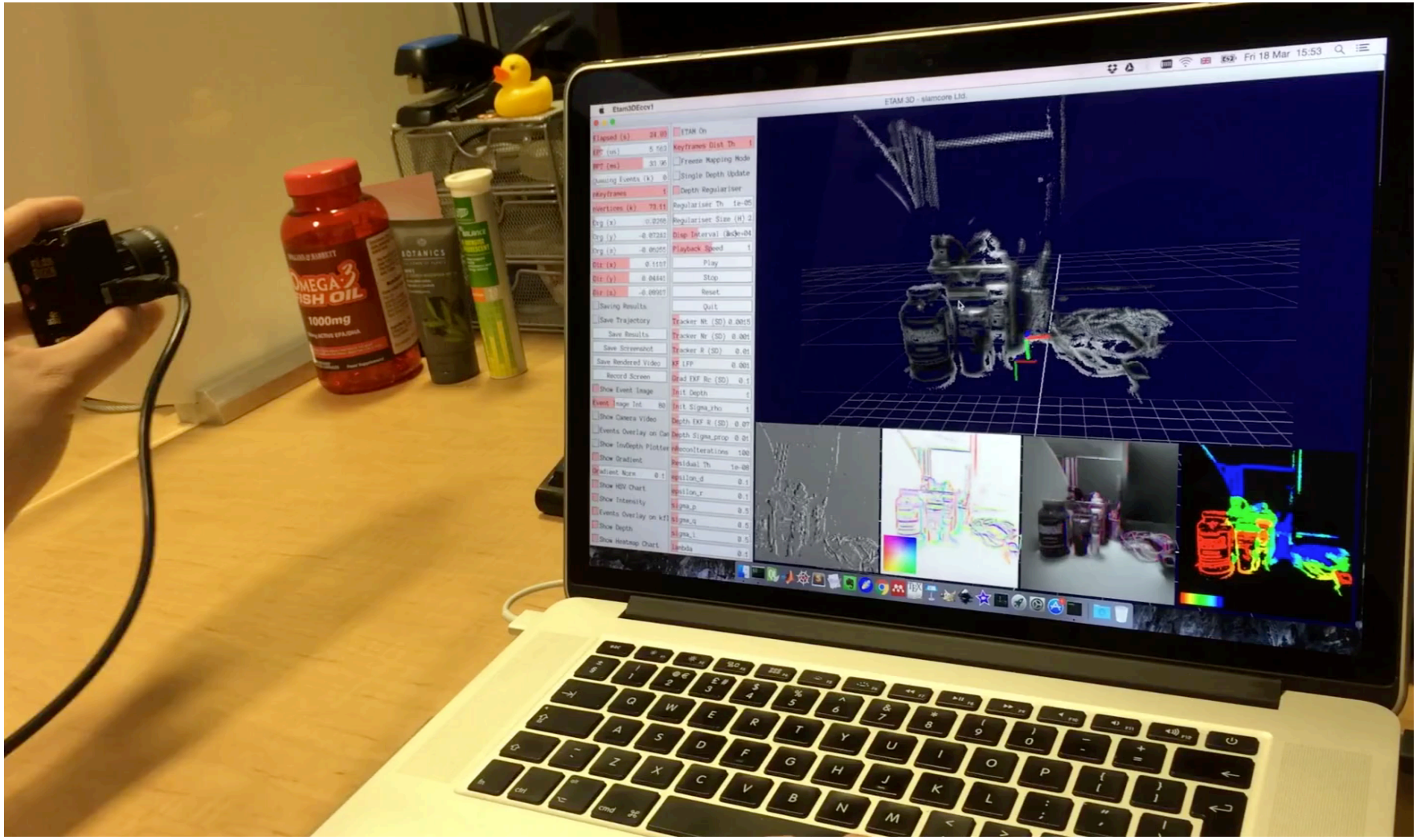
Multi-Level-of-Detail Reconstruction [3DV'16]



https://www.youtube.com/watch?v=UB_HDJU6LL4

Event Camera SLAM [ECCV'16]

Joint work with Hanme Kim and Andrew Davison



https://www.youtube.com/watch?v=yHLyhdMSw7w