
DTU

Perception for Autonomous Systems 34579:

Visual Odometry

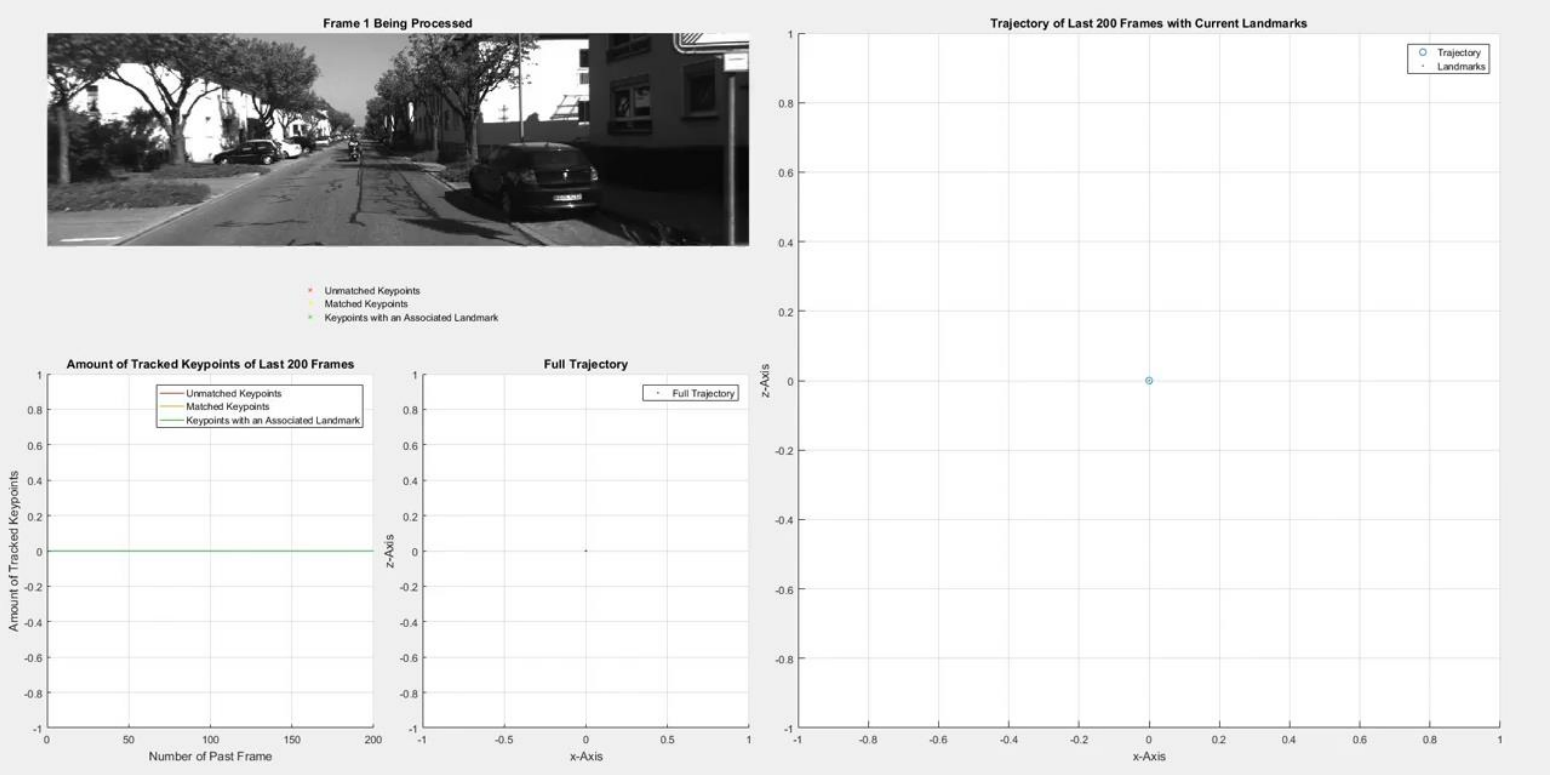
Lecturer: Evangelos Boukas—PhD

Outline

- Orientation (Attitude) Representations
- Relative Pose Estimation
  - 3D registration
  - PnP
  - Least Squares - SVD
- Visual Odometry
  - 3D-3D
  - 3D-2D
  - 2D-2D
- Local Bundle Adjustment
- Visual Inertial Odometry -VIO
  - Loosely Coupled EKF
  - Tightly Coupled EKF

What is Visual Odometry and what it is not

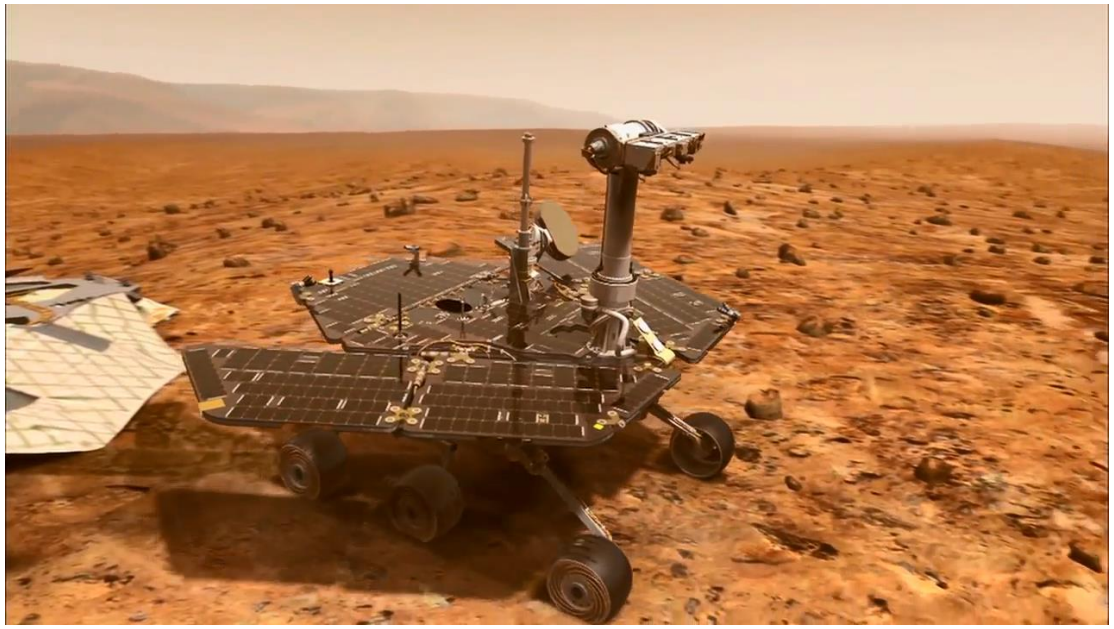
- Visual Odometry (VO) concerns the use of cameras to estimate the Pose (position and orientation) of a mobile system, by observing the apparent motion of the “static” world.
- VO assumes a static world where the only moving object is the mobile system
- VO does provide a map of the environment neither it uses previous states of the world to improve it’s accuracy (SLAM)

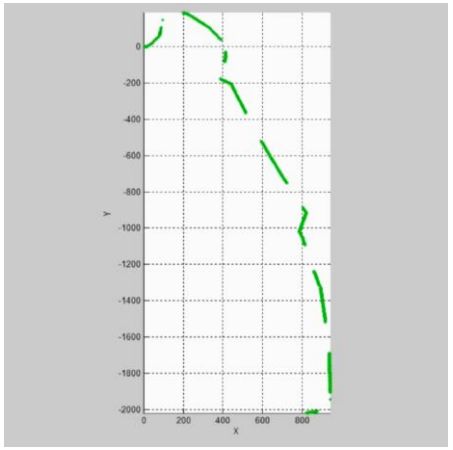


Video by prof. Scaramuzza University of Zurich

VO Facts

- It has been proposed as an alternative to Wheel odometry
- Its accuracy is –assuming reasonable trajectories- ~1%
- VO estimations are usually combined with other sensors
  - GPS, IMU, Laser, Wheel odometry
  - VINS, VIO stands for Visual Inertial Odometry
- VO has had some extraordinary usages



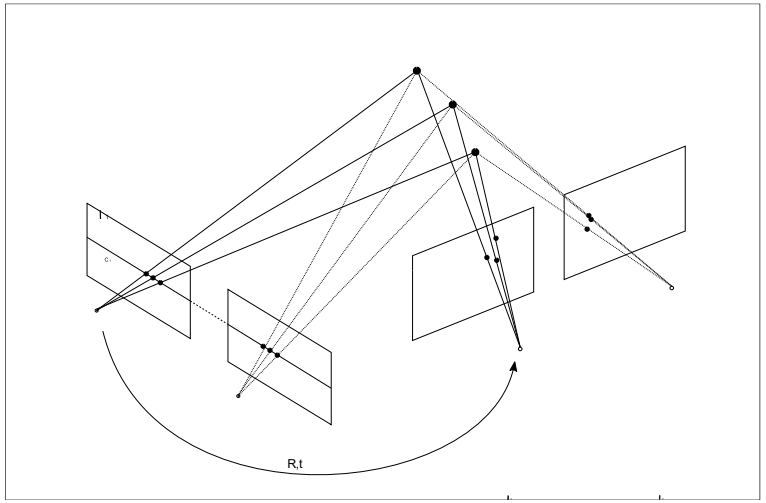


VO is neither SFM nor VSLAM

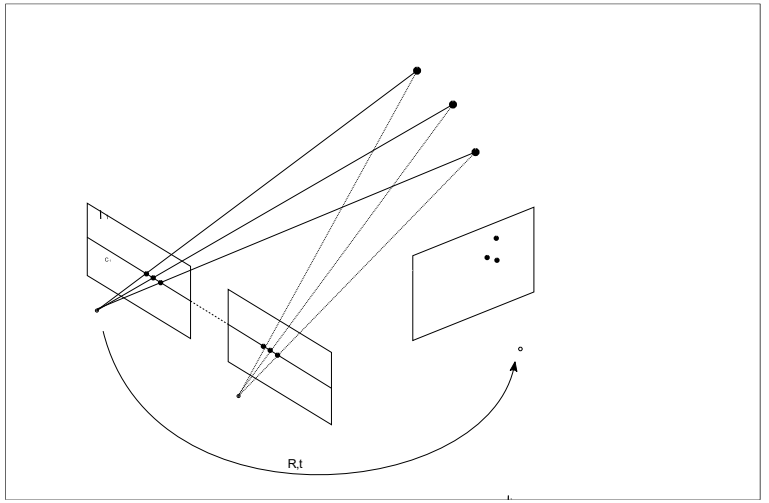
- Structure from motion tries to solve also for the feature points:
  - *"Given Calibrated point projections of p=1…N points in camera (or frame) f= 1 ... F* $\left(x_p^f, y_p^f\right)$
  - *Find the rigid transformation* $R^T t$ *and the point's 3D position* $\boldsymbol{X}_p = \left(X_p\ , Y_p\ , Z_p\ \right)$ *which satisfies the projection equations*

- Visual SLAM uses state estimation to exploit
  additional constrains and re-observations of the same areas(Loop-closures)
  to optimize the localization

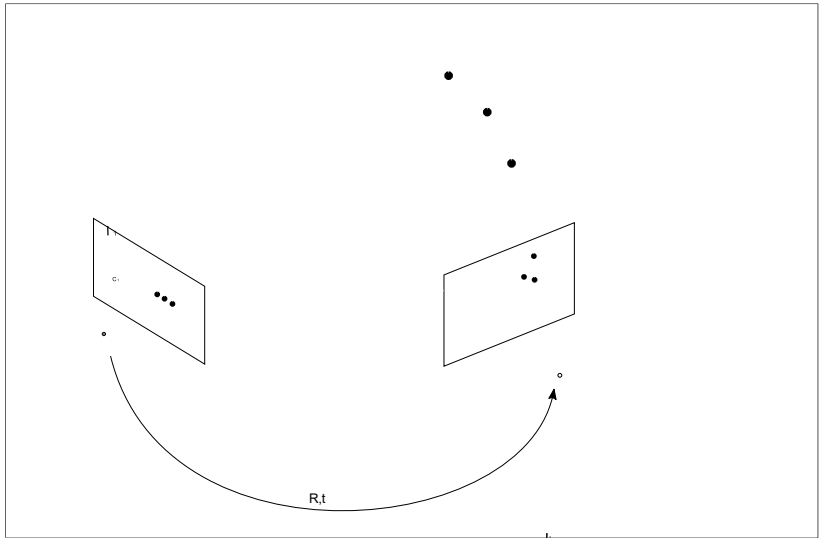
The 3 main variants of VO



3D-3D



3D-2D



2D-2D

Orientation

- Rotation Matrix
  - Pros (The king of Orientation)
    - Unique, no gimbal lock
  - Cons:
    - No perturbation, interpolation, unintuitive
- Euler anglers
  - Pros
    - Minimal representation, intuitive
  - Cons
    - Gimbal Lock, non commutative
- Axis Angle:
  - Pros
    - No gimbal lock, minimal representation, nice for perturbation, linear mapping to rotation matrix
  - Cons
    - Not linear “scaling” wrt magnitude
  - Exponential coordinates
- Quaternions
  - Pros
    - all the axis angle ones, smooth trajectory
  - Cons
    - No direct geometric representation

Shuster, M.D. (1993). "A Survey of Attitude Representations". *Journal of the Astronautical Sciences* **41** (4): 439–517. Bibcode: 1993JAnSc..41..439S

In-Image Motion tracking

- Two main approaches
  - Feature based
  - Optic Flow

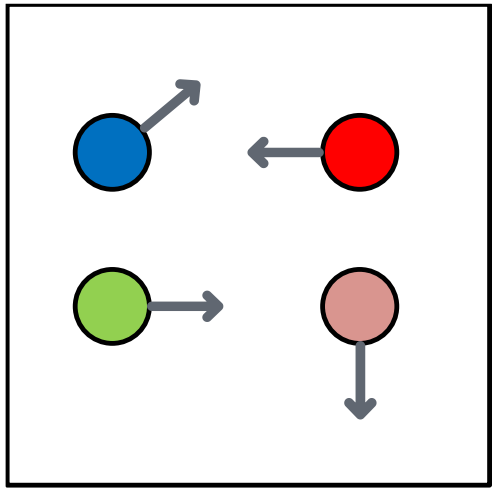
  - Feature based:
    - Calculate feature on both images
    - Match among the features

    or
    - Calculate features
    - Do block Matching around our initial point (for small motion)

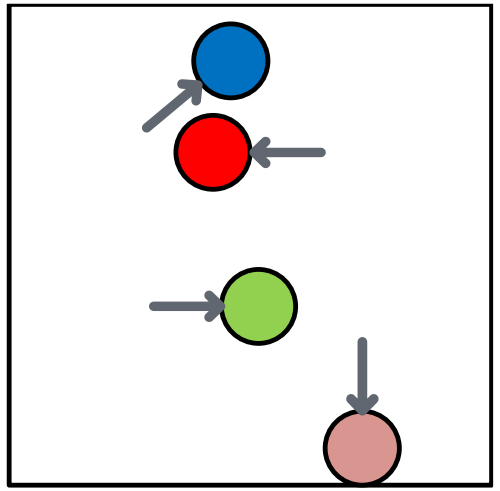
Motion tracking – Optical Flow

- Estimate the apparent motion

- Given a pixel in location I(x,y,t), find the “nearby pixels with the same color“
  - Same intensity (in the local window)
  - Limited displacement



I(X,Y,t)

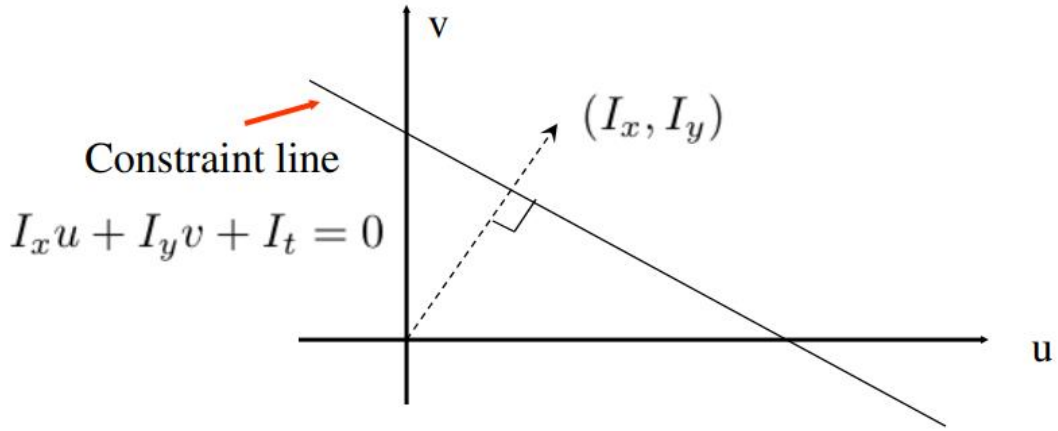


I(X+dX, Y+dY, t+dt)

- $I(x, y, t) = I(x + u, y + v, t + 1)$
- $I(x + u, y + v) \approx I(x, y) + \frac{\partial I}{dx} u + \frac{\partial I}{dy} \mathrm{v} \quad \rightarrow \quad I_t + \nabla I \cdot (u, v) = 0 \quad \rightarrow I_x \mathrm{u} + I_y \mathrm{v} + I_t = 0$
- This problem is called “global Optical Flow” and is hard to solve due to *under definition*

Motion tracking – Optical Flow – Lukas Kanade

- The previous function is a line in the u,v space
- $I_x u + I_y v + I_t = 0$



- *We can try to impose more constrains so the line becomes a point*
- By assuming that for an image neighborhood we have constant “velocity”:
- We want to minimize:

$$E(u,v) = \sum_{x,y \in \Omega} \left(I_x(x,y)u + I_y(x,y)v + I_t \right)^2$$

Motion tracking – Optical Flow – Lukas Kanade

- If we use a 5x5 window,
  that gives us 25 equations per pixel

- This does not work
  - For edges
  - For large areas
- How to solve it:
  The iterative approach
  - Estimate Motion
  - Warp Image
  - Repeat until no change

$$E(u,v) = \sum_{x,y \in \Omega} \left(I_x(x,y)u + I_y(x,y)v + I_t \right)^2$$

$$0 = I_t(\mathbf{p_i}) + \nabla I(\mathbf{p_i}) \cdot [u \; v]$$

$$\begin{bmatrix} I_x(\mathbf{p_1}) & I_y(\mathbf{p_1}) \\ I_x(\mathbf{p_2}) & I_y(\mathbf{p_2}) \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}}) & I_y(\mathbf{p_{25}}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p_1}) \\ I_t(\mathbf{p_2}) \\ \vdots \\ I_t(\mathbf{p_{25}}) \end{bmatrix}$$

$$\underset{25\times2}{A} \qquad \underset{2\times1}{d} \qquad \underset{25\times1}{b}$$

Relative Pose estimation

- Depends on the available data:
  – 3D or 2D

Relative Pose estimation – 2 corresponding 3D Point Clouds

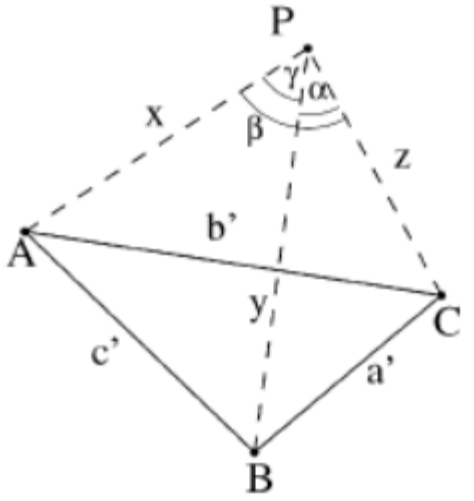
- 3D registration – case **where $f_k \; and f_{k-1}$ are in specified in 3D points**
  – PCA
  – SVD, RANSAC
  – ICP, combination of above

  What if we have correspondences?
  – Rigid Transformation using RANSAC (3 points)

Relative Pose estimation – 3D “Point clouds” and 2D Image Points

- The problem where f_{k-1} **is specified in 3D points and** f_k **in 2D image coordinates** - This problem is known as perspective from n points (PnP)

- A popular implementation is the P3P (perspective from 3 points)
  - Let P be the Center of Perspective
  - A, B, C, the “control points”, the 3D correspondence
  - Appling the cosine law (e.g.: $x^2+z^2- 2*x*z*\cos\beta =b'$), we get 2 quadratic equations with 2 unknowns, resulting to 4 possible solutions for R,t
  - Using in Ransac to find the correct, or employ a 4th point, check orientation consistency



- Many other implementations:
  - One of the most prominent is EPnP (n>=4)
    - Reformulate the problem with virtual “control points”

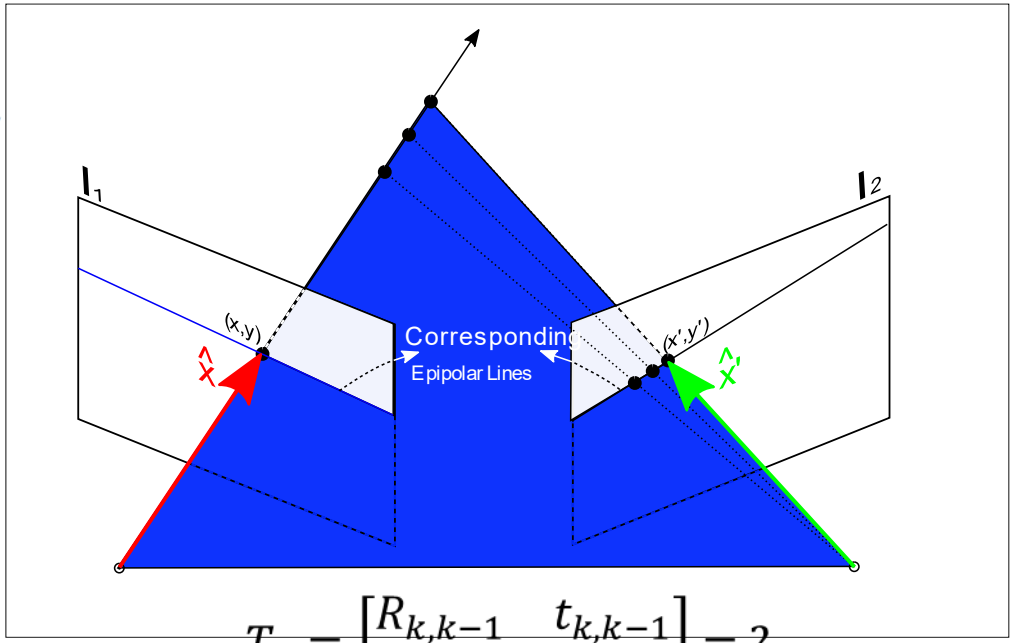
Relative Pose estimation - 2D Image Points

- The problem where $\boldsymbol{f_{k-1}}$ $\boldsymbol{and}$ $\boldsymbol{f_k}$ **are specified in 2D image coordinates**
- The minimal-case solution involves 5-point correspondences
- The solution is found by determining the transformation that minimizes the reprojection error of the corresponding points,

$$T_k = \begin{bmatrix} R_{k,k-1} & t_{k,k-1} \\ 0 & 1 \end{bmatrix} = \arg\min_{X^i, C_k} \sum_{i,k} \| p_k^i - g(X^i, C_k) \|^2$$

where p_k^i is the points on image k and $\mathrm{g}(X^i, C_K)$ the reprojection of the corresponding $k-1$ point on the camera k

Wait but WHY?



$$T_k = \begin{bmatrix} R_{k,k-1} & t_{k,k-1} \\ 0 & 1 \end{bmatrix} = ?$$

Relative Pose estimation - 2D Image Points

The Essential Matrix can be computed directly from the image coordinates (using SVD).

At least 5 points needed! The more points, the better!

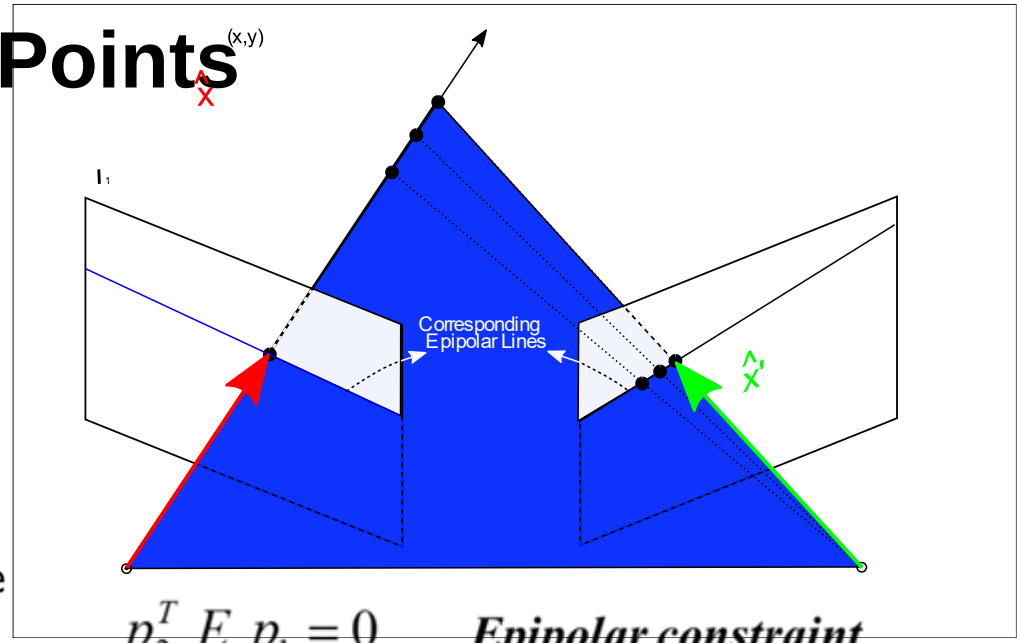
The Essential Matrix can be decomposed into $\boldsymbol{R}$ and $\boldsymbol{t}$ (again using SVD)

Let $p_1 = \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix}$, $p_2 = \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix}$ be the coordinates one feature correspondence

$$E = \begin{bmatrix} e_{11} & e_{12} & e_{13} \\ e_{21} & e_{22} & e_{23} \\ e_{31} & e_{32} & e_{33} \end{bmatrix} = \begin{bmatrix} e_{11} \\ \vdots \\ e_{33} \end{bmatrix}$$

$$p_2{}^{\mathrm{T}} E p_1 = 0 \Rightarrow [\boldsymbol{x_1x_2}\ \boldsymbol{y_1x_2}\ \boldsymbol{z_1x_2}\ \boldsymbol{x_1y_2}\ \boldsymbol{y_1y_2}\ \boldsymbol{z_1y_2}\ \boldsymbol{x_1z_2}\ \boldsymbol{y_1z_2}\ \boldsymbol{z_1z_2}]E = 0$$

which can be solved with SVD



$$p_2^T E\, p_1 = 0 \qquad \textit{Epipolar constraint}$$

$$\mathrm{E} = [t]_\times R \qquad \textit{essential matrix}$$

$$[\boldsymbol{t}]_\times = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix}$$

Relative Pose estimation - 2D Image Points

- The scale problem:
  - The essential matrix is calculated up to scale, That means that the t of Rt is also up to scale:
  - How can we recover it?

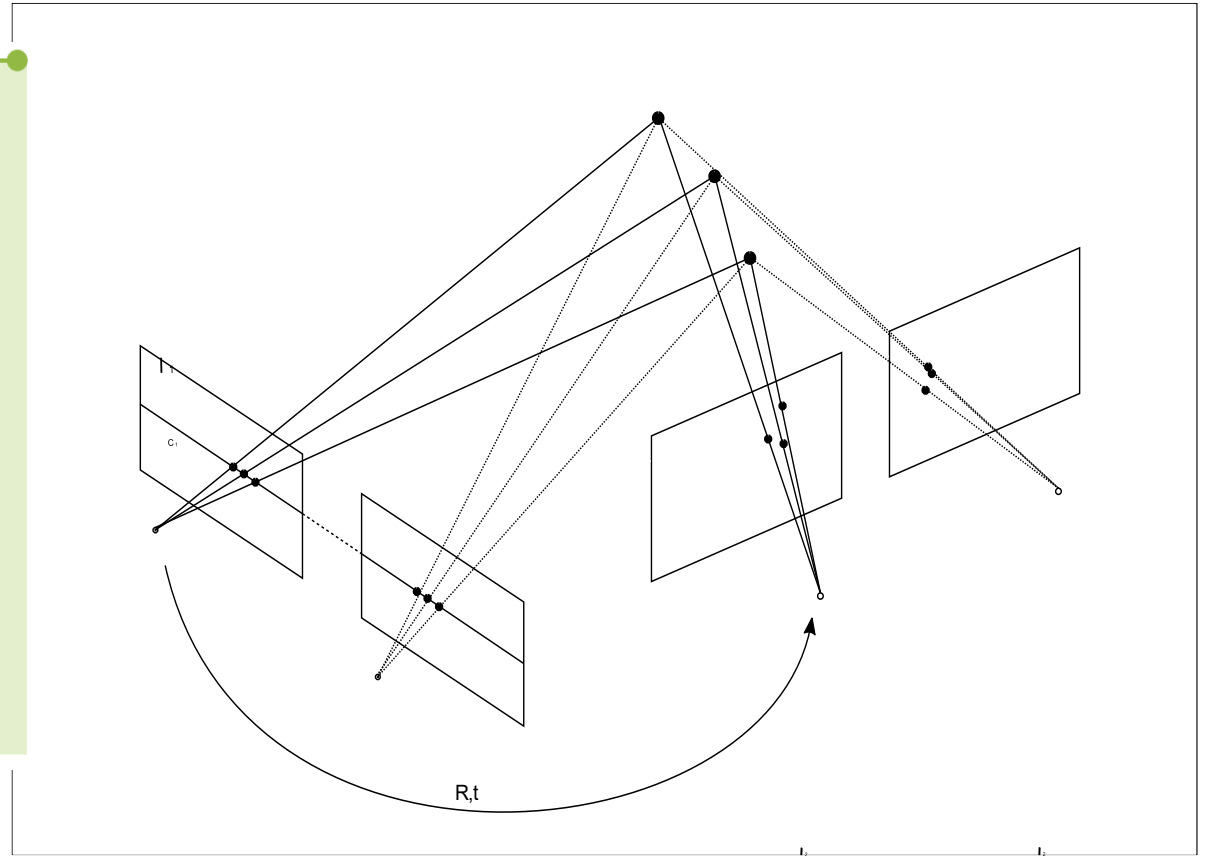
    We have to figure out a relative accurate movement between frames f_k and f_{k-1}

    Any ideas?

Visual Odometry 3D – to 3D

Algorithm 2. VO from 3-D-to-3-D correspondences.

1) Capture two stereo image pairs $I_{l,k-1}, I_{r,k-1}$ and $I_{l,k}, I_{r,k}$
2) Extract and match features between $I_{l,k-1}$ and $I_{l,k}$
3) Triangulate matched features for each stereo pair
4) Compute T_k from 3-D features X_{k-1} and X_k
5) Concatenate transformation by computing
   $C_k = C_{k-1}T_k$
6) Repeat from 1).



3D-3D

Visual Odometry 3D – to 2D

Algorithm 3. VO from 3-D-to-2-D Correspondences.

1) Do only once:

1.1) Capture two frames I_{k-2}, I_{k-1}

1.2) Extract and match features between them

1.3) Triangulate features from I_{k-2}, I_{k-1}

2) Do at each iteration:

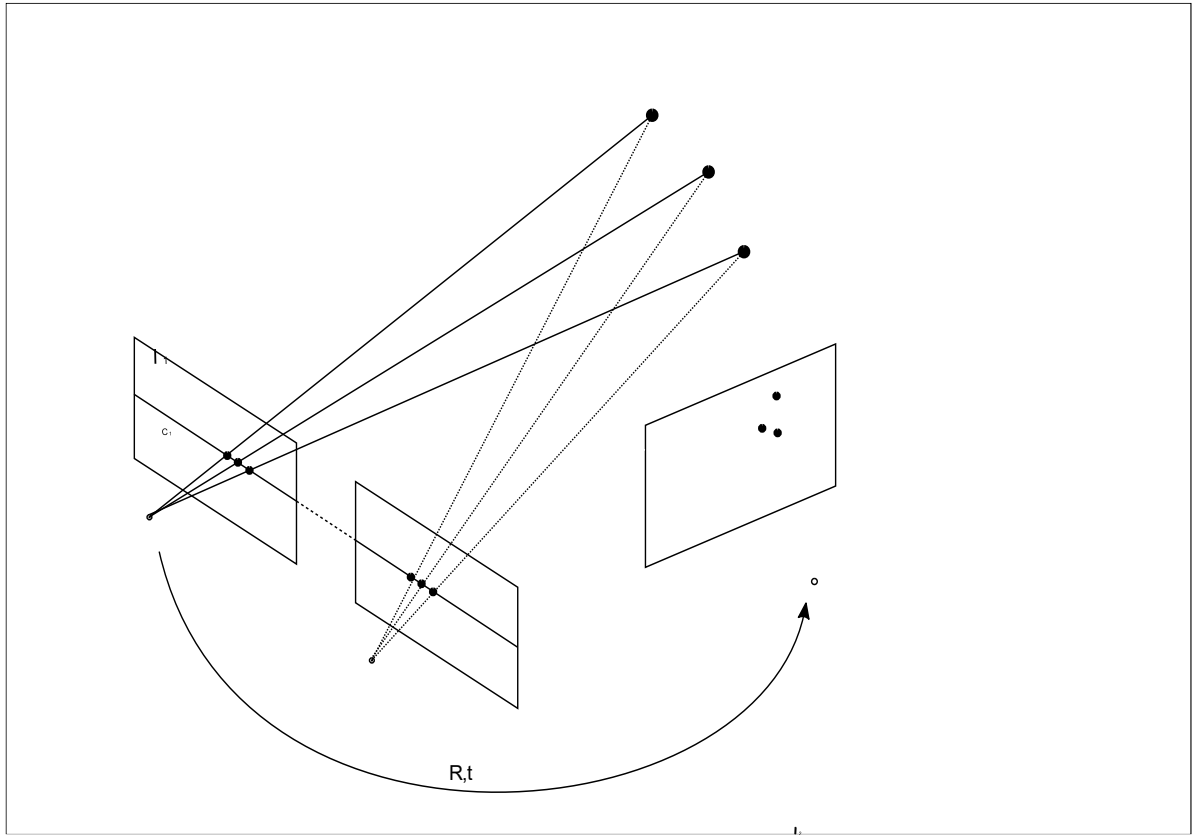
2.1) Capture new frame I_k

2.2) Extract features and match with previous frame I_{k-1}

2.3) Compute camera pose (PnP) from 3-D-to-2-D matches

2.4) Triangulate all new feature matches between I_k and I_{k-1}

2.5) Iterate from 2.1).

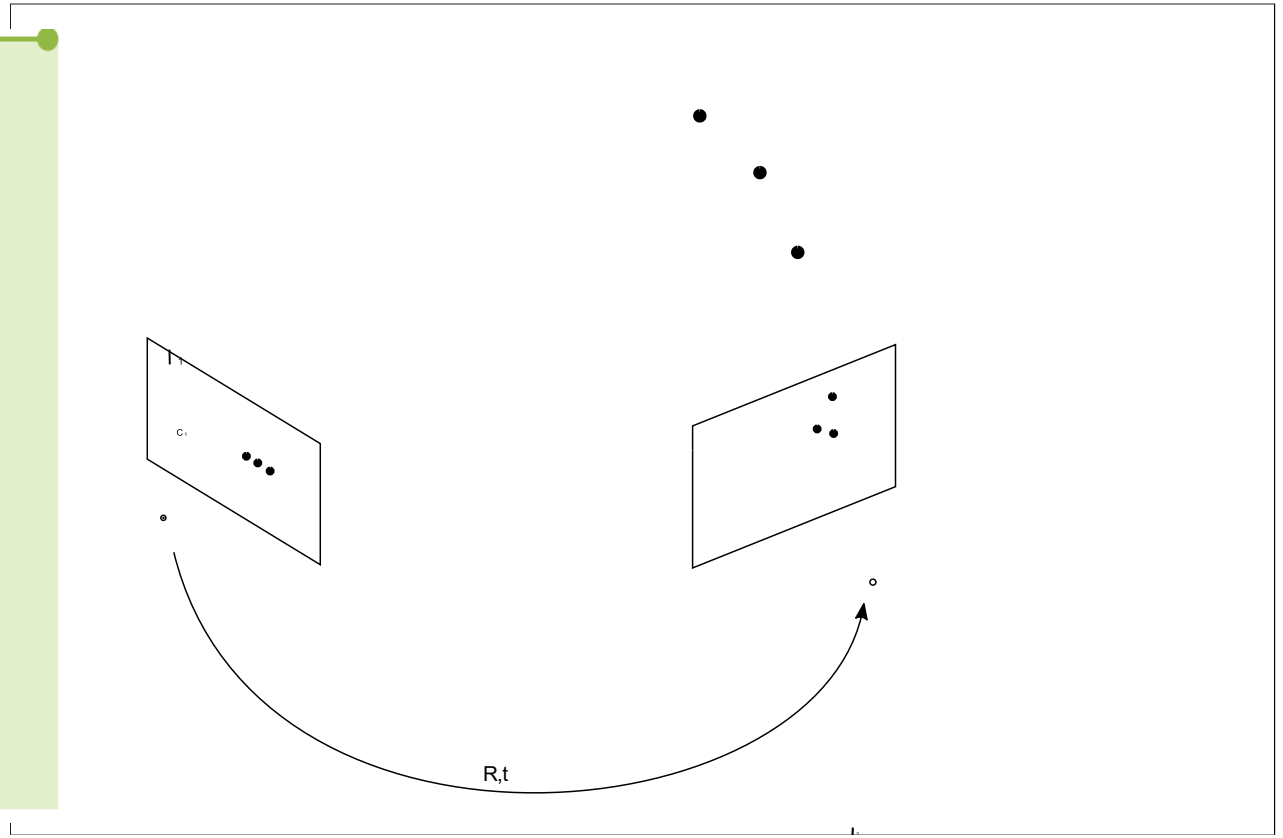


3D-2D

Visual Odometry 2D – to 2D

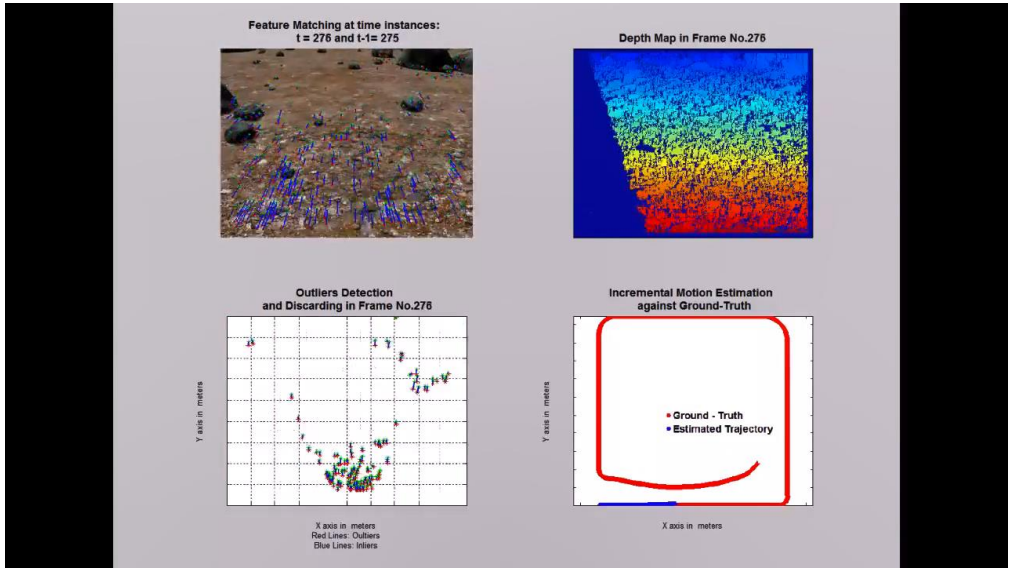
Algorithm 1. VO from 2-D-to-2-D correspondences.

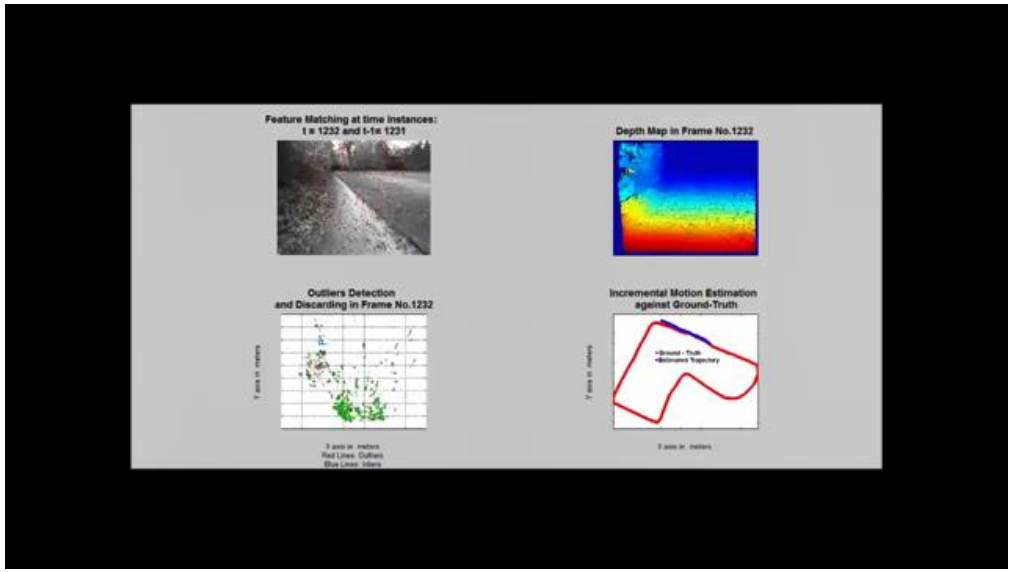
1) Capture new frame I_k
2) Extract and match features between I_{k-1} and I_k
3) Compute essential matrix for image pair I_{k-1}, I_k
4) Decompose essential matrix into R_k and t_k, and form T_k
5) Compute relative scale and rescale t_k accordingly
6) Concatenate transformation by computing $C_k = C_{k-1}T_k$
7) Repeat from 1).



Some Notes

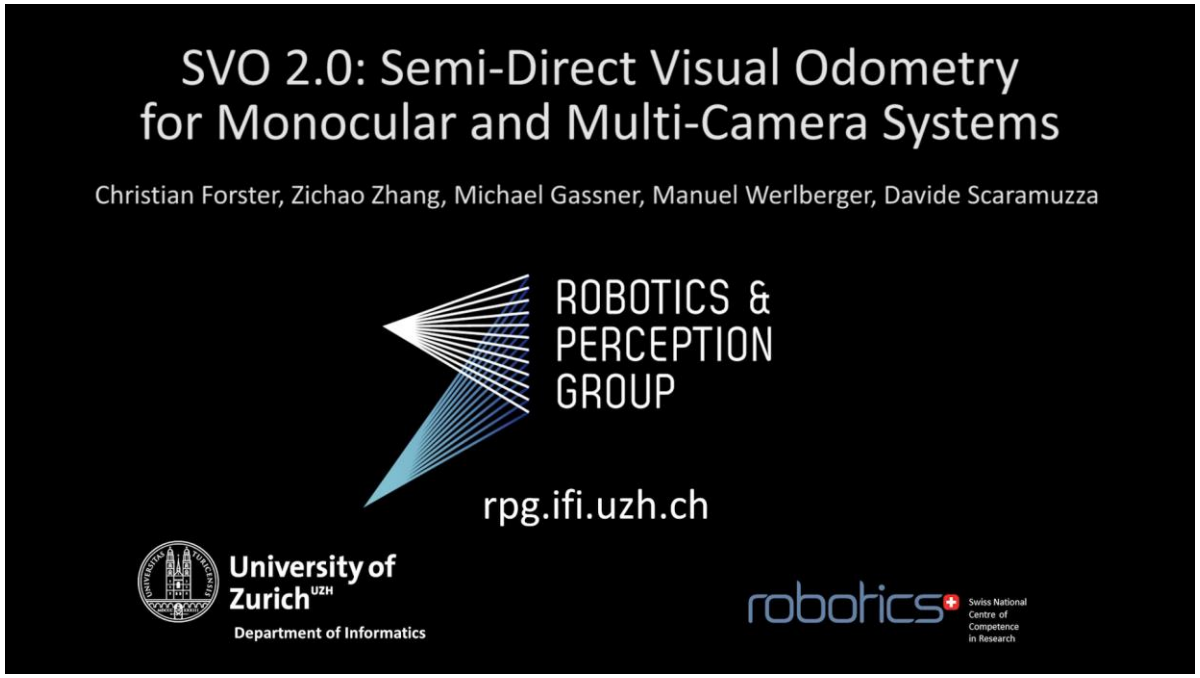
- 2D-2D and 3D-2D are better than 3D-3D. Why?
- Stereo VO even with 2D-2D is better. Why?
- Any ideas on improving VO?

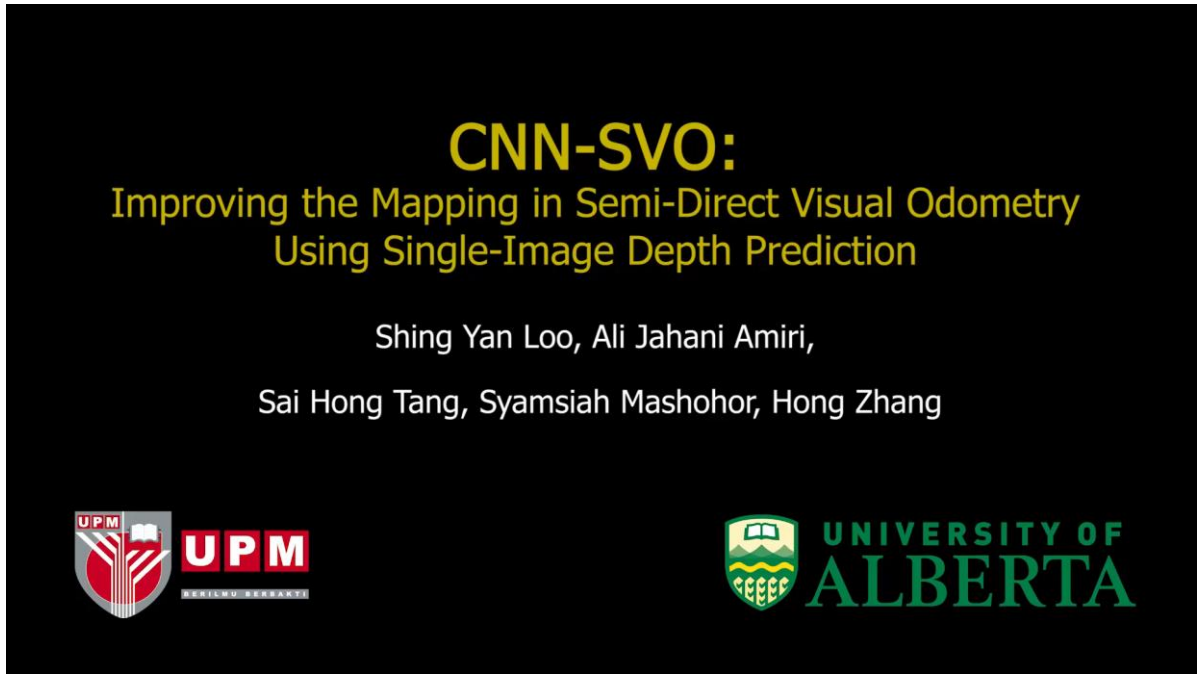




Some Notes

- 2D-2D and 3D-32 are better than 3D-3D. Why?
- Stereo VO even with 2D-2D is better. Why?
- Any ideas on improving VO?



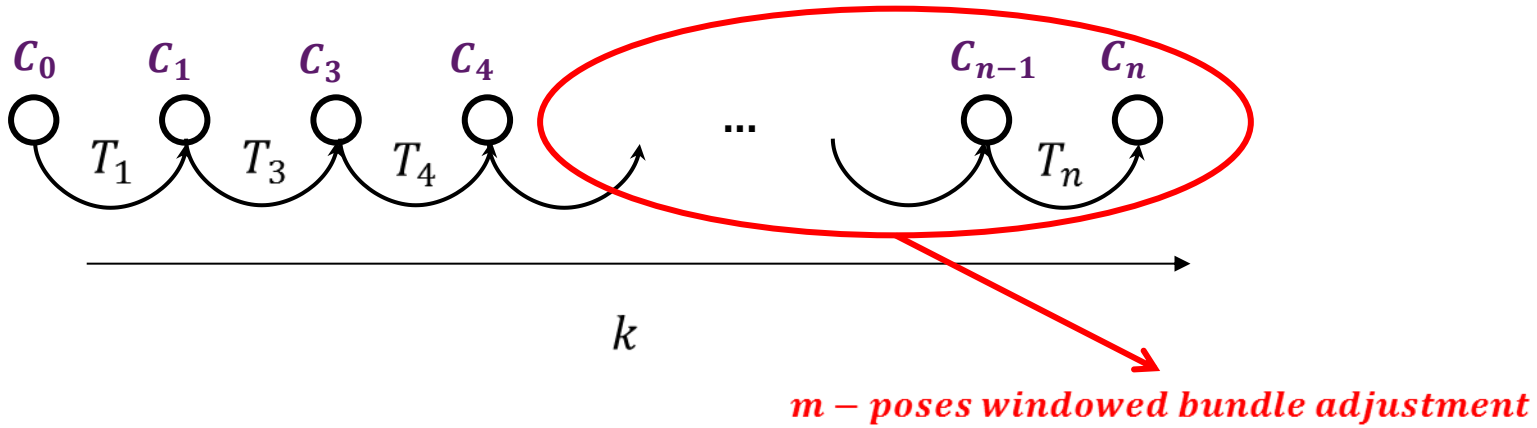


What happens over time?

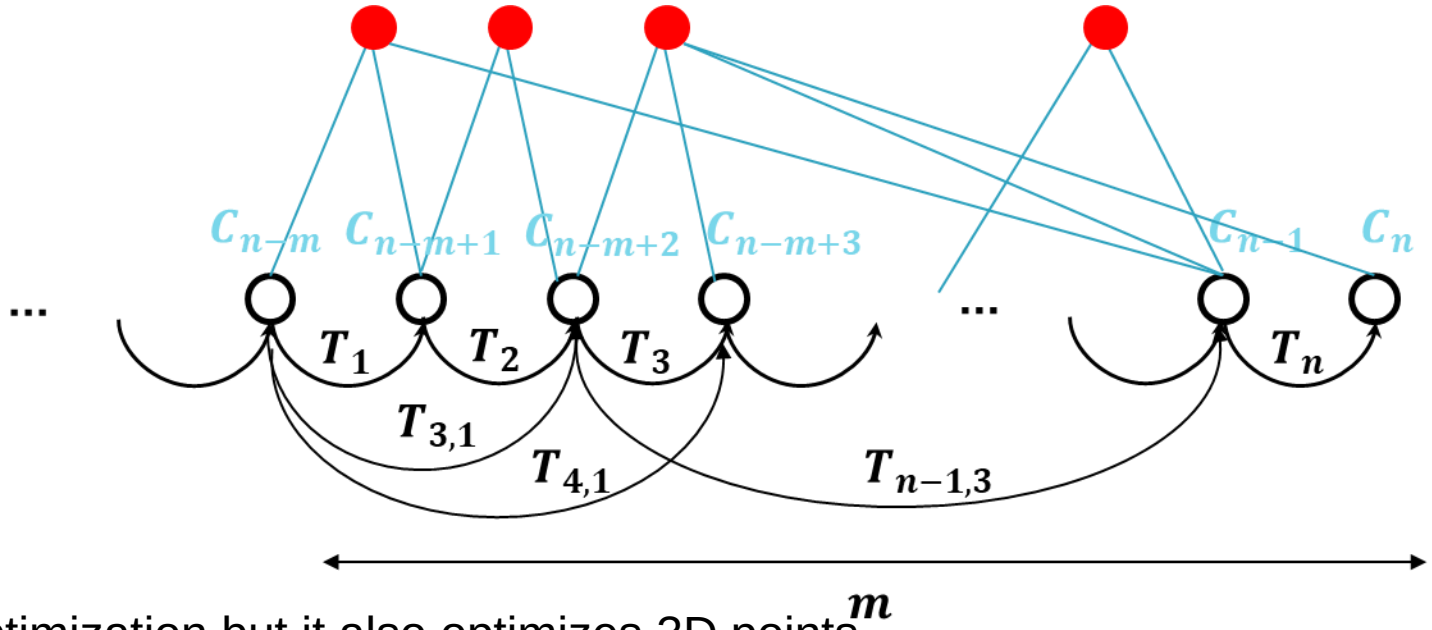
- VO accumulates the transformations T_k from frame f_{k-1} to frame f_k over time providing the full trajectory $C_{0:n}$.
- What is the problem with that?

- How can we solve it?

- We can optimize over multiple frames in a procedure which is called **bundle adjustment**



Windowed Bundle Adjustment (BA)



- Similar to pose-optimization but it also optimizes 3D points

- In order to not get stuck in local minima, the initialization should be close the minimum
- Levenberg-Marquadt can be used

Improving the Accuracy of VO

- Other sensors can be used such as
  - IMU (called inertial VO)
  - Compass
  - GPS
  - Laser

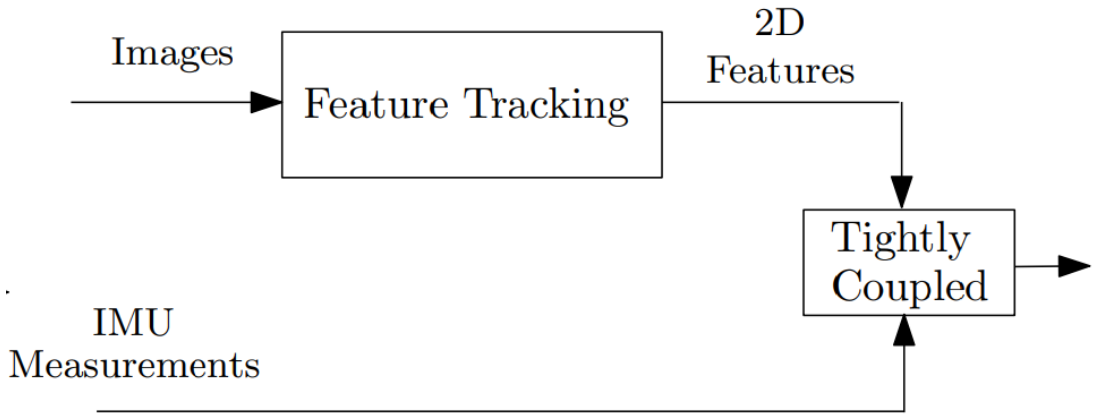
- An IMU combined with a single camera allows the estimation of the absolute scale. Why?

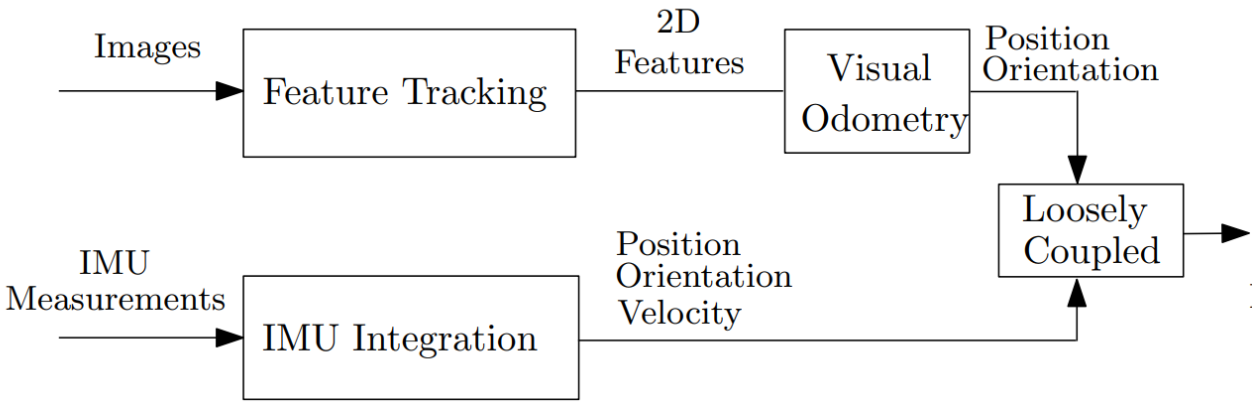
Visual Inertial Odometry VIO

"Visual-Inertial odometry (VIO) is the process of estimating the state (pose and velocity) of an agent (e.g., an aerial robot) by using only the input of one or more cameras plus one or more Inertial Measurement Units (IMUs) attached to it"

- Cameras are slow and information rich
- IMUs are fast but high noise

- Two main paradigms of Filtering for State Estimation:
  – Loosely coupled
  – Tightly Coupled
  (Depending on the integration of the visual info)





Scaramuzza, D. and Zhang, Z., 2019. Visual-Inertial Odometry of Aerial Robots. *arXiv preprint arXiv:1906.03289*.

Visual Inertial Odometry VIO

- Kalman state:

$$\mathbf{X}_i = [\mathbf{T}_{WI}^i,\ \mathbf{v}_{WI}^i,\ \mathbf{b}_a^i,\ \mathbf{b}_g^i], \quad i = 1,\ 2,\ 3,\ \ldots,\ N$$

- , where T_{w1}^i is the 6-DoF pose of the IMU, v_{w1}^i is the velocity of the IMU, b_a^i and B_g^i are the biases of the accelerometer and gyroscope respectively.

- a single moving camera allows us to measure the geometry of the 3D scene and the camera motion up to an unknown metric scale:
  - the projection function satisfies project(p) = project(s· p) for an arbitrary scalar s and an arbitrary point p;
  - a single IMU, instead, renders metric scale and gravity observable (due to the presence of gravity)

VIO, notable examples

- MSCKF, Multi-State Constraint Kalman Filter (MSCKF) –tightly
- OKVIS (Leutenegger et al 2013, 2015) - Open Keyframe-based Visual-Inertial SLAM (OKVIS) –tightly
- Robust Visual Inertial Odometry (ROVIO) is a visual-inertial state estimator based on an extended – tightly Kalman Filter (EKF)
- VINS-Mono –loosely
- SVO+MSF –loosely

SumUP

- Visual Odometry
  - 3D-3D
  - 3D-2D
  - 2D-2D
- Local Bundle Adjustment
- Visual Inertial Odometry -VIO
  - Loosely Coupled EKF
  - Tightly Coupled EKF

Perception for Autonomous Systems 34579:

Visual Odometry

Lecturer: Evangelos Boukas—PhD