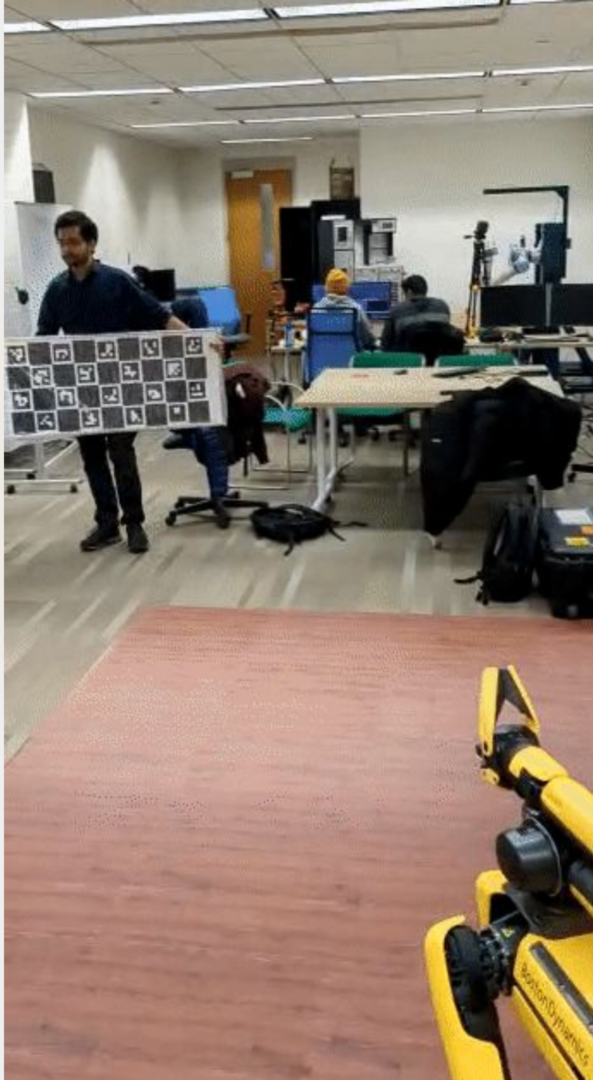
DeepRob

[Student] Lecture 20

by Nikhilanj, Ebasa, Ritik

Visual Odometry and Localization

University of Michigan and University of Minnesota



Agenda

Localization

Visual localization

Visual odometry

Methods in VO

PoseNet

TrainFlow

Localization

Estimating an object or robot's position in a known environment.

- Essential for navigation, mapping, and perception tasks
- Key component in robotics, autonomous vehicles, and augmented reality
- Often requires combining multiple techniques to enhance accuracy and robustness

Source: [Pure Visual Localization for Boston Dynamics Spot For Airlab]

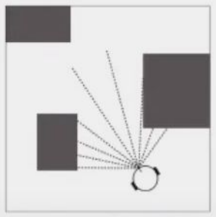
Types of Localization

1. Dead Reckoning (Position Tracking)
    - Initial position known
    - Blindly update pose based on differential movements

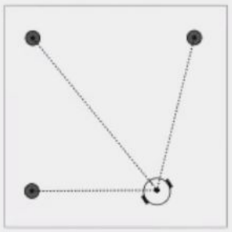
2. Global Localization
    - Initial position can be unknown
        1. Map-Based (with landmarks)
        2. Beacon-Based (with active infrastructure)

3. Global Localization and Position Tracking Combined
    - Combines the strengths of dead reckoning and global localization
    - Offers improved accuracy and robustness

odometry-based



map-based



beacon-based

Source: Mobile Robot Systems Course - Amanda Prorok, University of Cambridge

Pre-DL Methods with LiDAR/Range Data

LiDAR (Light Detection and Ranging) uses lasers to measure distances

Range data represents distances between sensors and objects in the environment

Common types of range sensors: ultrasonic, infrared, time-of-flight cameras

Limitations

- Susceptible to interference from environmental factors (e.g., rain, fog, dust)
- Limited field of view compared to cameras
- Higher cost and power consumption than some alternative sensors
- Limited semantic information, primarily provides geometric data

Source: Turtlebot 3 350 Lidar Sensor LDS-01

Localization in Robotics

Robotics often deals with uncertainty in localization. A robot's true state cannot be measured directly; it must be inferred

Key components of probabilistic localization

- Robot's belief about its state:
  Estimate of the robot's position and orientation within the environment

Localization in Robotics

Robotics often deals with uncertainty in localization. A robot's true state cannot be measured directly; it must be inferred

Key components of probabilistic localization

Robot's belief about its state:
Estimate of the robot's position and orientation within the environment

Robot motion model:
Describes how the robot's state changes over time due to its motion

Localization in Robotics

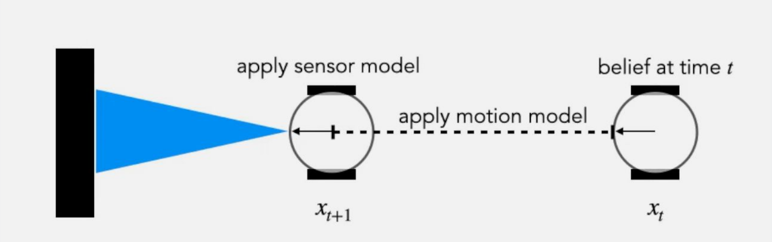
Robotics often deals with uncertainty in localization. A robot's true state cannot be measured directly; it must be inferred

Key components of probabilistic localization

Robot's belief about its state:
Estimate of the robot's position and orientation within the environment

Robot motion model:
Describes how the robot's state changes over time due to its motion

Robot sensor (observation) model:
Defines the relationship between the robot's state and sensor measurements



Source: Mobile Robot Systems Course - Amanda Prorok, University of Cambridge

Bayes' Rule in Robotics

Generative model:

- Let's assume x is the robot state and z is the measurement data

Bayes' Rule in Robotics

Generative model:

- Let's assume x is the robot state and z is the measurement data
- Describes how a state variable causes sensor measurements: $p(z|x)$

Bayes' Rule in Robotics

Generative model:

- Let's assume x is the robot state and z is the measurement data
- Describes how a state variable causes sensor measurements: $p(z|x)$
- Helps to estimate the robot's state based on the relationship between the state and the sensor measurements

DR

Bayes' Rule in Robotics

Generative model:

- Let's assume x is the robot state and z is the measurement data
- Describes how a state variable causes sensor measurements: p(z|x)
- Helps to estimate the robot's state based on the relationship between the state and the sensor measurements

Bayes' Rule

- Updates a robot's state estimate based on sensor measurements and prior belief

sensor model

normalizes density

$$p(x\,|\,z) = \frac{p(z\,|\,x)\; p(x)}{p(z)} = \frac{p(z\,|\,x)\; p(x)}{\sum_{x'} p(z\,|\,x')\; p(x')} \qquad p(x\,|\,z) = \eta\; p(z\,|\,x)\; p(x)$$

theorem of total probability

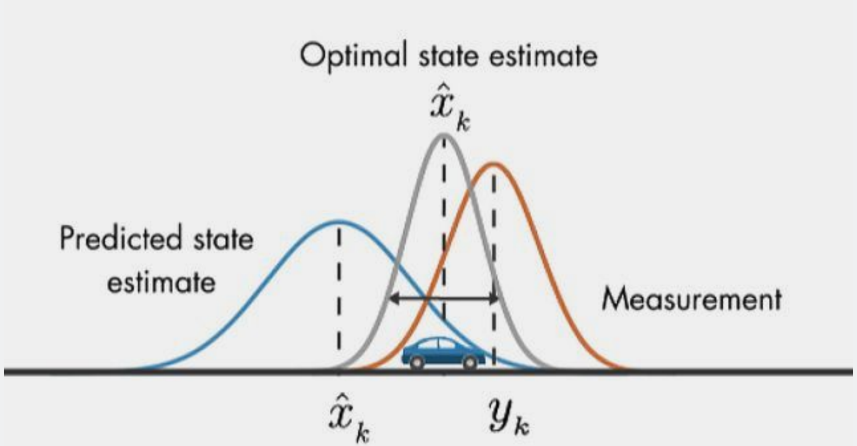
denominator does not depend on x

Source: Mobile Robot Systems Course - Amanda Prorok, University of Cambridge

Kalman Filter (KF)

- Assumes Gaussian distributions and linear dynamics
- Efficient and optimal for linear, Gaussian systems
- Easy to implement and suitable for real-time applications

Extended Kalman Filter (EKF): An extension of the Kalman Filter for non-linear systems



Source: Mathwork, Understanding Kalman Filter

KF
1960

1960
EKF

Particle Filter (PF)

A "particle" refers to a hypothetical representation of the robot's state at a specific point in time.

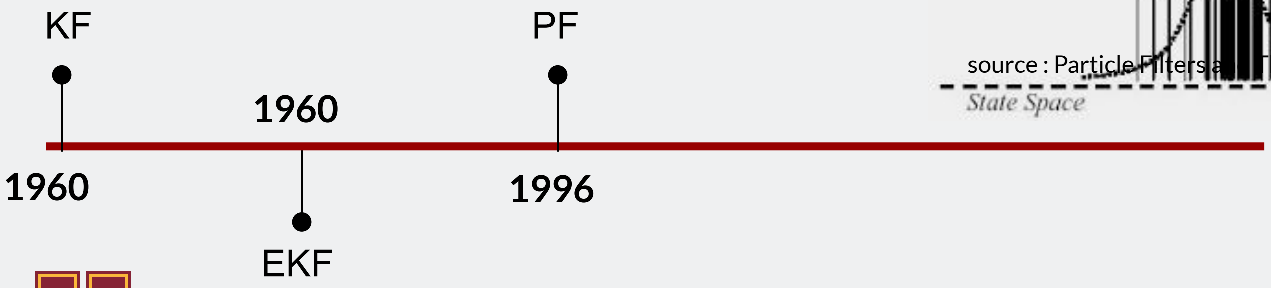
Represents the posterior distribution using a set of weighted particles.

Well-suited for non-linear, non-Gaussian systems.

Adapts to changing dynamics in the environment.

Can represent multimodal distributions.

source : Particle Filters and Their Applications, Kaijen et al. MIT



Visual Localization

Estimating the position and orientation of a robot or camera within its environment using visual data

Provides rich information from the environment

Can work in environments where other sensors may fail or be less accurate

Complements other localization methods (e.g., LiDAR/range-based)

Visual data can be more descriptive and versatile

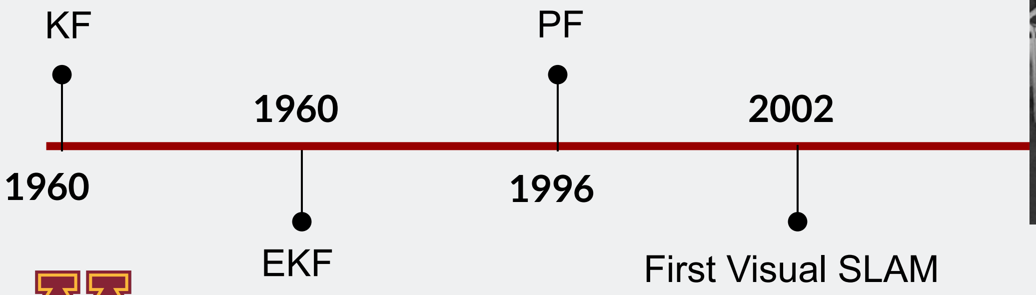
SLAM with Active Vision

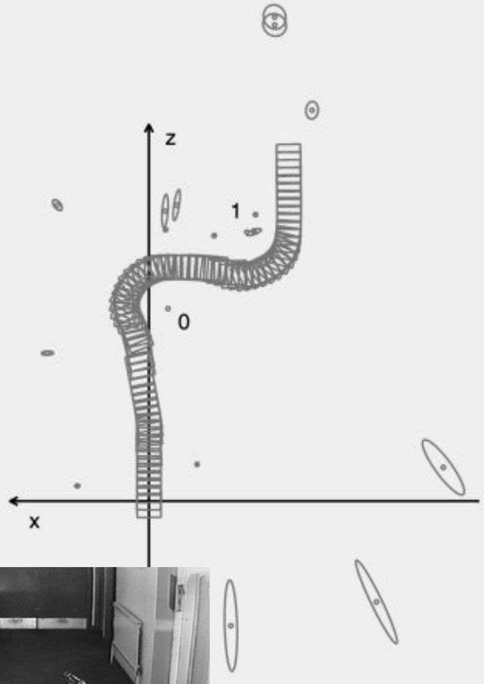
This work by Andrew Davison was the first to implement visual SLAM

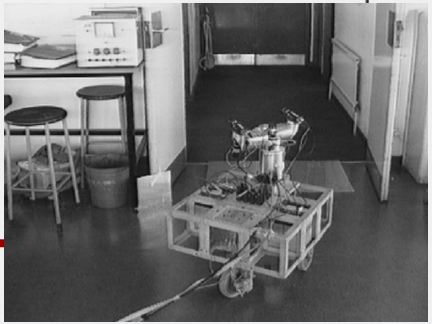
Focused measurement capability and wide field of view

Exploits naturally occurring, automatically detected features for long-term localization

Uncertainty-based measurement selection







Reference: https://ieeexplore.ieee.org/document/1017615

MonoSLAM: Real-time single camera SLAM

- First real-time monocular SLAM approach, achieving drift-free performance
- Online creation of a sparse, persistent map of natural landmarks within a probabilistic framework
- Active approach to mapping and measurement
- General motion model for smooth camera movement

KF 1960

EKF 1960

PF 1996

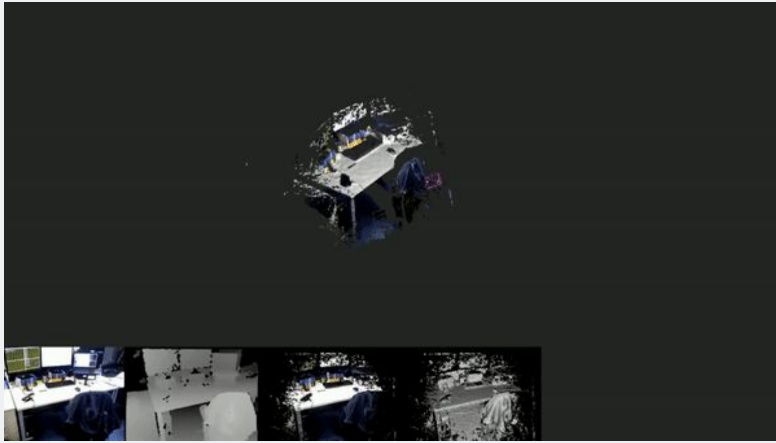
First Visual SLAM 2002

MonoSLAM 2007

Reference: https://ieeexplore.ieee.org/document/4160954

Dense SLAM

- Estimates camera or robot's position in real-time
- Captures detailed environmental structure
- Contrasts with sparse SLAM methods
- Utilizes more environmental data points



KF

1960

1960

EKF

PF

1996

2002

Visual SLAM

MonoSLAM

2007

2015

ElasticFusion

Reference: https://www.roboticsproceedings.org/rss11/p01.pdf

Questions?

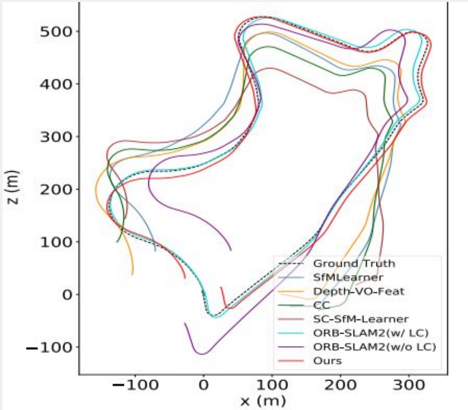
Visual Odometry

Visual Odometry

Process to estimate self-motion of an agent using input from one or more cameras attached to it.

Input

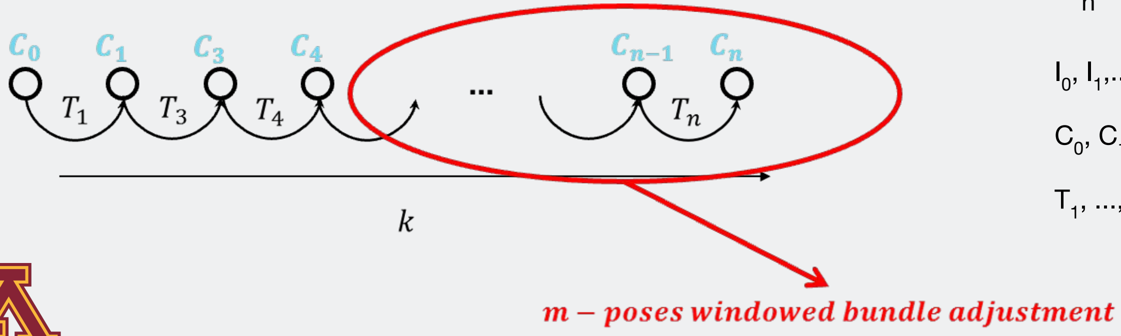
Output



Zhao, W., Liu, S., Shu, Y., & Liu, Y. J. (2020). Towards better generalization: Joint depth-pose learning without posenet. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 9151-9161)

Problem Formulation

- The main task in VO is to compute the relative transformations T_k from the images I_k and I_{k-1} & then to concatenate the transformations to recover the full trajectory $C_{0:n}$ of the camera.



$C_n = C_{n-1}T_n$

$I_0, I_1,..,I_{k-1}, I_k$: Image Sequence

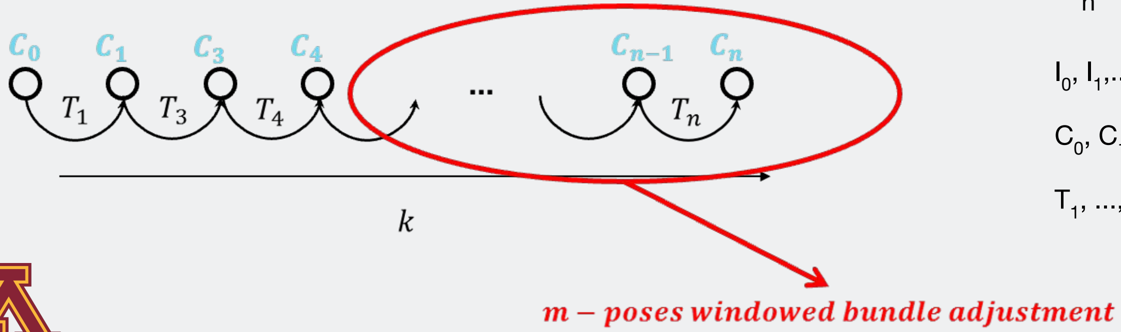
$C_0, C_1.. C_n$: Camera Poses

$T_1, ..., T_n$: Transformations

Slides are referenced from [Davide Scaramuzzas Tutorial on Visual Odometry]

Problem Formulation

- The main task in VO is to compute the relative transformations T_k from the images I_k and I_{k-1} & then to concatenate the transformations to recover the full trajectory $C_{0:n}$ of the camera.
- This means that VO recovers the path incrementally, pose after pose.



$C_n = C_{n-1}T_n$

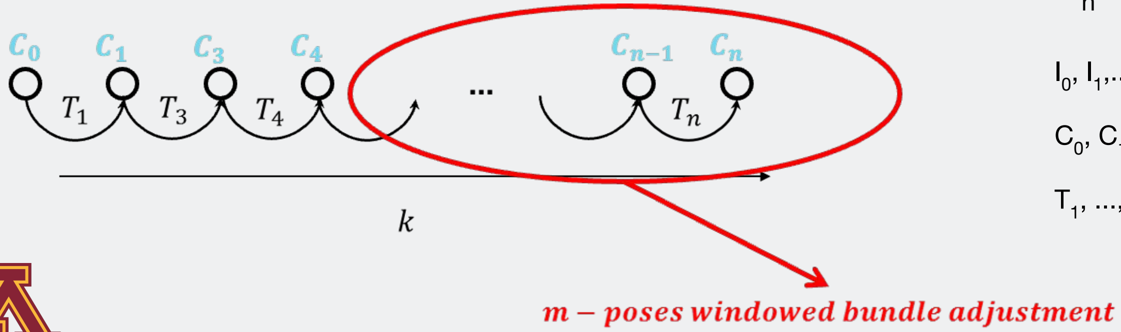
$I_0, I_1,..,I_{k-1}, I_k$: Image Sequence

$C_0, C_1.. C_n$: Camera Poses

$T_1, ..., T_n$: Transformations

Problem Formulation

- The main task in VO is to compute the relative transformations T_k from the images I_k and I_{k-1} & then to concatenate the transformations to recover the full trajectory $C_{0:n}$ of the camera.
- This means that VO recovers the path incrementally, pose after pose.
- An iterative refinement over last m poses can be performed after this step to obtain a more accurate estimate of the local trajectory.



$C_n = C_{n-1} T_n$

$I_0, I_1, .., I_{k-1}, I_k$: Image Sequence

$C_0, C_1 .. C_n$: Camera Poses

$T_1, ..., T_n$: Transformations

Steps Involved

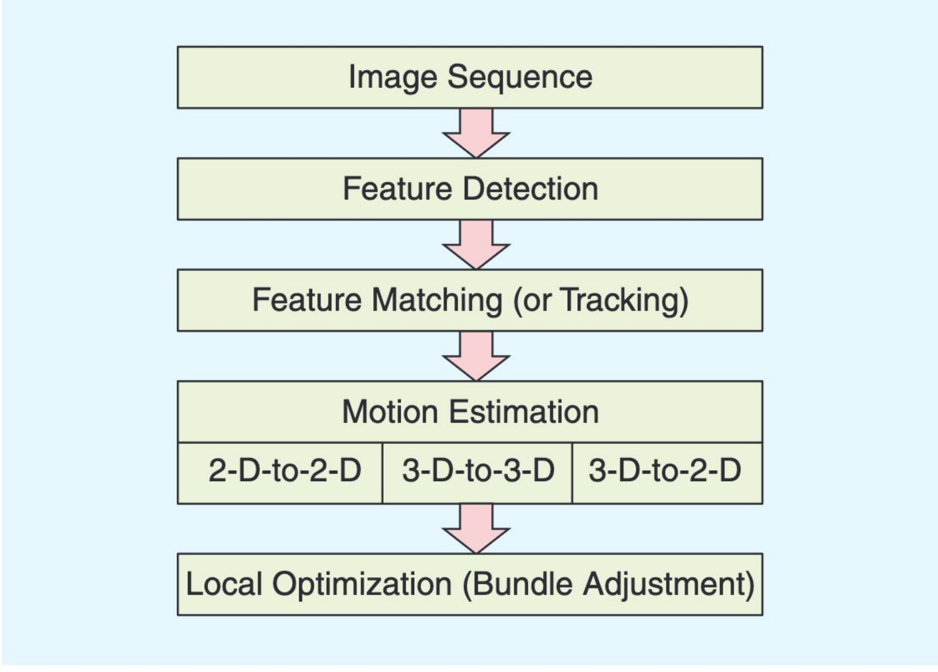
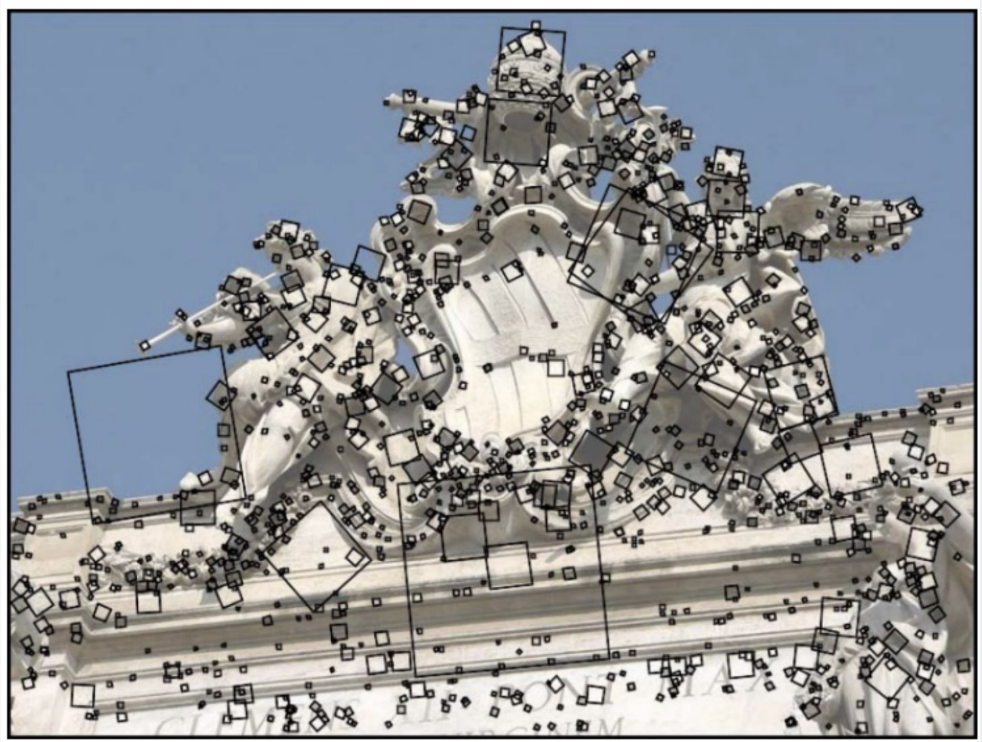


Image from Scaramuzza and Fraundorfer, 2011

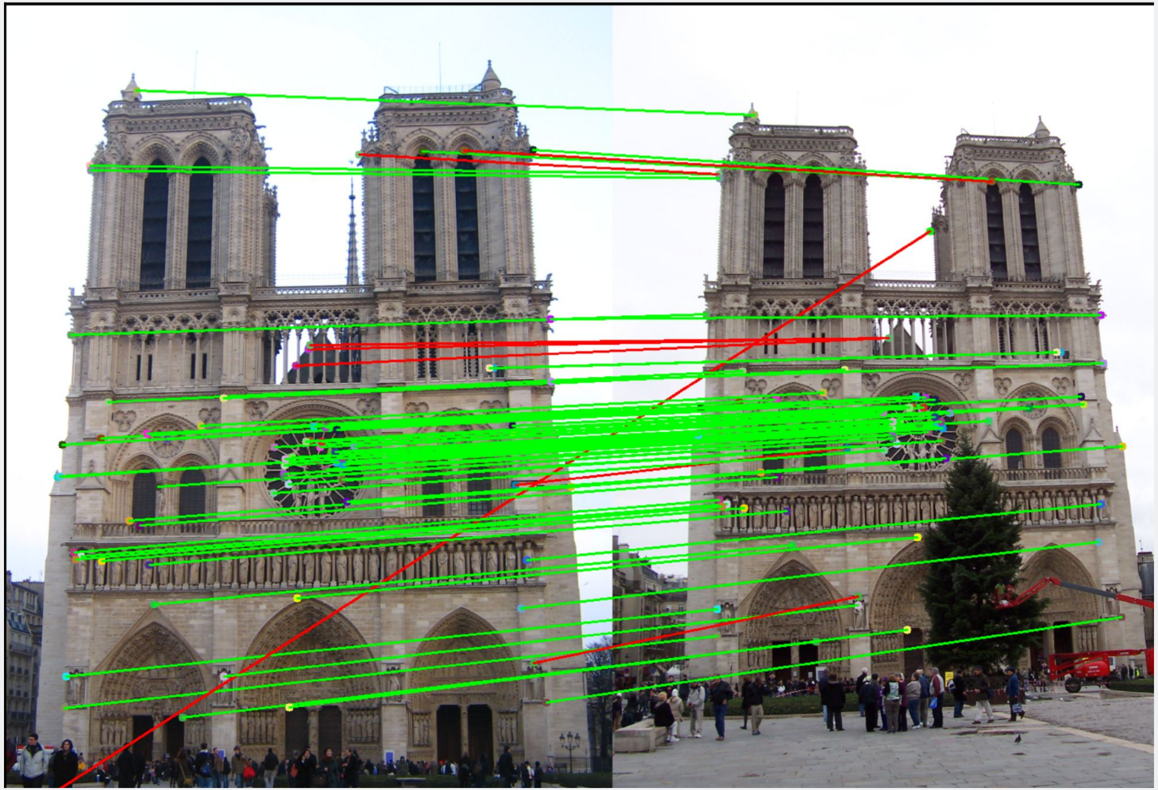
Feature Detection

Detections Algorithms:
SIFT, SURF, ORB, etc



Schonberger and Frahm: Structure-from-Motion Revisited. CVPR, 2016.

Feature Matching



Source: GaTech

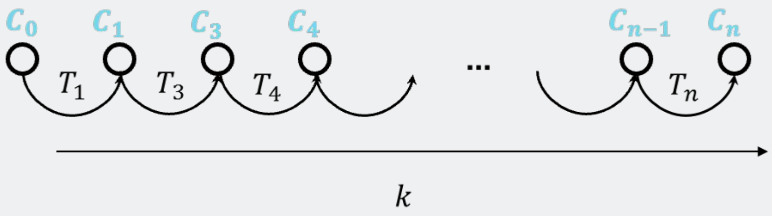
Motion Estimation

Core step in VO computation

Computes the camera motion T_k between previous and current frame

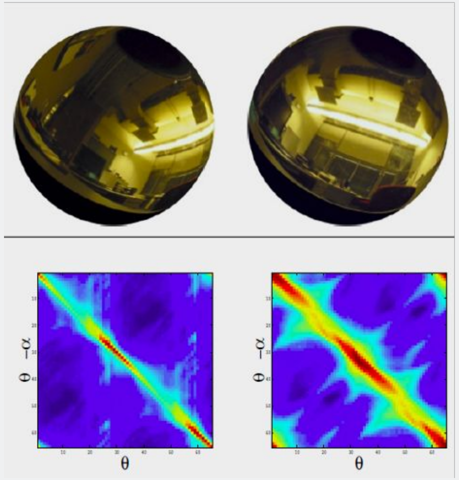
$$T_k = \begin{bmatrix} R_{k,k-1} & t_{k,k-1} \\ 0 & 1 \end{bmatrix}$$

By concatenating all these single movements, full trajectory of the camera can be recovered



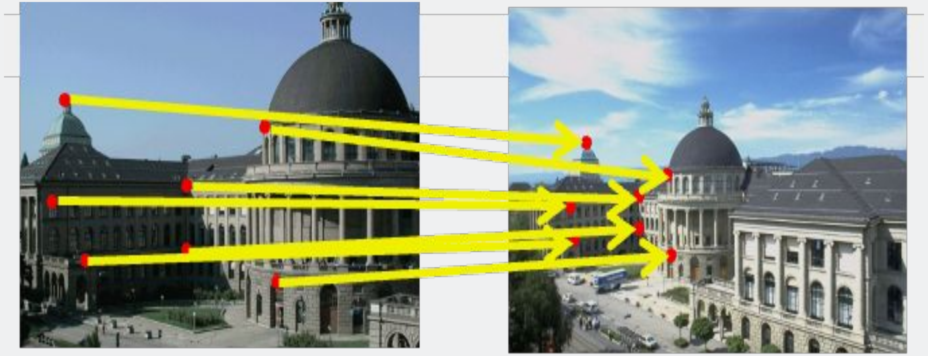
Appearance or Feature Based?

Appearance Based



Makadia et al. «Correspondence-free structure from motion», IJCV'07

Feature Based



Feature based is faster and more accurate so most VO techniques use that

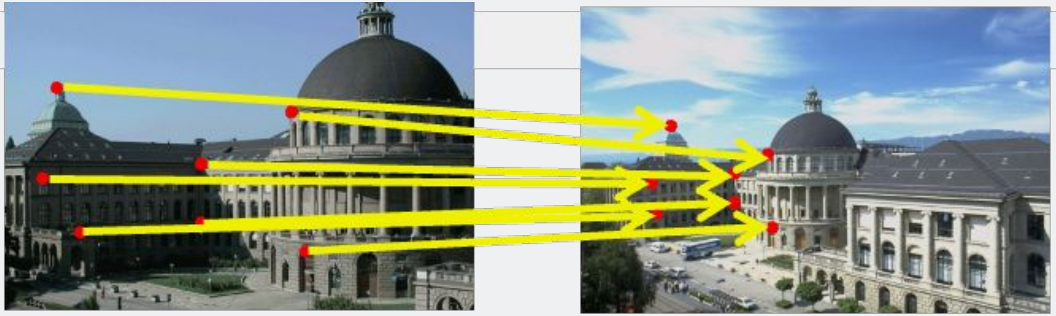
Motion Estimation

Depending on whether the feature correspondences f_{k-1} and f_k are specified in 2D or 3D, there are 3 different cases:

- **2D to 2D**: both f_{k-1} and f_k are specified in 2D image coordinates
- **3D to 2D**: f_{k-1} are specified in 3D and f_k are its corresponding 2D reprojections on Image I_k
- **3D to 3D**: both f_{k-1} and f_k are specified in 3D. For this, you need to triangulate the 3D points at each time instance

2D to 2D

- Both f_{k-1} and f_k are specified in 2D
- Can be solved by recovering the Fundamental Matrix through the 8 Point Algorithm



I_{k-1} I_k

Slides are referenced from [Davide Scaramuzzas Tutorial on Visual Odometry]

2D to 2D

We have the correspondences between two images (Through feature detection and matching we did earlier)

So how do we relate those correspondences between the two images?

And how do we recover the relative pose between the cameras from it?

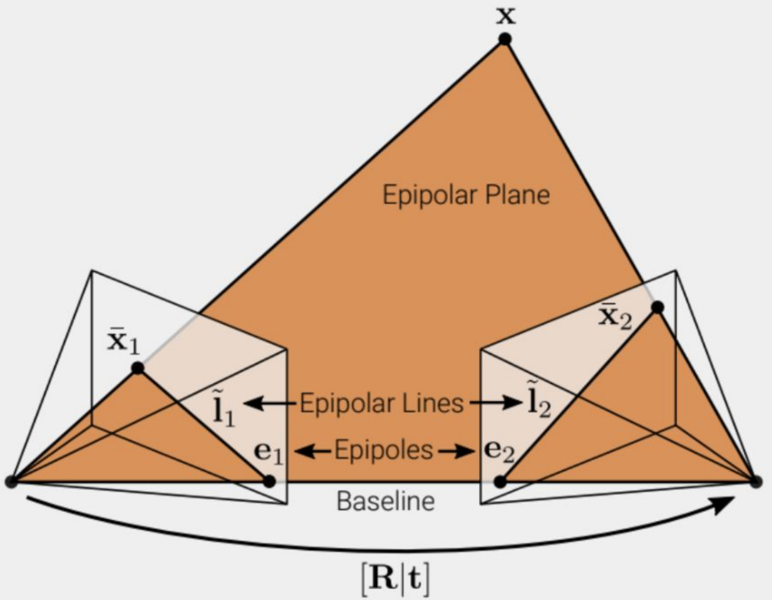
2D to 2D

There is a way to relate the correspondences:

- **Epipolar Geometry:** Study of the relationship between two camera views of the same scene
- **Fundamental Matrix:** Transformation that maps a feature in one image to its corresponding feature in another image
- **8 Point Algorithm:** Algorithm used to estimate the Fundamental Matrix

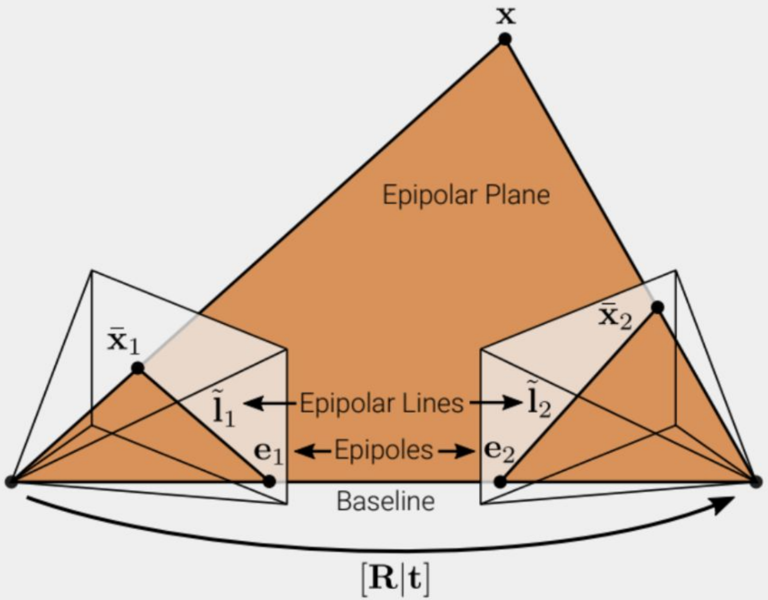
Epipolar Geometry

- Let $\mathbf{R}$ and $\mathbf{t}$ denote the relative pose between **two perspective cameras**



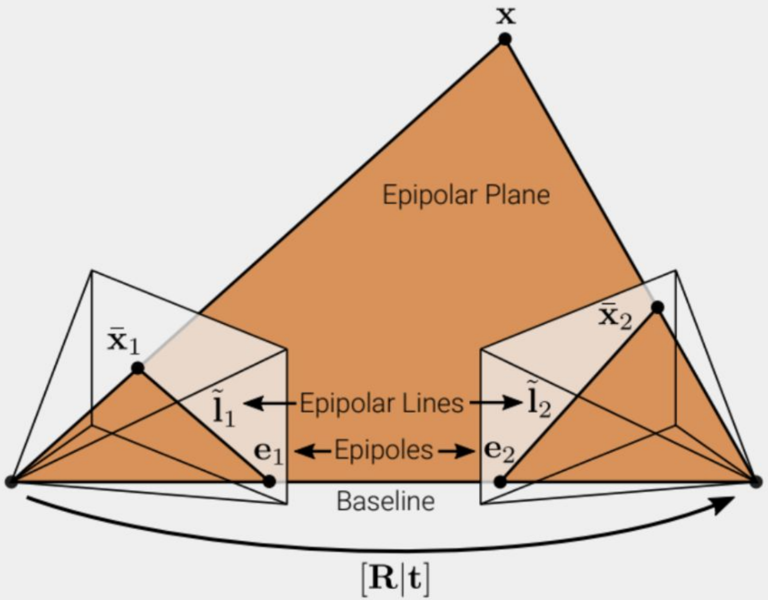
Epipolar Geometry

- Let $\mathbf{R}$ and $\mathbf{t}$ denote the relative pose between **two perspective cameras**
- A 3D point $\mathbf{x}$ is projected to pixel $\bar{\mathbf{x}}_1$ in image 1 and to pixel $\bar{\mathbf{x}}_2$ in image 2



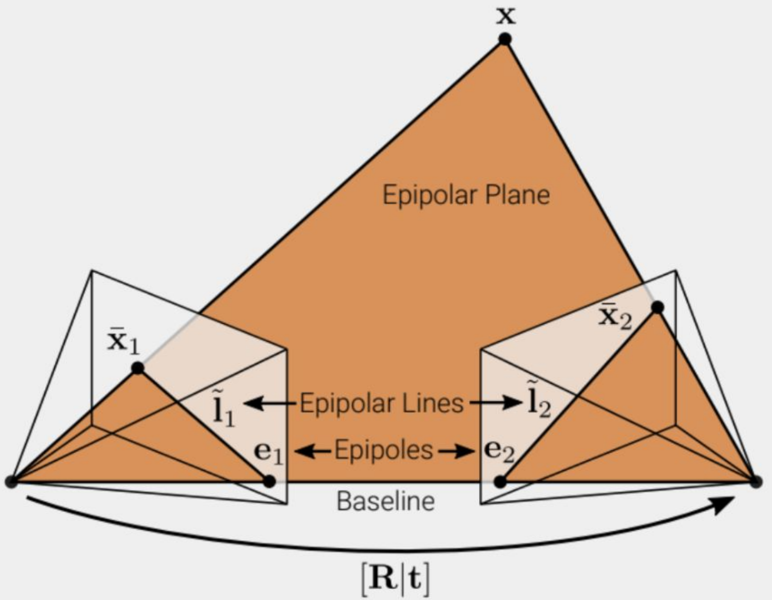
Epipolar Geometry

- Let $\mathbf{R}$ and $\mathbf{t}$ denote the relative pose between **two perspective cameras**
- A 3D point $\mathbf{x}$ is projected to pixel $\bar{\mathbf{x}}_1$ in image 1 and to pixel $\bar{\mathbf{x}}_2$ in image 2
- The 3D point $\mathbf{x}$ and the two camera centers span the **epipolar plane**



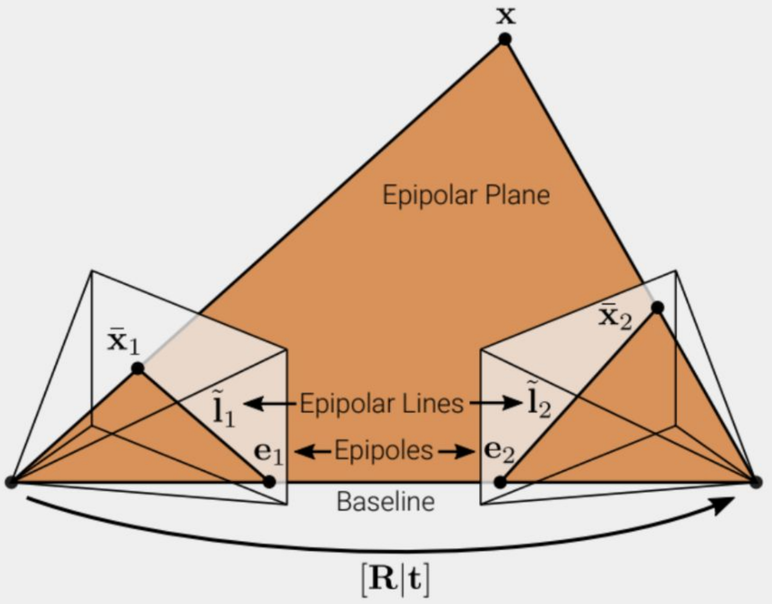
Epipolar Geometry

- Let $\mathbf{R}$ and $\mathbf{t}$ denote the relative pose between **two perspective cameras**
- A 3D point $\mathbf{x}$ is projected to pixel $\bar{\mathbf{x}}_1$ in image 1 and to pixel $\bar{\mathbf{x}}_2$ in image 2
- The 3D point $\mathbf{x}$ and the two camera centers span the **epipolar plane**
- The correspondence of pixel $\bar{\mathbf{x}}_1$ in image 2 must lie on the **epipolar line** $\tilde{\mathbf{l}}_2$ in image 2



Epipolar Geometry

- Let $\mathbf{R}$ and $\mathbf{t}$ denote the relative pose between **two perspective cameras**
- A 3D point $\mathbf{x}$ is projected to pixel $\bar{\mathbf{x}}_1$ in image 1 and to pixel $\bar{\mathbf{x}}_2$ in image 2
- The 3D point $\mathbf{x}$ and the two camera centers span the **epipolar plane**
- The correspondence of pixel $\bar{\mathbf{x}}_1$ in image 2 must lie on the **epipolar line** $\tilde{\mathbf{l}}_2$ in image 2
- All epipolar lines pass through the **epipole**



Epipolar Geometry

- 2D search space reduced to 1D search space
- We don't know the essential/fundamental matrix which can project one point on its corresponding epipolar line on the 2nd image.
- Use feature matching to find corresponding points, which in turn are used to find the fundamental matrix

8 Point Algorithm

Fundamental Matrix, F (3x3) has 8 unknowns (instead of 9 because we define it upto a scale, hence removing one unknown)

At least 8 pairs of corresponding points → 8 P A → Fundamental Matrix → Rotation, translation

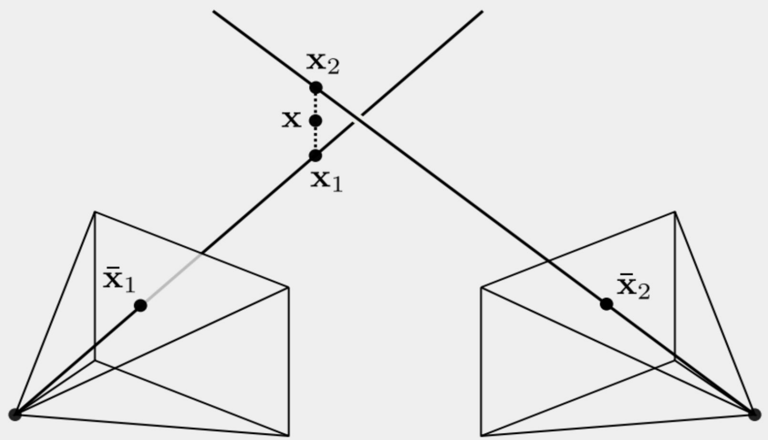
Based on linear Algebra and uses Singular value Decomposition (SVD)

$$X_1^T F X_2 = 0$$

Triangulation

- Triangulation: Process of determining the 3D location of a point in space by measuring its projections in at least two different 2D views.

- Projection matrix: A mathematical matrix that transforms 3D points into 2D points in an image plane.



Given a set of (noisy) matched points on image plane: x and projection matrix: P, we can find the 3D coordinate X as

$$X = Px$$

Use multiple points to triangulate for the exact 3D point

Visual Odometry Vs Localization

Localization: We need pre-existing knowledge of the environment such as the map

Visual Odometry: We estimate camera motion from camera images

Methods for Visual Odometry

Methods for Visual Odometry

Initial Years

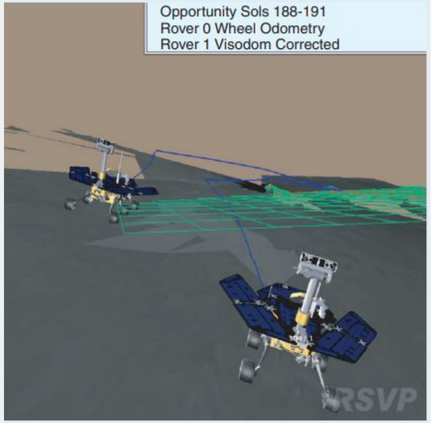
Stanford Cart (with sliding cam)
https://web.stanford.edu/~learnest/sail/oldcart.html

Moravec, “Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover”, Ph. D thesis, Stanford University, 1980

Introduced the motion estimation pipeline + corner detector

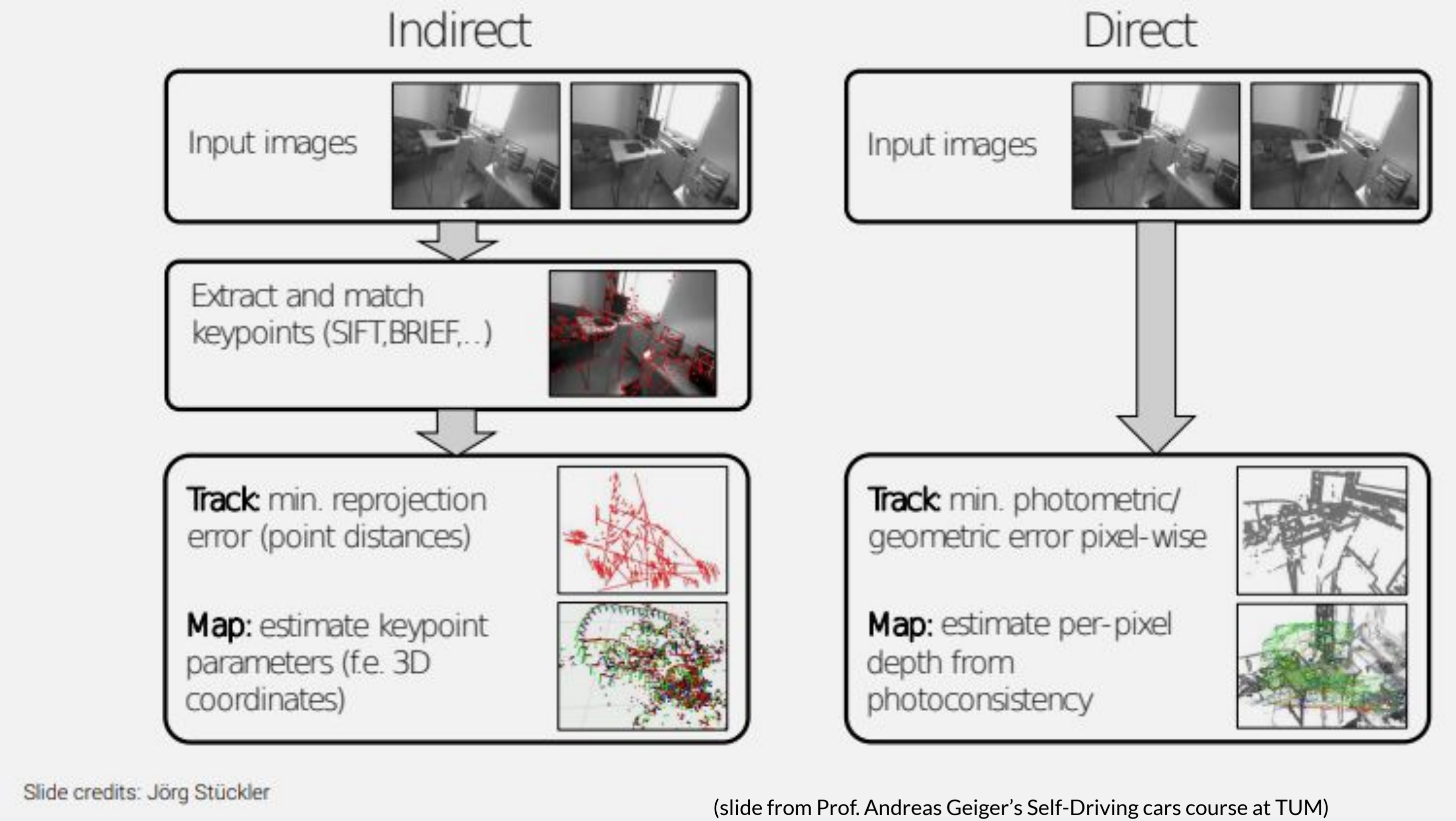
Major initial research in VO was driven by NASA/JPL for the 2004 Mars Exploration Rover mission

A long time ago, at a university far far away…



https://robotics.jpl.nasa.gov/media/documents/vo_ras.pdf

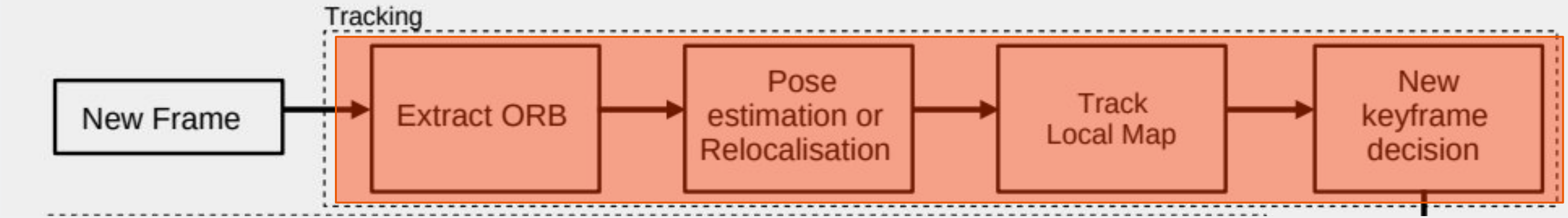
Now, back to the present day



(slide from Prof. Andreas Geiger's Self-Driving cars course at TUM)

Methods for Visual Odometry - Pre-DL

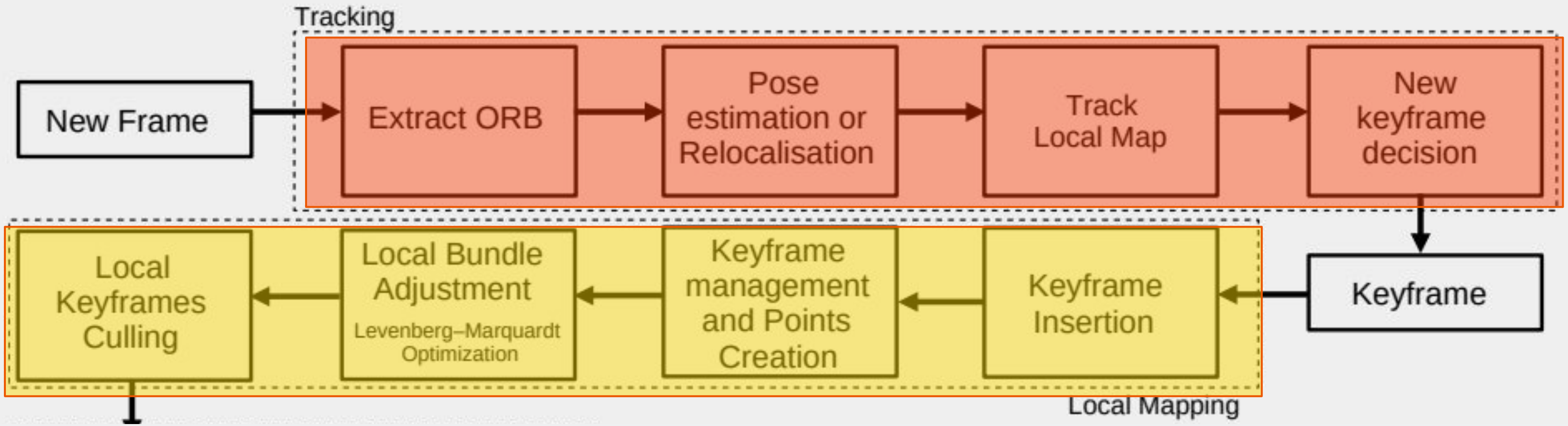
ORB-SLAM2 – Mur-Artal et al., IEEE Trans. On Robotics, 2017



Core Idea: The traditional feature-based pipeline with several optimizations

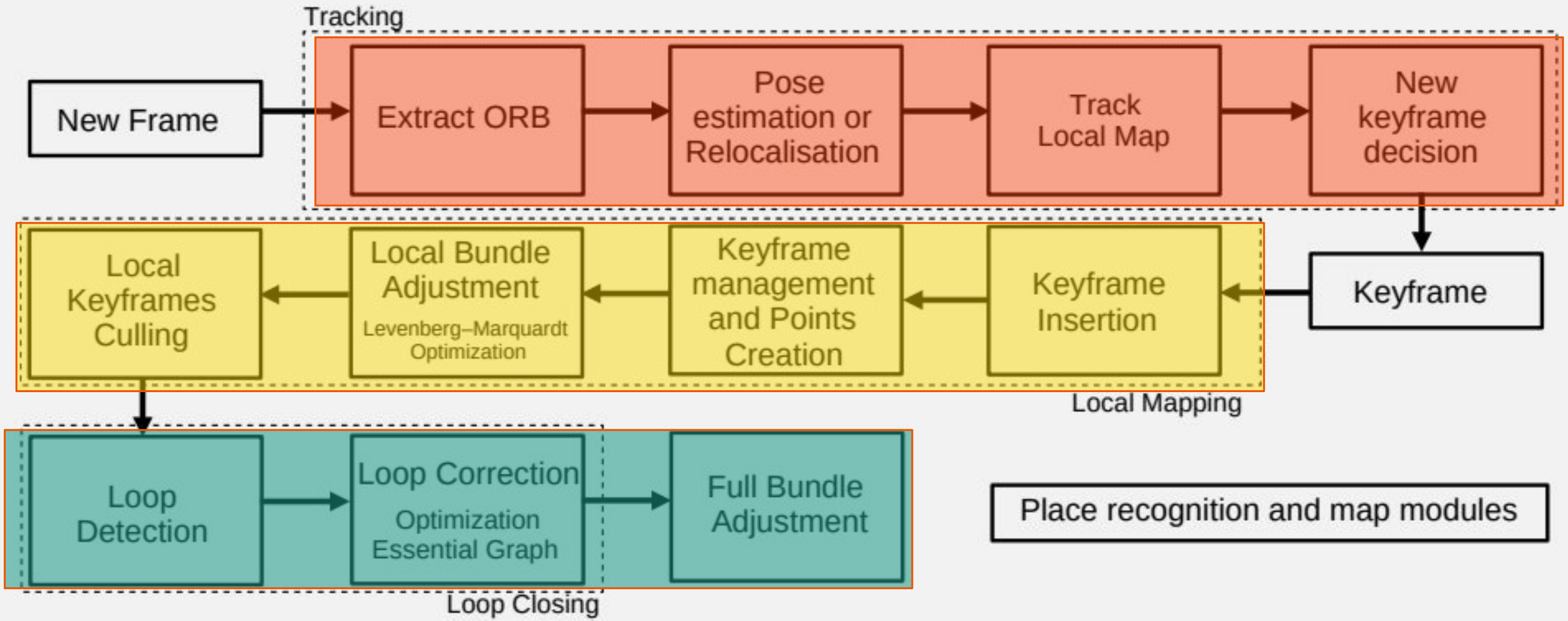
Methods for Visual Odometry - Pre-DL

ORB-SLAM2 – Mur-Artal et al., IEEE Trans. On Robotics, 2017



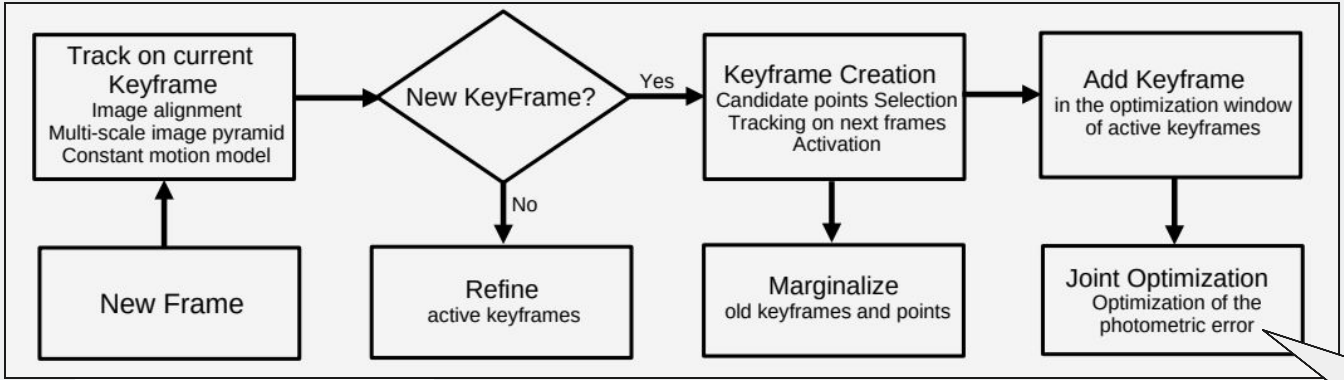
Methods for Visual Odometry - Pre-DL

ORB-SLAM2 – Mur-Artal et al., IEEE Trans. On Robotics, 2017



Methods for Visual Odometry - Pre-DL

Direct Sparse Odometry (DSO) - Engel et al., IEEE TAPMI 2018



Source: https://doi.org/10.3390/robotics11010024

Photometric error = weighted sum of squared distances over a neighborhood of pixels,
where weights = f(inv. depth, camera intrinsics, pose, brightness transfer)

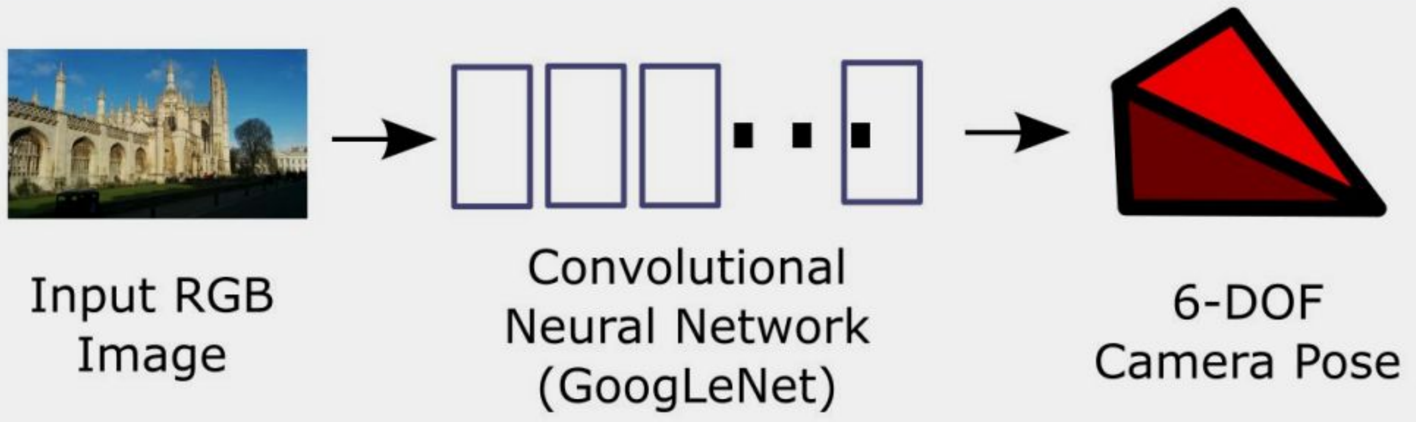
Now, let's look at Deep Learning based methods

Methods for Visual Odometry - using Deep Learning

PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization - Kendall et al., ICCV 2015

Softmax classifiers replaced with regressors - Fully connected layers output 7D vector (3D position + 4D quaternion) instead of softmax.

Extra 2048-d fully connected layer before final regressor.



Methods for Visual Odometry - using Deep Learning

DeepVO - Wang et al., ICRA 2017

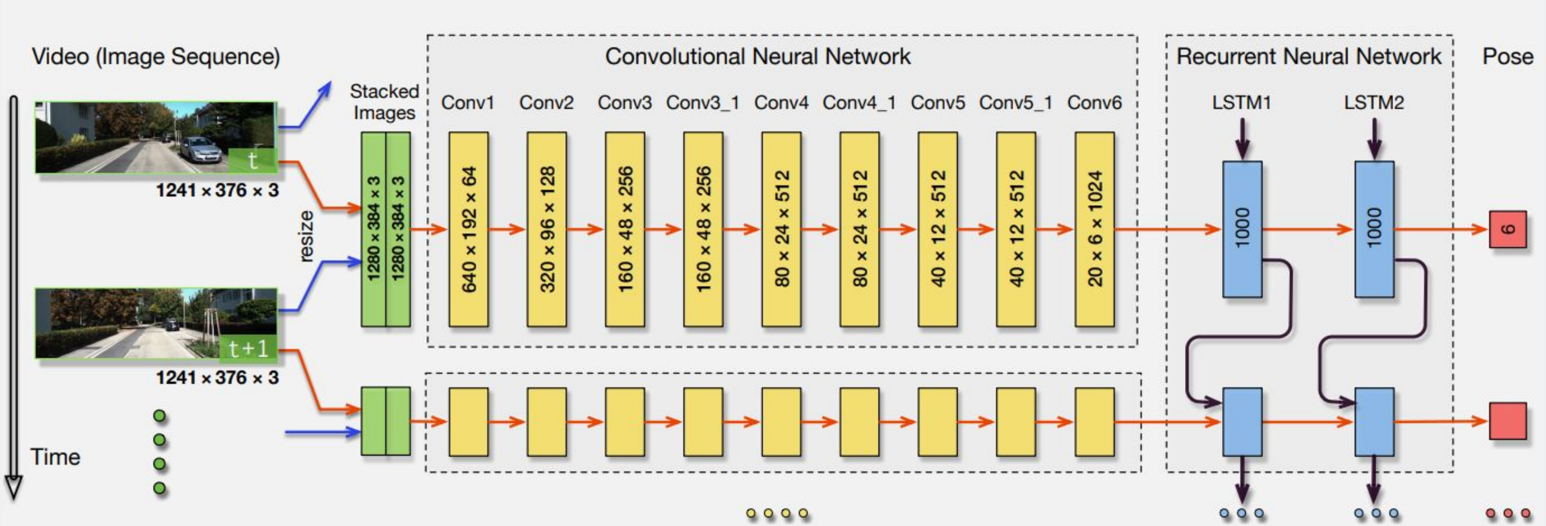
Images Sequence → ConvNet → RNN with LSTM → Camera Pose

Core Idea: CNNs + RNNs for end-to-end **supervised** learning of VO

Methods for Visual Odometry - using Deep Learning

DeepVO - Wang et al., ICRA 2017

Essentially - extract features using CNN, track features using RNN

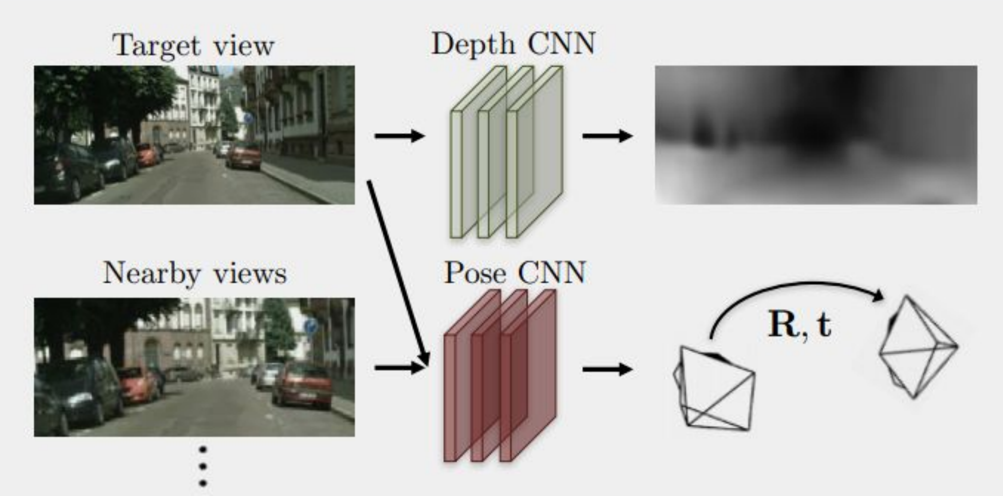


Architecture

Methods for Visual Odometry - using Deep Learning

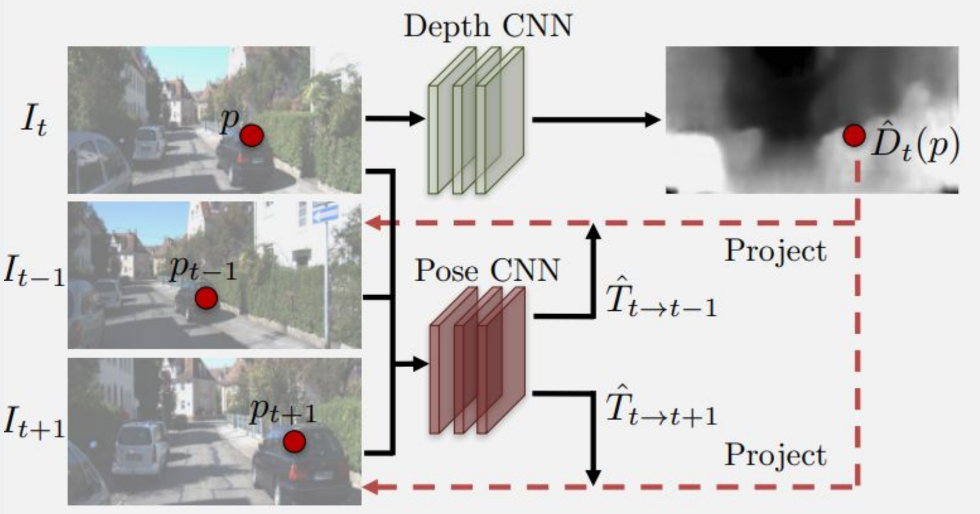
SfMLearner: Unsupervised Learning of Depth and Ego-Motion from Video – Zhou et al., CVPR 2017

Core Idea: Compute depth and pose using CNNs, use outputs to warp input images for loss



Methods for Visual Odometry - using Deep Learning

SfMLearner: Unsupervised Learning of Depth and Ego-Motion from Video – Zhou et al., CVPR 2017



Architecture

Most recent works in VO have similar architecture !

Questions?

Towards Better Generalization: Joint Depth-Pose Learning without PoseNet

Published at: **CVPR 2020**

Authors:

Wang Zhao, Shaohui Liu, Yezhi Shu, Yong-Jin Liu
(Tsinghua University)

Towards Better Generalization: Joint Depth-Pose Learning without PoseNet

Wang Zhao Shaohui Liu Yezhi Shu Yong-Jin Liu*
Department of Computer Science and Technology, Tsinghua University, Beijing, China
zhao-w19@mails.tsinghua.edu.cn, blueber2y@gmail.com,
shuyz19@mails.tsinghua.edu.cn, liuyongjin@tsinghua.edu.cn

Abstract

In this work, we tackle the essential problem of scale inconsistency for self-supervised joint depth-pose learning. Most existing methods assume that a consistent scale of depth and pose can be learned across all input samples, which makes the learning problem harder, resulting in degraded performance and limited generalization in indoor environments and long-sequence visual odometry application. To address this issue, we propose a novel system that explicitly disentangles scale from the network estimation. Instead of relying on PoseNet architecture, our method recovers relative pose by directly solving fundamental matrix from dense optical flow correspondence and makes use of a two-view triangulation module to recover an up-to-scale 3D structure. Then, we align the scale of the depth prediction with the triangulated point cloud and use the transformed depth map for depth error computation and dense reprojection check. Our whole system can be jointly trained end-to-end. Extensive experiments show that our system not only reaches state-of-the-art performance on KITTI depth and flow estimation, but also significantly improves the generalization ability of existing self-supervised depth-pose learning methods under a variety of challenging scenarios, and achieves state-of-the-art results among self-supervised learning-based methods on KITTI Odometry and NYUv2 dataset. Furthermore, we present some interesting findings on the limitation of PoseNet-based relative pose estimation methods in terms of generalization ability. Code is available at https://github.com/B1ueber2y/TrianFlow.

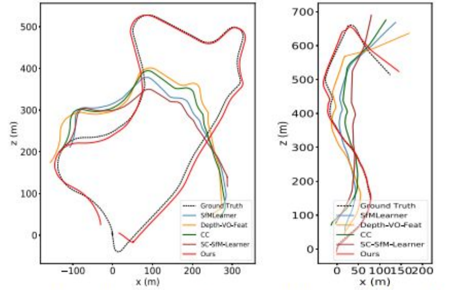


Figure 1. Visual odometry results on sampled sequence 09 and 10 from KITTI Odometry dataset. We sample the original sequences with large stride (stride=3) to simulate fast camera ego-motion that is unseen during training. Surprisingly, all tested PoseNet-based methods get similar failure on trajectory estimation under this challenging scenario. Our system significantly improves the generalization ability and robustness and still works reasonably well on both sequences. See more discussions in Sec 4.4.

on the golden rule of feature correspondence and multi-view geometry, a recent trend of deep learning based methods [42, 15, 66] try to jointly learn the prediction of monocular depth and ego-motion in a self-supervised manner, aiming to make use of the great learning ability of deep networks to learn geometric priors from large amount of training data.

The key to those self-supervised learning methods is to build a task consistency for training separated CNN networks, where depth and pose predictions are jointly constrained by depth reprojection and image reconstruction error. While achieving fairly good results, most exist-

The big question - What ?

Let's break it down first:

Depth of every pixel in the scene

Pioneered 6DOF camera regression using ConvNets [1]

Joint Depth-Pose Learning without PoseNet

Learning both networks together (self-supervised)

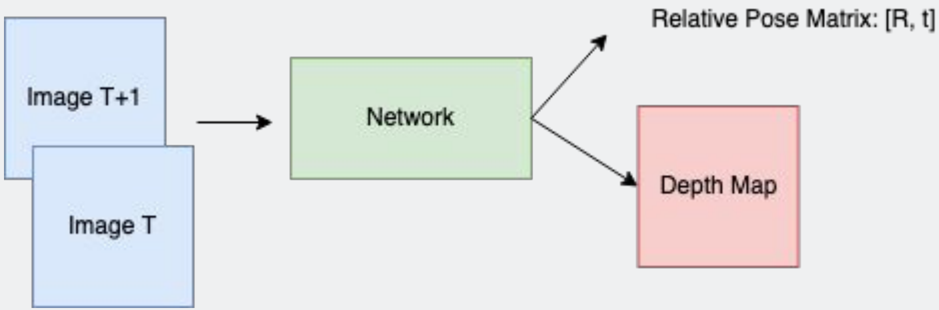
Relative pose of camera for each image

Distilled version:
Learning Monocular Depth Estimation and Camera Pose together, in a self-supervised fashion, without using a PoseNet-style network

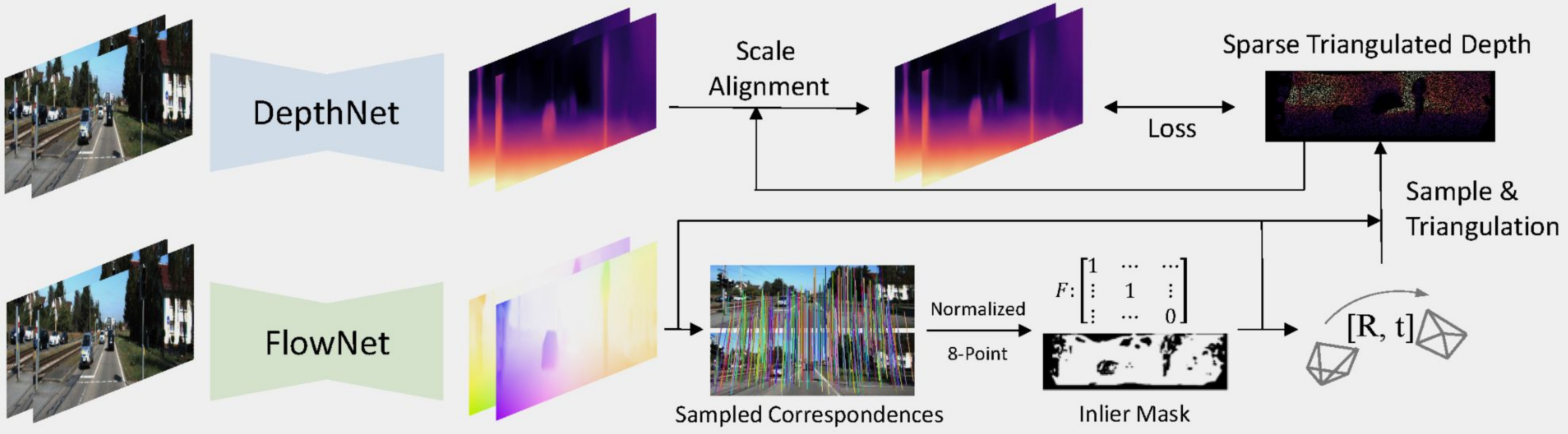
[1] PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization, Kendall et al., **ICCV 2015**

Overview

Sequence of
Monocular
Images



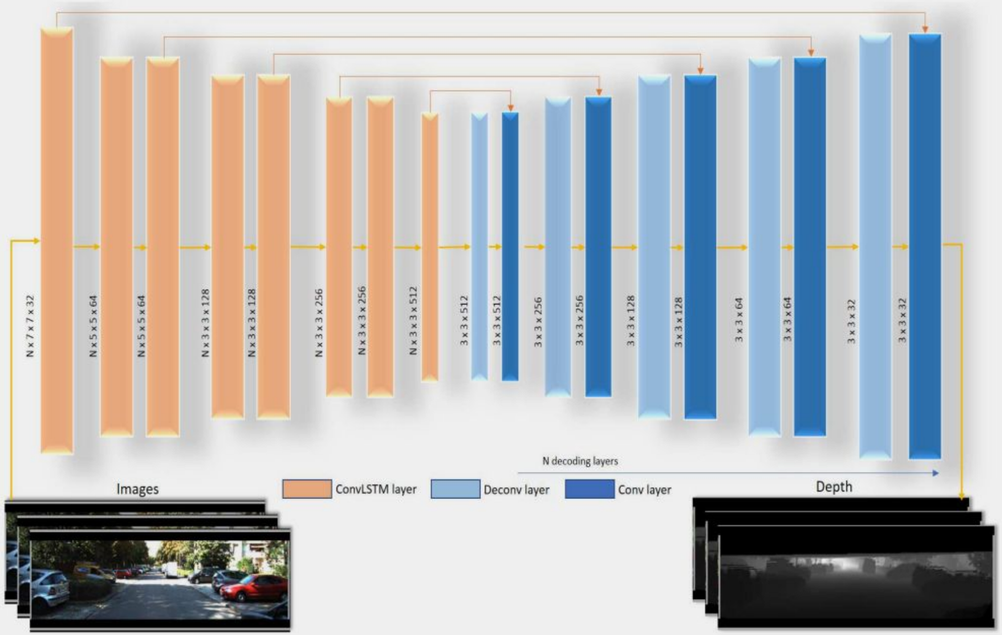
Overview



DepthNet
Scale
Alignment
Loss
Sparse Triangulated Depth
Sample &
Triangulation
FlowNet
Sampled Correspondences
Normalized
8-Point
$F: \begin{bmatrix} 1 & \cdots & \cdots \\ \vdots & 1 & \vdots \\ \vdots & \cdots & 0 \end{bmatrix}$
Inlier Mask
[R, t]

DepthNet

- ConvLSTM-based architecture
- Depth prediction from video sequences
- Captures spatio-temporal information
- Preserves spatial correlations better than traditional LSTM
- Effective in extrapolating depth maps for future or unseen image frames

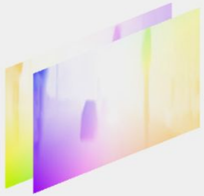


Note: The Depthnet that is used in this paper is not the actual depthnet https://arxiv.org/abs/1806.01260

FlowNet

Displacement of Pixels from one frame to the next

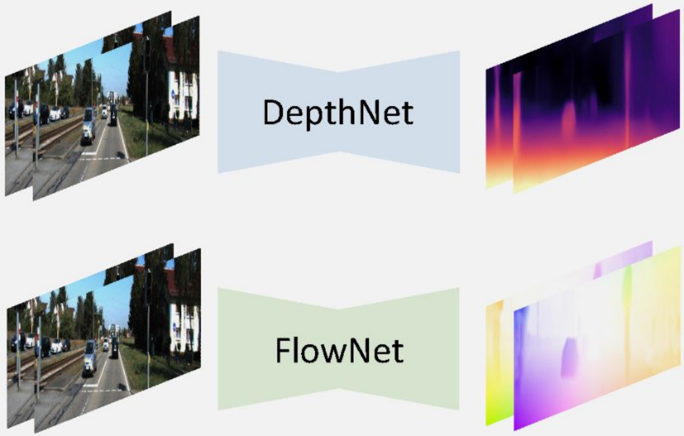
FlowNET



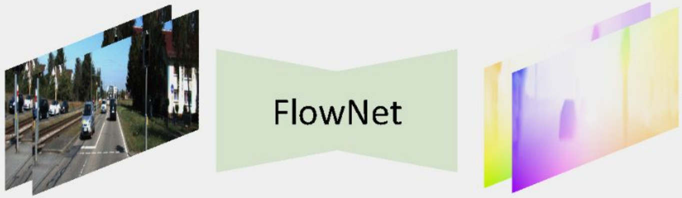
Occlusion Mask

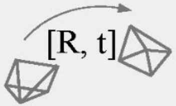
Backward-Forward Score

Network

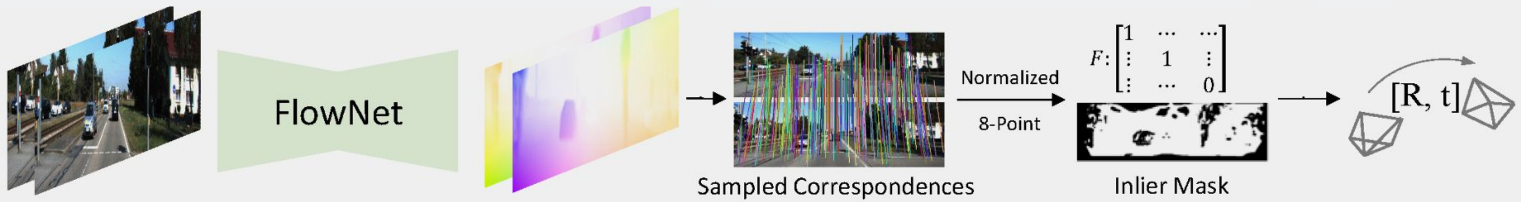


Network





Network


FlowNet
Normalized
8-Point
$F: \begin{bmatrix} 1 & \cdots & \cdots \\ \vdots & 1 & \vdots \\ \vdots & \cdots & 0 \end{bmatrix}$
[R, t]
Sampled Correspondences
Inlier Mask

Recap - 8 Point Algorithm

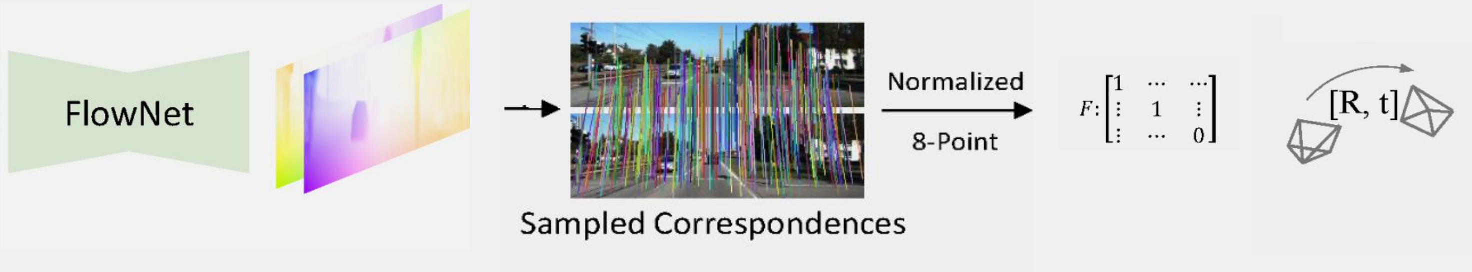
Correspondences come from Optical Flow (FlowNet)!

Fundamental Matrix, F (3x3) has 8 unknowns (instead of 9 because we define it upto a scale, hence removing one unknown)

At least 8 pairs of Corresponding points → 8 P A → Fundamental Matrix → Rotation, translation

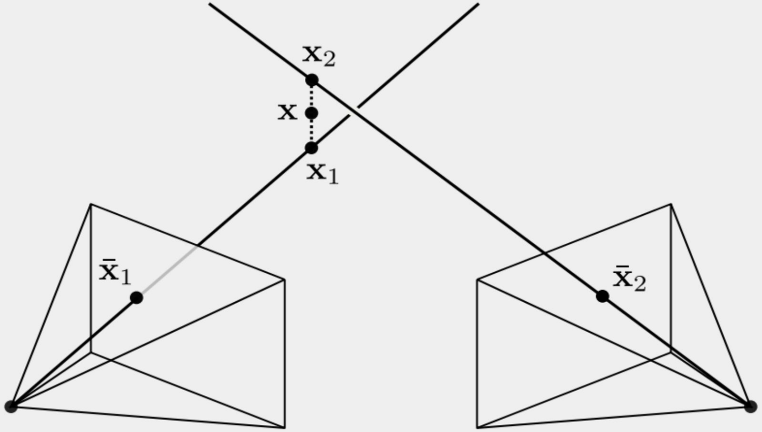
Based on linear Algebra and uses Singular value Decomposition (SVD)

Fundamental Matrix



Recap - Triangulation

- Triangulation: Process of determining the 3D location of a point in space by measuring its projections in at least two different 2D views.
- Projection matrix: A mathematical matrix that transforms 3D points into 2D points in an image plane.



Given a set of (noisy) matched points on image plane: x and projection matrix: P, we can find the 3D coordinate X as

```
X = Px
```

Use multiple points to triangulate for the exact 3D point

Triangulation

But how do we choose which points to triangulate with?

Ans: Inlier Scores Map

Triangulation

But how do we choose which points to triangulate with?
Ans: Inlier Scores Map

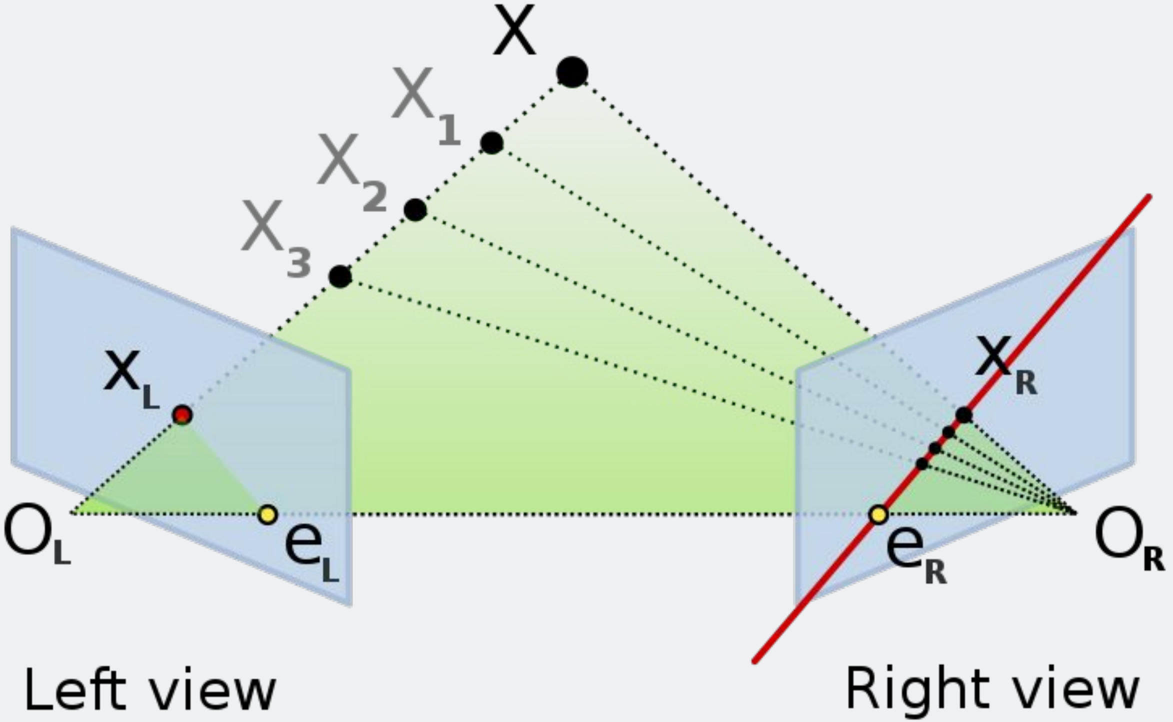
Inlier Scores Map: Fundamental Matrix is used to find correspondences for triangulation

$$F: \begin{bmatrix} 1 & \cdots & \cdots \\ \vdots & 1 & \vdots \\ \vdots & \cdots & 0 \end{bmatrix} \longrightarrow$$

Inlier Mask

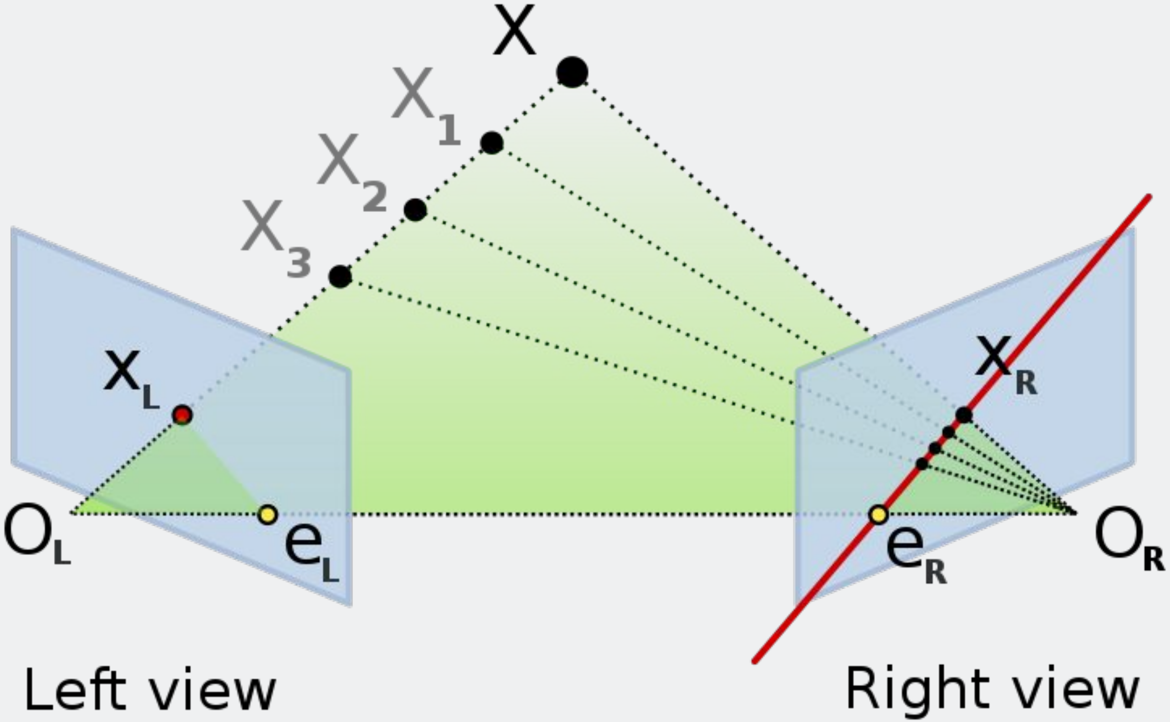
Recap - Epipolar Geometry

- 1D Search Space!



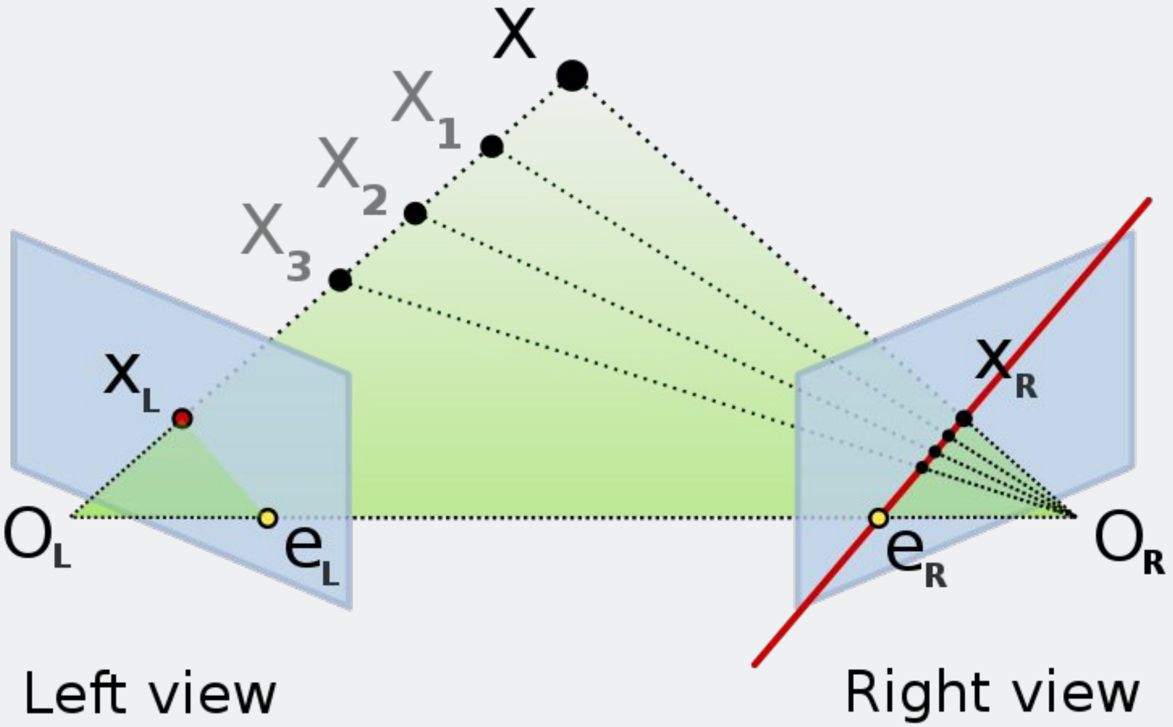
Recap - Epipolar Geometry

- 1D Search Space!
- Find the best match for the given point



Recap - Epipolar Geometry

- 1D Search Space!
- Find the best match for the given point
- Get a list of Correspondences in the form of an inlier mask



Explain Paper - Fundamental Matrix estimation

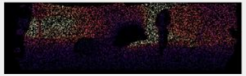
Inlier Mask

Occlusion Mask

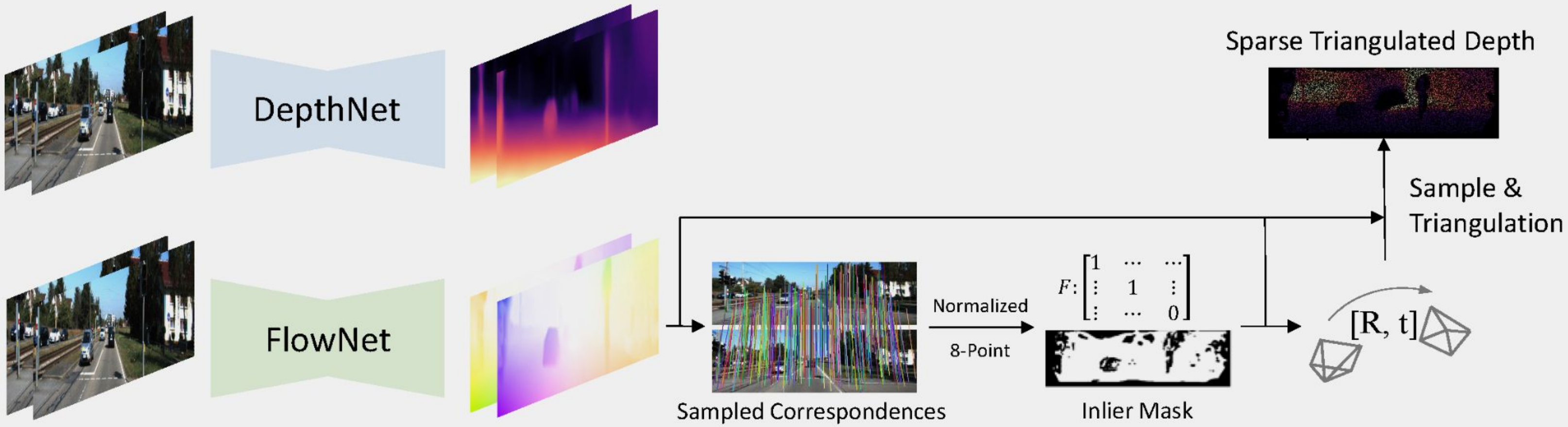
Backward-Forward Score

Mid-Point Triangulation

Sparse Triangulated Depth

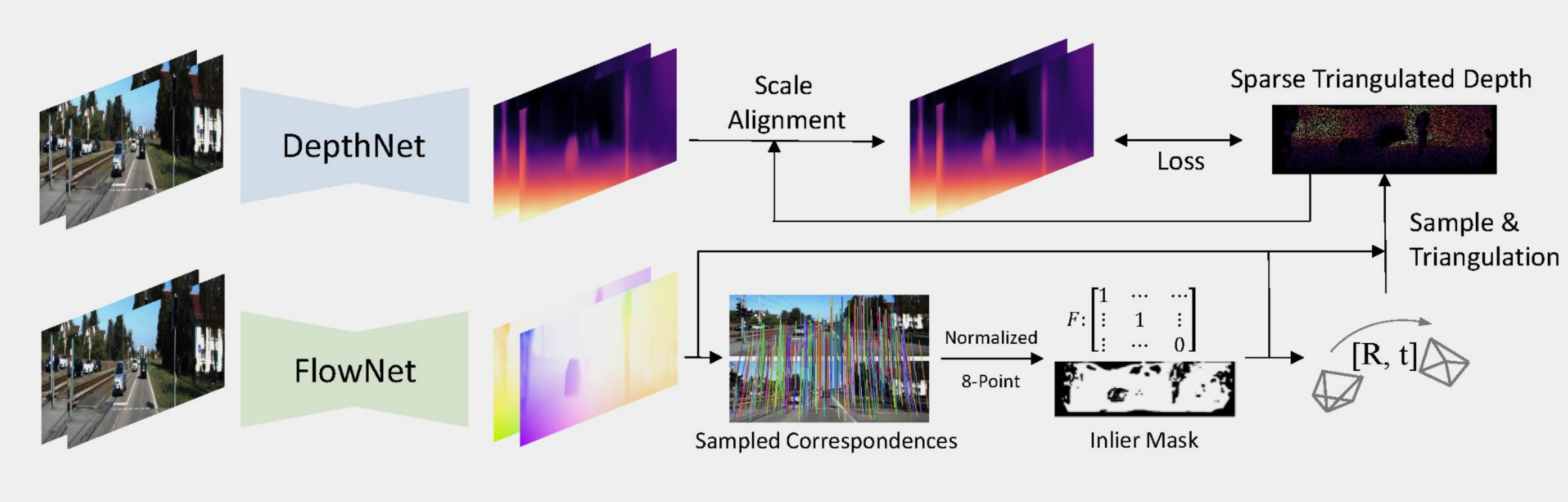


Overview



DepthNet
FlowNet
Sparse Triangulated Depth
Sample &
Triangulation
Normalized
8-Point
$F: \begin{bmatrix} 1 & \cdots & \cdots \\ \vdots & 1 & \vdots \\ \vdots & \cdots & 0 \end{bmatrix}$
[R, t]
Sampled Correspondences
Inlier Mask

Overview



DepthNet
FlowNet
Scale
Alignment
Loss
Sparse Triangulated Depth
Sample &
Triangulation
Sampled Correspondences
Normalized
8-Point
$F: \begin{bmatrix} 1 & \cdots & \cdots \\ \vdots & 1 & \vdots \\ \vdots & \cdots & 0 \end{bmatrix}$
Inlier Mask
[R, t]

Scale Alignment

So why are we doing all this?
→ To find a Depth Map with a good scale

Scale Alignment

So why are we doing all this?
$\longrightarrow$ To find a Depth Map with a good scale

We can now obtain the depth from the 3D reconstruction
$\longrightarrow$ D_{tri} (Pseudo-Ground truth Depth)

Scale Alignment

We align the Predicted Depth Map with the triangulated 3D structure

We align it by multiplying the ***Depth Map*** with a scale **s**

$$\boldsymbol{D_t = sD}$$

Scale Alignment

Training and Finding '**s**':

- Minimize the error between *Transformed Depth* and *Pseudo Ground Truth Depth*

$$L_d = \left(\frac{D_{tri} - D_t}{D_{tri}}\right)^2$$

Where, $D_t = sD$

Loss Functions

$$L = w_1 L_f + w_2 L_d + w_3 L_p + w_4 L_s$$

Flow

Depth

Reprojection

Depth smoothness

Questions?

KITTI dataset

- The KITTI Odometry dataset is a widely-used benchmark for evaluating visual odometry.
- 22 sequences of stereo image pairs of which 11 ground truth sequences of image pairs.

TUM RGBD DATASET

- Consists of several sequences recorded in different ***indoor*** environments - offices, hallways, labs, etc.
- 14 different sequences recorded in various indoor environments **with large textureless surfaces**
- Useful for **more complex camera ego-motions**.

Dataset: https://cvg.cit.tum.de/data/datasets/rgbd-dataset

Summary

Localization: Determining a robot's position and orientation within its environment

- Pre-DL methods with LiDAR/range data: PF, KF, EKF
- Visual Localization and Pre-DL methods. We looked at pioneer works by Andrew Davison's in SLAM

Visual Odometry: Estimating a robot's motion using visual input from cameras

- Pre-DL methods with visual data
- Difference between odometry and localization
- DL Methods used in Visual Odometry

Our paper: Mainly focus on Joint Depth-Pose Learning without PoseNet

Thank You!