Monocular SLAM and beyond

Autonomous Mobile Robots

Margarita Chli – University of Edinburgh

Paul Furgale, Marco Hutter, Martin Rufli, Davide Scaramuzza, Roland Siegwart

Monocular SLAM

Vision
for SLAM

- Images = information-rich snapshots of a scene
- Compactness + affordability of cameras
- HW advances

SLAM using a single, handheld camera:

- Hard but … (e.g. cannot recover depth from 1 image)
- **very** applicable, compact, affordable, …

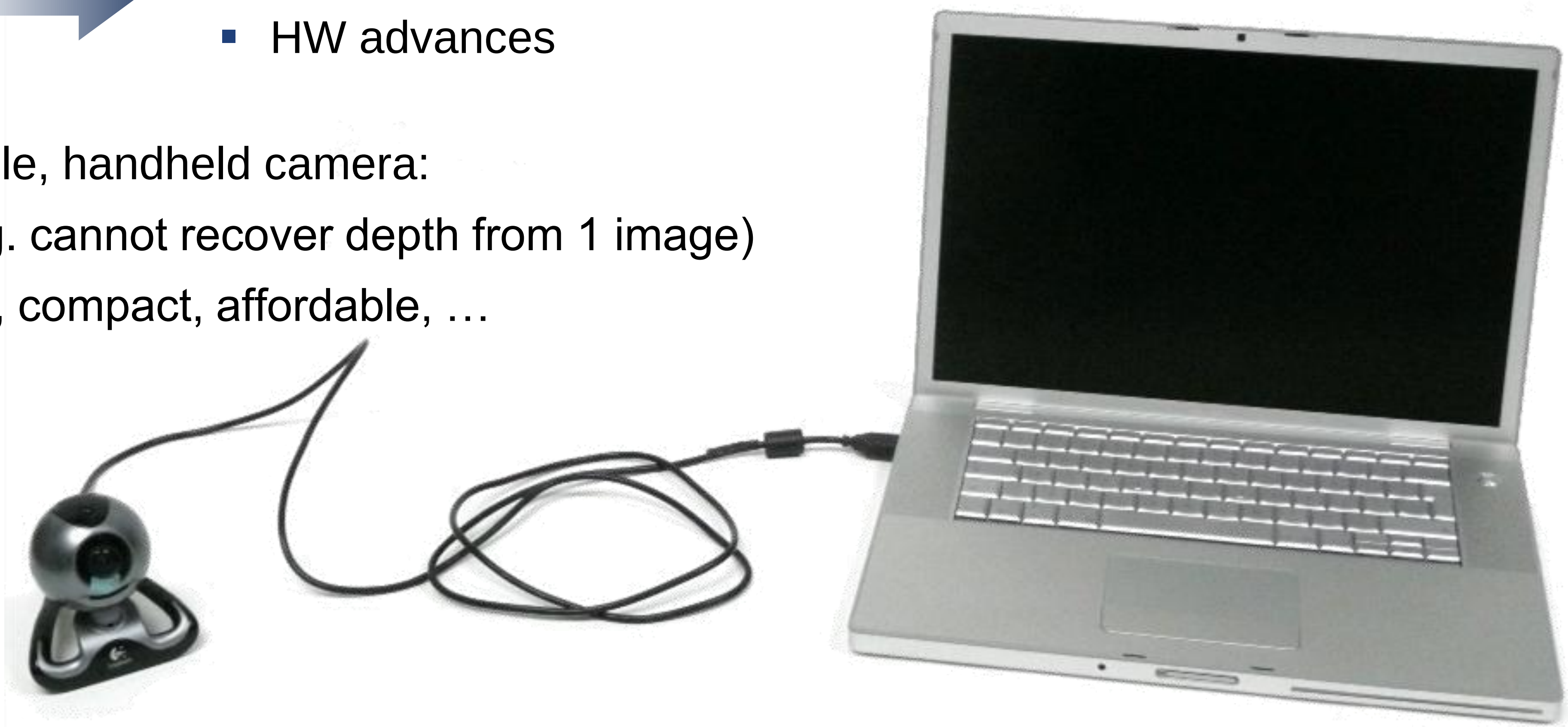


Image Courtesy of G. Klein

Monocular SLAM | from SFM to SLAM

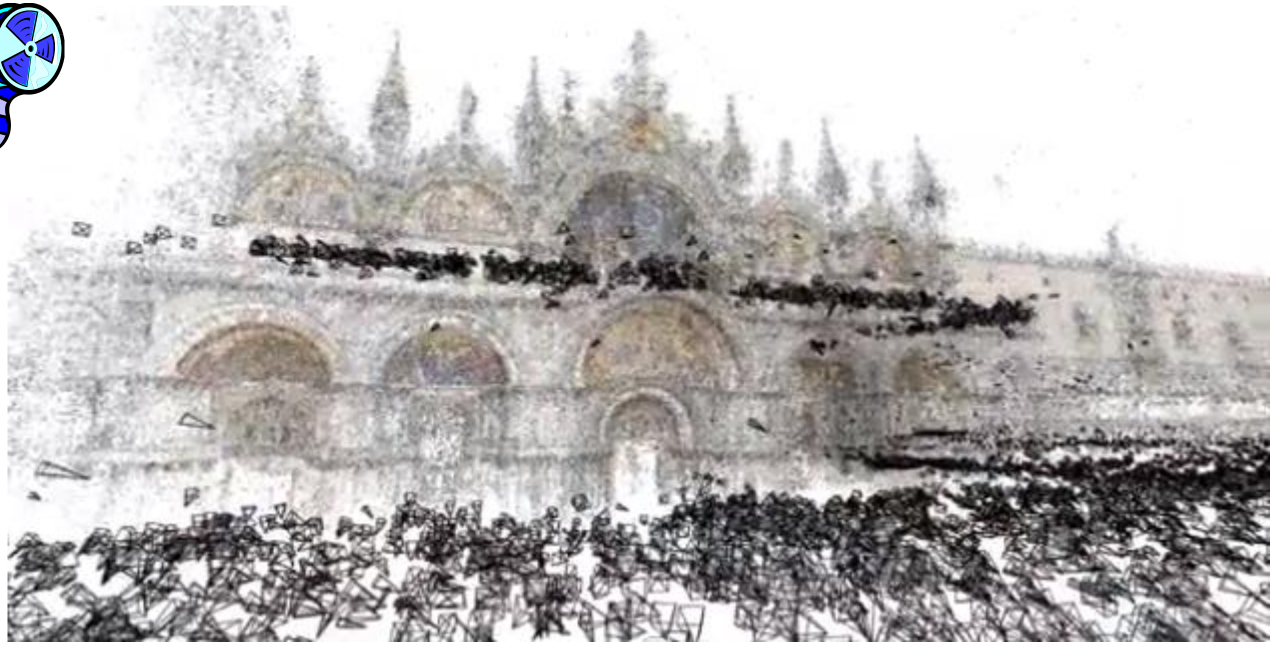
Structure from Motion (SFM):

- Take some images of the object/scene to reconstruct
- Extract features (points, lines, …) from all images and match them
- Process all images simultaneously
- Optimization to recover both:
    - camera motion and
    - 3D structure

  up to a scale factor
- **Not real-time, unordered images**

[Seitz, Szeliski ICCV 2009]

San Marco square, Venice
14’079 images, 4’515’157 points

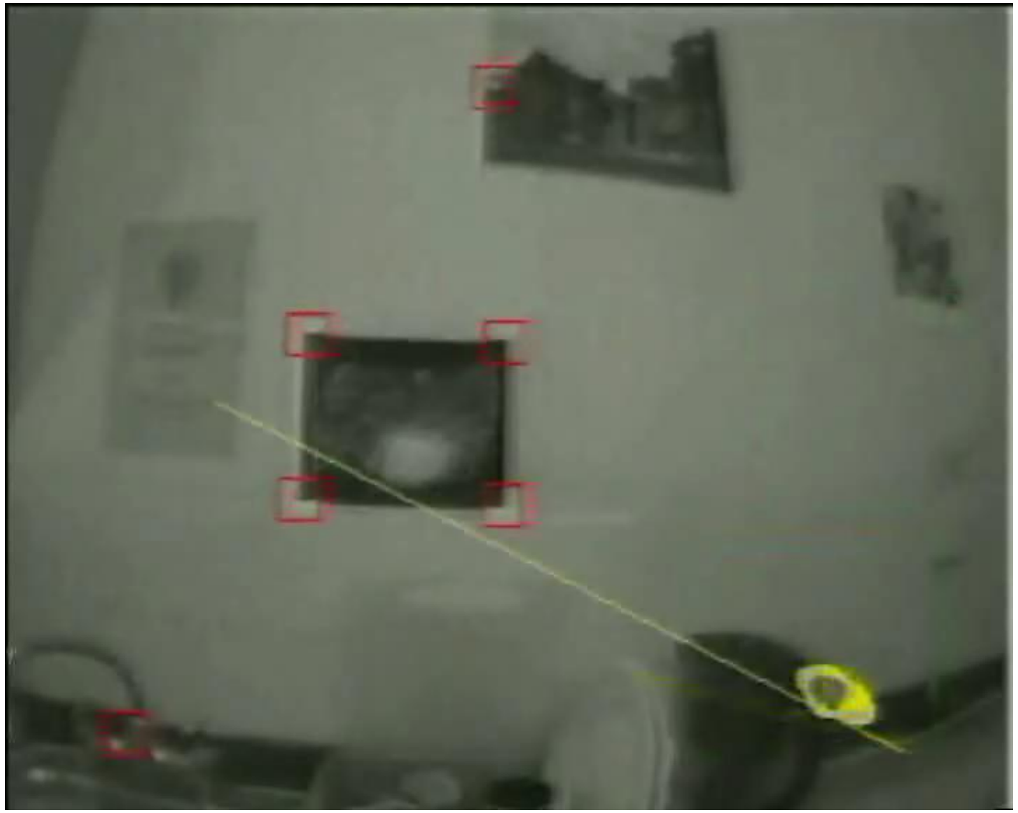


MonoSLAM [Davison et al., PAMI 2007]

- Can we track the motion of a **hand-held** camera while it is moving? i.e. **online**

Videos courtesy of Andrew J. Davison

scene view



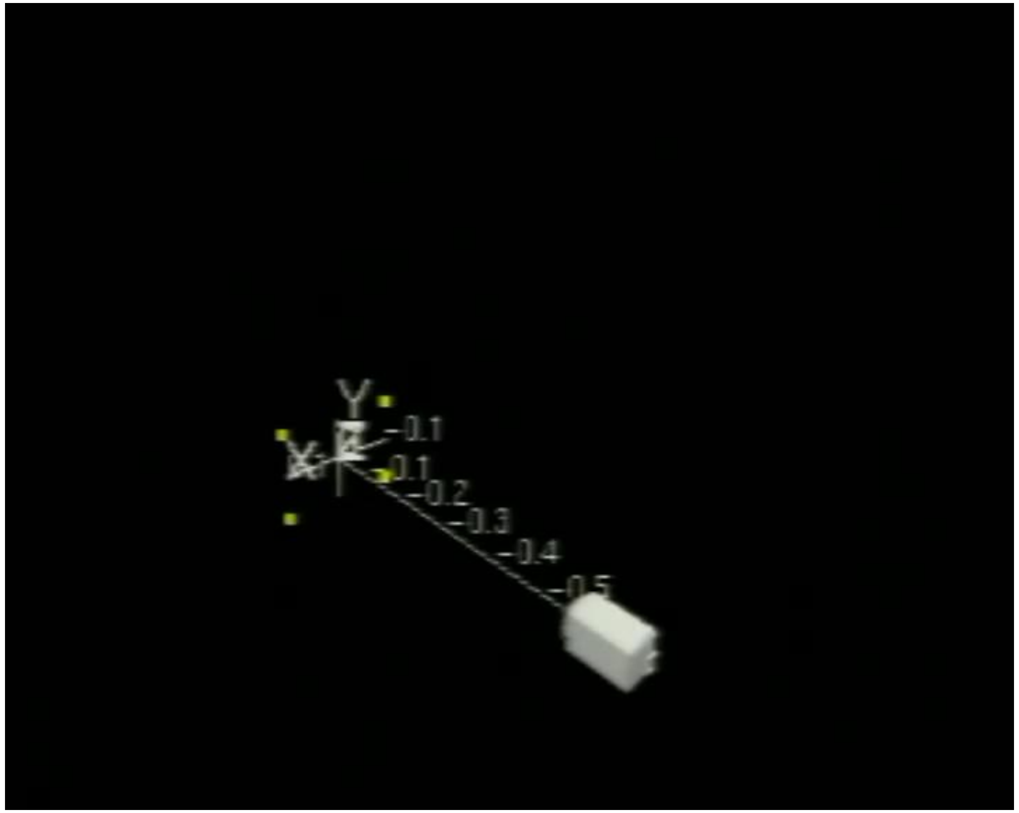
camera view

MonoSLAM [Davison et al., PAMI 2007]

- EKF SLAM using a single camera, grabbing frames at 30Hz
- Ellipses (in camera view) and ellipsoids (in map view) represent uncertainty

Videos courtesy of Andrew J. Davison

scene view



internal SLAM map

MonoSLAM | representation of the world

- The belief about the state of the world $\mathbf{x}$ is approximated with a single, multivariate Gaussian distribution:
- Using the notation of Davison et al. [PAMI 2007]:

$$p(\mathbf{x}) = (2\pi)^{-\frac{d}{2}} |\mathrm{P}|^{-\frac{1}{2}} \, exp\{-\frac{1}{2}(\mathbf{x}-\hat{\mathbf{x}})^\top \mathrm{P}^{-1}(\mathbf{x}-\hat{\mathbf{x}})\}$$

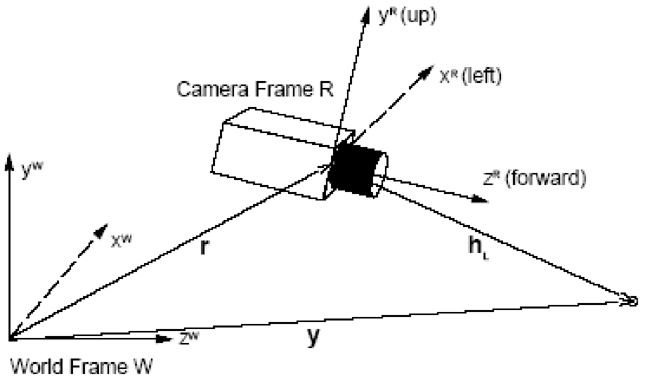
d denotes the dimension of $\hat{\mathbf{x}}$ and P is a square $(d \times d)$ matrix

Landmark's state
e.g. 3D position for point-features

Mean
(state vector)

$$\hat{\mathbf{x}} = \begin{pmatrix} \hat{\mathbf{x}}_c \\ \hat{\mathbf{y}}_1 \\ \hat{\mathbf{y}}_2 \\ \vdots \end{pmatrix}, \quad \mathrm{P} = \begin{bmatrix} \mathrm{P}_{xx} & \mathrm{P}_{xy_1} & \mathrm{P}_{xy_2} & \cdots \\ \mathrm{P}_{y_1x} & \mathrm{P}_{y_1y_1} & \mathrm{P}_{y_1y_2} & \cdots \\ \mathrm{P}_{y_2x} & \mathrm{P}_{y_2y_1} & \mathrm{P}_{y_2y_2} & \cdots \\ \vdots & \vdots & \vdots & \end{bmatrix}$$

Covariance
matrix



Camera state

$$\mathbf{x}_c = \begin{pmatrix} \mathbf{r}^W \\ \mathbf{q}^{CW} \\ \mathbf{v}^W \\ \omega^C \end{pmatrix}$$

: Position [3 dim.]
: Orientation using quaternions [4 dim.]
: Linear velocity [3 dim.]
: Angular velocity [3 dim.]

MonoSLAM | motion & probabilistic prediction

- How has the camera moved from frame *t-1* to frame *t*?

$$\hat{x}_t = f(x_{t-1}, u_t)$$

$$\hat{P}_t = F_x P_{t-1} {F_x}^T + F_u Q_t {F_u}^T$$

- The camera is **hand-held** ⇨ no odometry or control input data ⇨ Use a motion model
- But how can we model the unknown intentions of a human carrier?
- MonoSLAM uses a **constant velocity motion model**, imposing a certain smoothness on the expected camera motion
- *“on average we expect undetermined accelerations occur with a Gaussian profile”* [Davison et al., PAMI 2007]

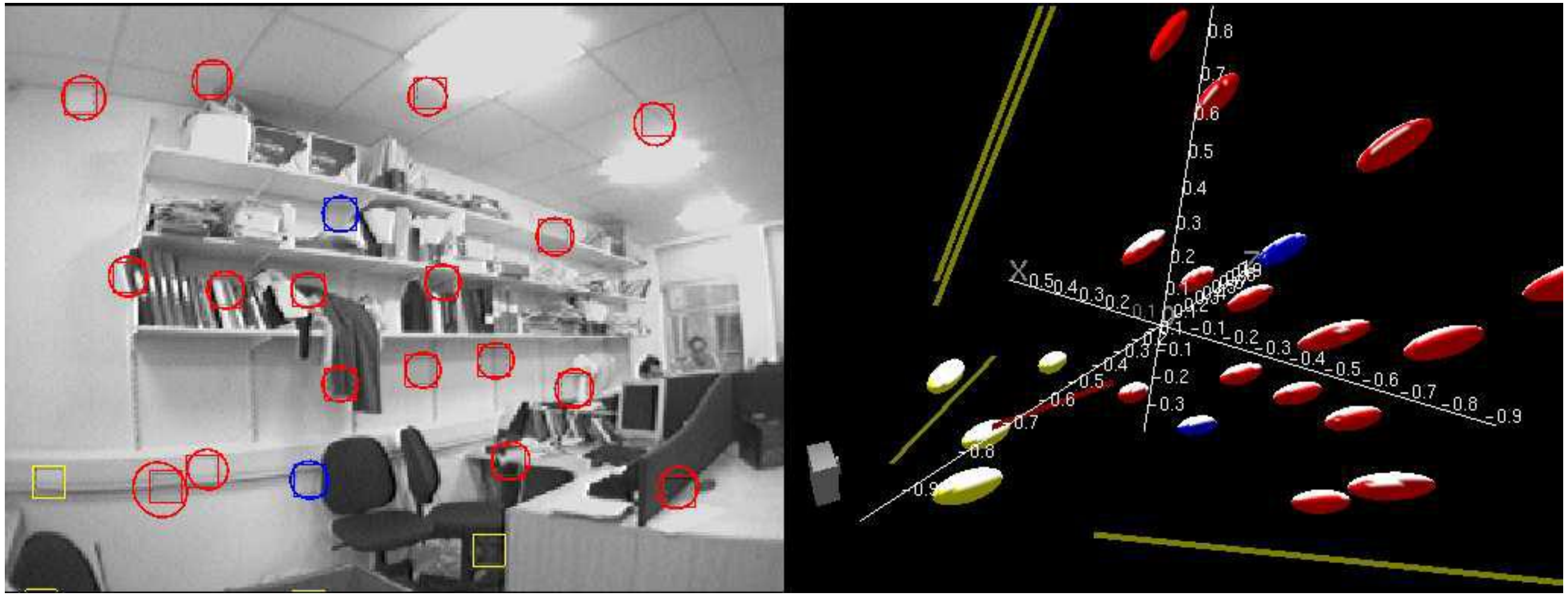
$$\mathbf{f}_v = \begin{pmatrix} \mathbf{r}^W_{new} \\ \mathbf{q}^{WR}_{new} \\ \mathbf{v}^W_{new} \\ \omega^W_{new} \end{pmatrix} = \begin{pmatrix} \mathbf{r}^W + (\mathbf{v}^W + \mathbf{V}^W)\Delta t \\ \mathbf{q}^{WR} \times \mathbf{q}((\omega^W + \mathbf{\Omega}^W)\Delta t) \\ \mathbf{v}^W + \mathbf{V}^W \\ \omega^W + \mathbf{\Omega}^W \end{pmatrix}$$

In each time step, the unknown angular & linear accelerations cause an impulse in velocity:

$$\mathbf{n} = \begin{pmatrix} \mathbf{V}^W \\ \mathbf{\Omega}^W \end{pmatrix} = \begin{pmatrix} \mathbf{a}^W \Delta t \\ \alpha^W \Delta t \end{pmatrix}$$

MonoSLAM | motion & probabilistic prediction

- Based on the predicted new camera pose
  ⇨ predict **which** known features will be visible and **where** they will lie in the image
- Use measurement model h to identify the predicted location $\hat{z}_i = h_i(\hat{x}_t, y_i)$ of each feature and an associated search region (defined in the corresponding diagonal block of $\Sigma_{IN} = H\hat{P}_t H^T + R$)
- Essentially: project the 3D ellipsoids in image space



MonoSLAM | measurement & EKF update

- Search for the known feature-patches inside their corresponding search regions to get the set of all observations

- Update the state using the EKF equations:

$$x_t = \hat{x}_t + K_t(z_{0:n-1} - h_{0:n-1}(\hat{x}_t, y_{0:n-1}))$$

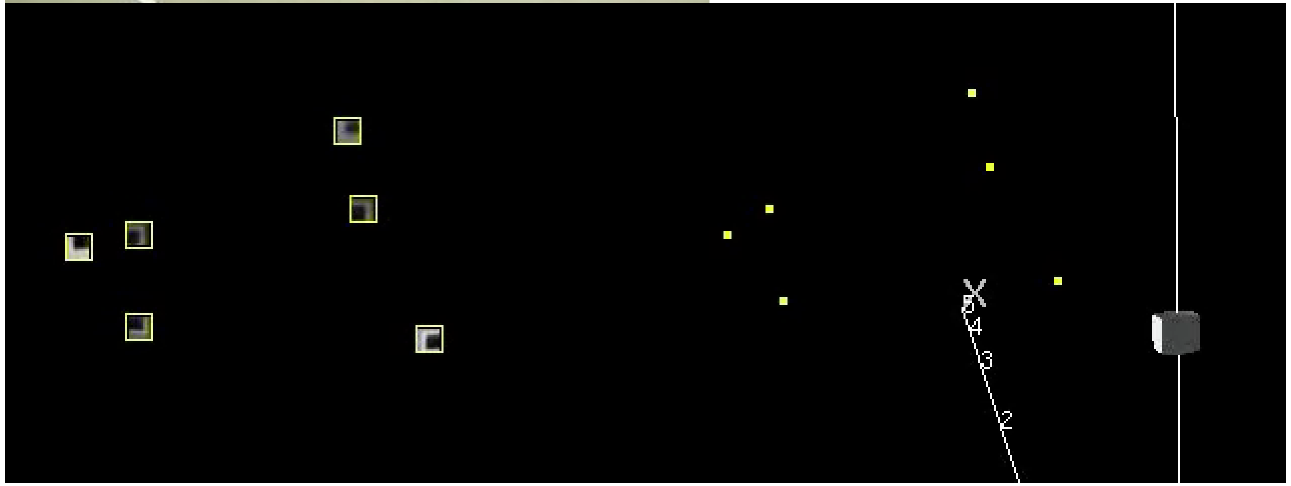
$$P_t = \hat{P}_t - K_t \Sigma_{IN} K_t^T$$

where:

$$\Sigma_{IN} = H\hat{P}_t H^T + R$$

$$K_t = \hat{P}_t H(\Sigma_{IN})^{-1}$$

MonoSLAM | MonoSLAM in action



- Small circular loop within a large room
- No re-observation of ‘old’ features until closing of large loop

Videos courtesy of Andrew J. Davison

EKF SLAM | correlations

- At start up: the robot makes the first measurements and the covariance matrix is populated assuming that these (**initial**) features are **uncorrelated**
  ⇨ off-diagonal elements are zero.

$$P_{y_0} = \begin{bmatrix} P_{xx} & 0 & 0 & \dots & 0 & 0 \\ 0 & P_{m_0 m_0} & 0 & \dots & 0 & 0 \\ 0 & 0 & P_{m_1 m_1} & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & P_{m_{n-2} m_{n-2}} & 0 \\ 0 & 0 & 0 & \dots & 0 & P_{m_{n-1} m_{n-1}} \end{bmatrix}$$

- When the robot starts moving & taking new measurements, both the robot pose and features start becoming correlated.

$$\hat{P}_{y_t} = F_y P_{y_{t-1}} F_y^{\,T} + F_u Q_t F_u^{\,T}$$

- Soon the covariance matrix becomes **dense**…

EKF SLAM | correlations

- Correlations arise as
  - the uncertainty in the robot pose is used to obtain the uncertainty of the observed features.
  - the feature measurements are used to update the robot pose.

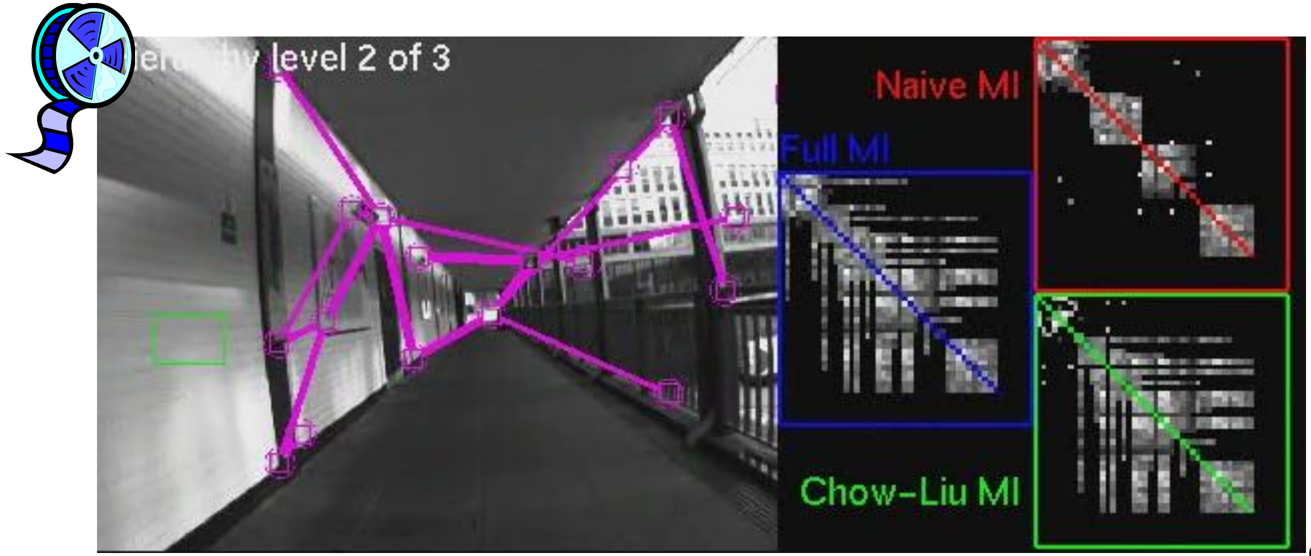
- **Regularly covisible** features become correlated
- When their motion is **coherent** their correlation is even stronger

- Correlations very important for **convergence**:
  The more observations are made, the more the correlations between the features will grow, the better the solution to SLAM.

Chli & Davison, ICRA 2009

EKF SLAM | drawbacks

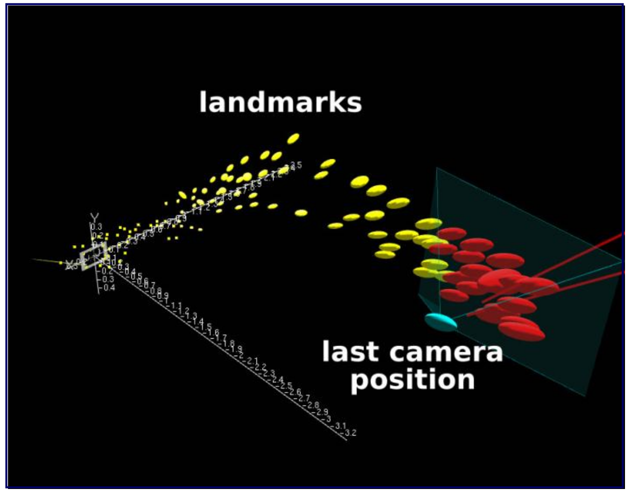
- The state vector in EKF SLAM is much larger than the state vector in EKF localization
- Newly observed features are added to the state vector ⇨ The covariance matrix **grows quadratically** with the no. features ⇨ **computationally expensive for large-scale SLAM**.
- Approach to attack this: sparsify the structure of the covariance matrix (via approximations)



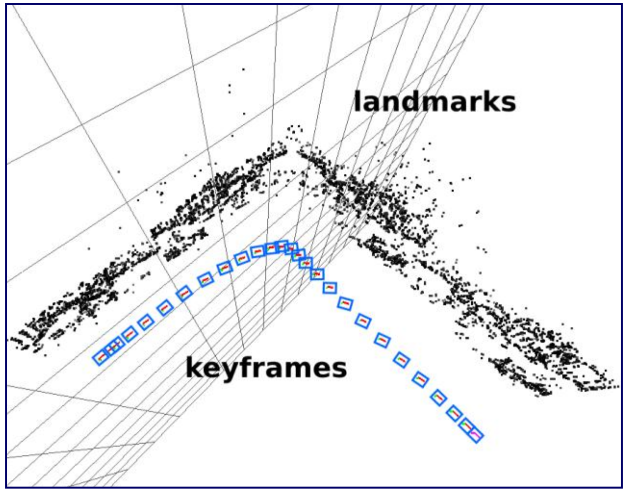
Chli & Davison, ICRA 2009

SLAM challenges | real-time single camera SLAM systems

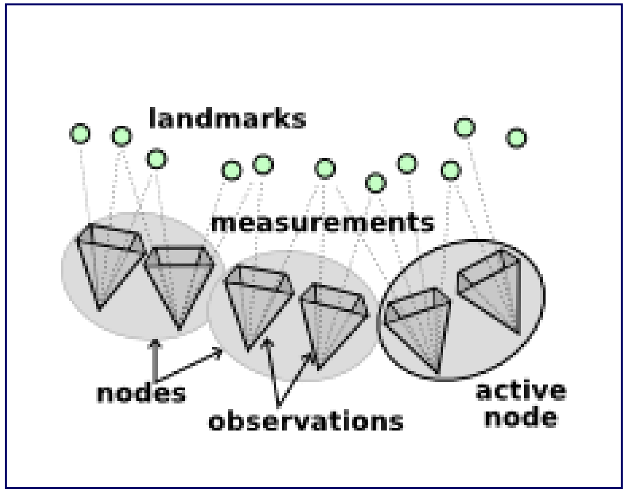
MonoSLAM is computationally expensive with increasing no. features



MonoSLAM
[Davison et al. 2007]



PTAM
[Klein, Murray 2008]



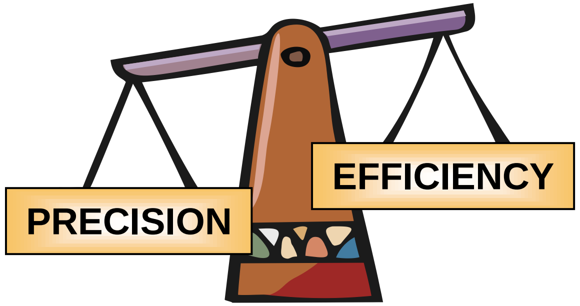
Graph-SLAM
[Eade, Drummond 2007]

SLAM challenges

- Faster motion
- Larger scales
- Robustness
- Richer maps
- Low computation for embedded apps

High-Speed Gaze Controller for Millisecond-order Pan/tilt Camera [Okumura, Oku and Ishikaw, ICRA 2011]

- Handle **larger** amounts of data more **effectively**
- Competing goals:

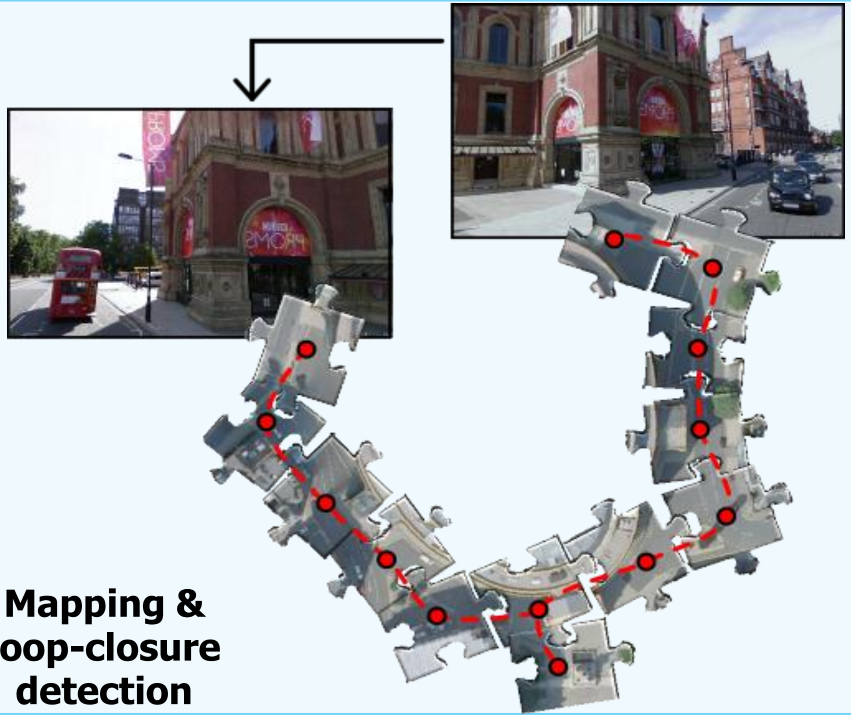


SLAM Challenges | components for scalable SLAM

1

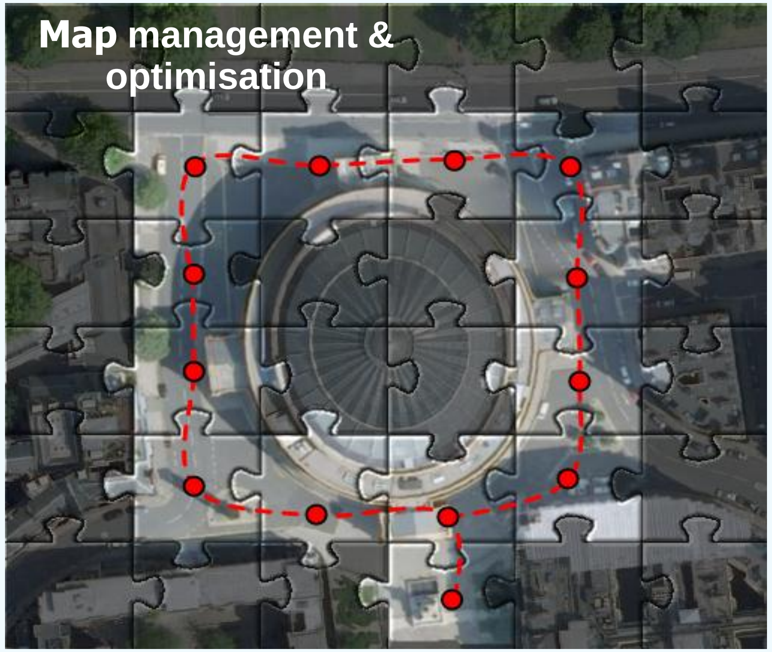
Robust local motion estimation

2



Mapping & loop-closure detection

3



[Chli,2009, PhD thesis]

SLAM today | vision-based SLAM for MAVs

- Visual-inertial SLAM onboard a Micro Aerial Vehicle

[Achtelik et al., IROS 2012]

SLAM today | DTAM: Dense Tracking And Mapping

[Newcombe, Davison ICCV 2011]