Vision Algorithms For Mobile Robotics

GianAndrea Müller

January 16, 2019

Contents

1 Definitions

Definition 1. ***Computer vision*** *is defined as automatic extraction of meaningful information from images and videos of either semantic or geometric nature.*

Definition 2. ***Structure from Motion (SFM)*** *is more general than VO and tackles the problem of 3D reconstruction and 6DOF pose estimation from unordered images sets.*

Definition 3. ***Visual SLAM*** *is simultaneous localization and mapping. It focuses on a globally consistent estimation by performing loop detection and graph optimization in connection with visual odometry. Worse performance, better accuracy then VO.*

$$\boxed{\textit{Visual SLAM} = \textit{VO} + \textit{Loop detection} + \textit{graph optimization}}$$

Definition 4. ***Visual Odometry (VO)*** *is the process of incrementally estimating the pose of the vehicle by examining the changes that motion induces on the images of its onboard cameras in real time.*

Definition 5. ***Bundle adjustment*** *is the problem of refining a visual reconstruction to produce jointly optimal 3D structure and viewing parameter (camera pose and/or calibration) estimates. Optimal means that the parameter estimates are found by minimizing some cost function that quantifies the model fitting error, and jointly that the solution is simultaneously optimal with respect to both structure and camera variations. The name refers to the 'bundles' of light rays leaving each 3D feature and converging on each camera centre, which are 'adjusted' optimally with respect to both feature and camera positions.*

Definition 6. ***Photogrammetry*** *is the science of making measurements from photographs, especially for recovering the exact positions of surface points.*

Definition 7. ***Pose-graph****: a network of nodes and edges where the nodes are robot poses and edges are constraints between poses.*

Definition 8. ***Loop closure****: a constraint between the a recent robot pose and a past pose when the robot revisits a previously visited location. Loop closure is highly sensitive to the current estimate of where the robot is. If your current estimate is bad you may not realize you are visiting a previously visited location! There are global loop closure approaches which try to match what the robot sees to everything seen in the past in order to find a closure, such approaches may be computationally expensive.*

Definition 9. ***LIDAR*** *is used as an acronym for light detection and ranging and describes the process of measuring the distance to a target by illuminating the target with pulsed laser light and measuring the reflected pulses with a sensor.*

Definition 10. ***Subpixel accuracy*** *is description of a point of interest not in terms of its integer pixel position but in terms of where it lies exactly between pixels.*

Definition 11. ***Feature based methods*** *extract a sparse set of features, match them in successive frames, robustly recover camera pose and structure and then refine both through reprojection error minimization.*

Definition 12. *In contrast to feature-based methods* ***direct methods*** *estimate structure and motion directly from intensity values in the image. Their advantage is the full exploitation of the available information which makes for greater robustness. On the other hand the evaluation of the photometric error more expensive than the calculation of the reprojection error, but the time saved when not finding and matching features makes up for that.*

2 Research Collection

2.1 Direct Image Alignment

LSD-SLAM (2014 TUM, DE) *Large-Scale Direct Monocular SLAM*

SVO (2014 UZH, CH) *Fast Semi-Direct Monocular Visual Odometry*

DTAM (2011 ICL, UK) *Dense Tracking and Mapping in Real-Time*

2.2 Features

Harris *A combined corner and edge detector*

CANNY (1986 IEEE) *Canny edge detection*

Scale-space theory (1994 CVAP) *Scale-space theory: A basic tool for analysing structures a different scales. (Basics for SIFT)*

SIFT (2004 IJCV) *Distinctive Image Features from Scale-Invariant Keypoints*

SIFT Object Recognition (2004) *Object Recognition from Local Scale-Invariant Features*

SURF (2006) *Speeded-Up Robuts Features*

- *Similar to SIFT, but faster with shorter descriptors*
- *Approximated computation for detection and descriptor*

FAST detector (2005) *Fusing Points and Lines for High Performance Tracking*

- *Studies intensity of pixels on circle around candidate pixel C*
- *C is a FAST corner if a set of N contiguous pixels on the circle are either all brighter or darker than the intensity of C + threshold.*
- *Very fast detector, about 100 Megapixels/s*

BRIEF descriptor (2010) *Binary Robust Independent Elementary Features*

- *Goal: High speed*
- *Detection based on FAST*
- *The descriptor pattern is generated randomly (or by machine learning) only once and then used for all patches.*

\+ *Binary descriptor allows very fast Hamming distance matching.*

- *Not scale/rotation invariant.*

ORB descriptor (2011) *Oriented FAST and Rotated BRIEF*

- *Keypoint detector based on FAST*
- *BRIEF descriptors are steered according to keypoint orientation to provide rotation invariance*
- *Good binary features are learned by minimizing the correlation on a set of training patches*

BRISK descriptor (2011) *Binary Robust Invariant Scalable Keypoints*

- *Detect corners in scale-space using FAST*
- *Rotation and scale invariant*
- *Compare intensity in patter-defined regions*
- *Approximately 10 times faster than SURF for detection and description*
- *Slower than BRIEF but scale and rotation invariant*

FREAK descriptor (2012) *Fast Retina Keypoint*

- *Rotation and Scale invariant*
- *Fast compact and robust descriptor*
- *Sampling patterns based on human retina*
- *Faster to compute, less memory and more robust than SIFT, SURF or BRISK*

2.3 Multi View Geometry

Stereo Rectification (2000) *A Compact Algorithm for Rectification of Stereo Pairs*
PTAM (2007 ISMAR) *Parallel Tracking and Mapping for Small AR Workspaces*
DSO (2017 PAMI) *Direct Sparse Odometry*
ORB-SLAM (2015 TRO) *ORB-SLAM: a Versatile and Accurate Monocular SLAM System*

2.4 Dense 3D Reconstruction

REMODE (2014 ICRA) *Probabilistic, Monocular Dense Reconstruction in Real Time*

2.5 Template Tracking

Lucas-Kanade Tracker (1981) *An Iterative Image Registration Technique with an Application to Stereo Vision*
Lucas-Kanade Tracker (2004 IJCV) *Lucas-Kanade 20 Years On: A Unifying Framework*

2.6 Visual Inertial Fusion

Ultimate SLAM (2018 RAL) *Ultimate SLAM? Combining Events, Images, and IMU for Robust Visual SLAM in HDR and High-Speed Scenarios*

2.6.1 Closed Form

Closed-Form Solutions for Attitude, Speed, Absolute Scale and Bias Determination (2012 TOR)

Closed-Form Solution of Visual-Inertial SFM(2014 IJCV)

Simultaneous State Initialization and Gyroscope Bias Calibration in VI Aided Navigation (2017 RAL)

2.6.2 Filters

A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation (2007 ICRA)

Analysis and Improvement of the Consistency of EKF based SLAM(2008 ICRA)

Visual-Inertial Navigation, Mapping and Localization: A Scalable Real-Time Causal Approach (2011 IJRR)

Observability-Constrained Vision-Aided Inertial Navigation (2012 ISER)

Robust Visual Inertial Odometry Using a Direct EKF-Based Approach (2015 IROS)

A Square Root Inverse Filter for Efficient Vision-Aided Inertial Navigation on Mobile Devices (2015 RSS)

Consistency Analysis and Improvement of Vision-Aided Inertial Navigation (2014 TOR)

ROVIO (2013 RSS) *Keyframe-Based Visual-Inertial SLAM Using Nonlinear Optimization*

2.6.3 Fixed-Lag Smoothers

Sliding Window Filter with Application to Planetary Landing (2010 JFT)

Motion Tracking with Fixed-Lag Smoothing (2011 ICRA)

Keyframe-Based Visual-Inertial Odometry Using Nonlinear Optimization (2015 IJRR)

2.6.4 Full-Smoothing Methods

Camera Trajectory Estimation using Inertial Sensor Measurements and SFM Resulst (2001 CVPR)

Motion Estimation from Image and Inertial Measurements (2004)

Airborne Smoothing and Mapping using Vision and Inertial Sensors (2009 ICRA)

Information Fusion in Navigation Systems Via Factor Graph Based Incremental Smoothing (2013 RAS)

IMU Preintegration on Manifold for Efficient Visual-Inertial Maximum-a-Posteriori Estimation (2015 RSS)

A Spline-Based Trajectory Representation for Sensor Fusion and Rolling Shutter Cameras (2015 IJCV)

Autonomous, Vision-Based Flight and Live Dense 3D Mapping with Quadrotor Micro Aerial Vehicle (2016 TRO)

SVO, Semidirect Visual Odometry for Monocular and Multicamera Systems (2017 TRO)

2.7 Even Based Vision

Active Exposure Control for Robust Visual Odometry in HDR Environments (2017 ICRA)

Low-latency localization by Active LED Markers tracking using a DVS (2013 IROS)

Low-Latency Event-Based Visual Odometry (2014 ICRA)

Event-based, 6-DOF Pose Tracking for High-Speed Maneuvers (2014 IROS)

Event-based, 6-DOF Camera Tracking from Photometric Depth Maps (2018 TPAMI)

ESIM (2018 CORL) *Open Event Camera Simulator*

Real-time Visual-Inertial Odometry for Event Cameras using Keyframe-based Nonlinear Optimization (2017 BMVC)

A 240 x 180 130 dB 3 us Latency Global Shutter Spatiotemporal Vision Sensor (2014 JSSC)

Asynchronous, Photometric Feature Tracking using Events and Frames (2018 ECCV)

The Event-Camera Dataset and Simulator (2017 IJRR) *Event-based data for pose estimation, visual odometry, and SLAM*

Accurate Angular Velocity Estimation With an Event Camera (2017 RAL)

EVO (2017 RAL) *A Geometric Approach to Event-Based 6-DOF Parallel Tracking and Mapping in Real Time*

EMVS (2016 BMVC) *Event-based Multi-View Stereo*

Fast Event-Based Corner Detection (2017 BMVC)

Simultaneous Mosaicing and Tracking with an Event Camera (2014 BMVC)

Interacting Maps for Fast Visual Interpretation (2011 IJCNN)

3 Basics VO

Advantages

- More accurate than wheel odometry.
- Not affected by wheel slippage.
- Can be used complementary to
  - wheel encoders
  - GPS
  - inertial measurement units
  - laser odometry

Assumptions

- sufficient illumination
- dominance of static scene
- enough texture
- sufficient scene overlap

3.1 Working Principle

$$\boxed{T_k = \begin{bmatrix} R_{k,k-1} & t_{k,k-1} \\ 0 & 1 \end{bmatrix}} \text{ Homogeneous transformation matrix}$$

Working Principle

1. Compute the relative motion T_k from images I_{k-1} to I_k.
2. Concatenate them to recover the full trajectory:
$$C_n = C_{n-1} T_n$$
3. An optimization over the last m poses can be done to refine the trajectory locally (Pose-graph or Bundle adjustment).

3.2 How to find T_k

In general:

$$\boxed{T_k = \arg\min_T \iint_{\bar{B}} \rho \left[I_k \left(\pi(\mathbf{T} \cdot \pi^{-1}(\mathbf{u}, d_\mathbf{u})) \right) - I_{k-1}(\mathbf{u}) \right] d\mathbf{u}}$$

Direct image alignment

$$\boxed{T_{k,k-1} = \arg\min_T \sum_i ||I_k(\mathbf{u}'_i) - I_{k-1}(u_i)||^2_\sigma}$$

minimizes the per-pixel intensity difference.
This can be done at different resolutions: dense, semi-dense or sparse.

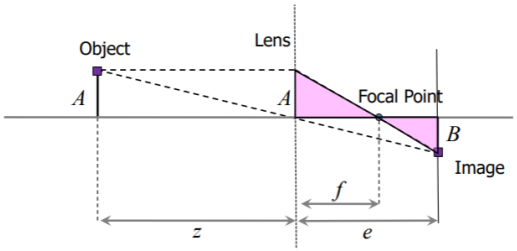
4 Image Formation

4.1 Pinhole Camera

Tradeoff between narrowness of the hole and diffraction issues. Best choice: $\sim 0.35\,\text{mm}$.

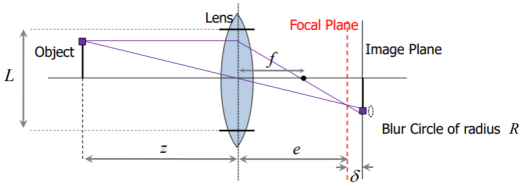
4.2 Converging Lens

- Rays passing through the optical center are not deviated.
- All light rays that are parallel to the optical axis converge in the focal point.



$\boxed{\frac{1}{f} = \frac{1}{z} + \frac{1}{e}}$ Thins lens equation

f	focal length	[m]
z	distance between image and lens	[m]
e	distance between object and lens	[m]



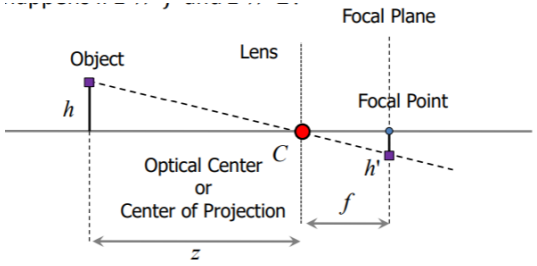
$\boxed{R = \frac{L\delta}{2e}}$ Blur Circle Formula

If the blur circle radius is smaller than a single pixel on the sensor, distance can no longer be calculated!

R	blur radius	[m]
L	aperture	[m]
δ	distance from the film	[m]

4.3 Pin-hole approximation

If $z \gg f$ and $z \gg L$:



Thus when focussing on objects at an infinite distance, the image plane lies in the focal plane.

$$\boxed{f \approx e}$$

Do not confuse the center of projection and the focal point!

$\boxed{\frac{h'}{h} = \frac{f}{z} \Rightarrow h' = \frac{f}{z}h}$ Relation between image and object

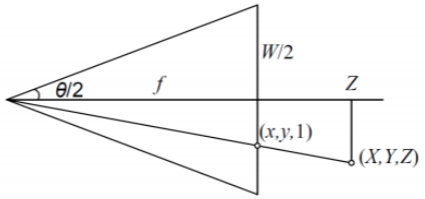
$\boxed{\begin{aligned} x' &= f\frac{x_c}{z_c} \\ y' &= f\frac{y_c}{z_c} \end{aligned}}$ Perspective Camera Equations

4.4 Perspective Projection

Straight lines remain straight, angles are not preserved!

All parallel lines in reality intersect in the vanishing point in the projected picture, except for lines that are parallel to the camera plane, for those the vanishing point is infinitely far. All vanishing points lie on the vanishing lines, parallel planes intersect on the vanishing line.

Definition 13. ***Depth of Field (DOF)*** *is the distance between the nearest and farthest objects in a scene that appear acceptably sharp in the image.*

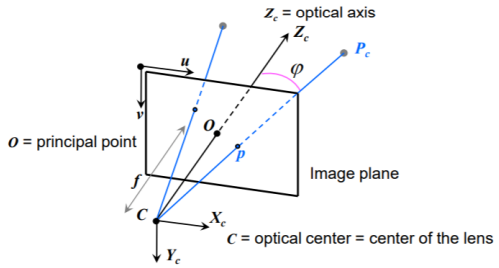


Note that the coordinates of the point on the screen $(x, y, 1)$ are homogeneous 2D coordinates, thus 1 is not the coordinate in z direction.

W	width of the normalized screen	[m]
θ	angle of view	[°]

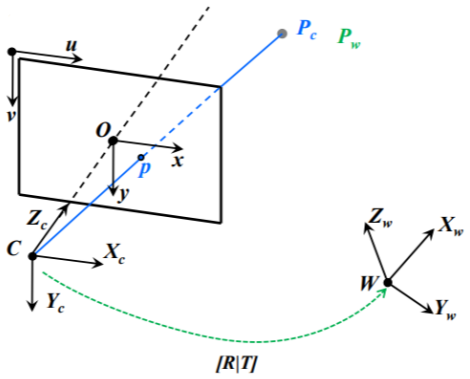
$\boxed{\tan(\frac{\theta}{2}) = \frac{W}{2f} \text{ or } f = \frac{W}{2}\left[\tan(\frac{\theta}{2})\right]^{-1}}$ Relation between field of view and focal length

4.5 Perspective Camera Model



1. Given a point in world coordinates P_w calculate its location in camera coordinates P_c using a homogeneous transformation $T = [R|t]$:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R & t \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$



2. Using similarity of triangles the point in camera frame is projected to the 1 meter image plane to get p in **calibrated** or **normalized coordinates**.

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \frac{X_c}{Z_c} \\ \frac{Y_c}{Z_c} \end{bmatrix}$$

There is a homogeneous form of the same coordinates:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{X_c}{Z_c} \\ \frac{Y_c}{Z_c} \\ \frac{Z_c}{Z_c} \end{bmatrix}$$

3. Optionally apply lens distortion to get to **distorted normalized coordinates** p_d:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = (1 + k_1 r^2 + k_2 r^4) \begin{bmatrix} x \\ y \end{bmatrix}$$

4. To accomplish the projection of the camera coordinates to the Z_c meter image plane and the transformation to pixel coordinates the following equations are used:

$$\tilde{u} = u_0 + k_u x \Rightarrow u = u_0 + \frac{k_u f X_c}{Z_c}$$
$$\tilde{v} = v_0 + k_v y \Rightarrow v = v_0 + \frac{k_v f Y_c}{Z_C}$$
$$\tilde{w} = Z_C$$

$O = (u_0, v_0)$	Image center	$[\mathrm{m}] \in \mathbb{R}^2$
k_u, k_v	scale factors for pixel dimensions	$\left[\frac{pixel}{\mathrm{m}}\right]$
$\alpha_u = k_u f, \alpha_v = k_v f$	focal length in pixels	$[pixel]$

Next the coordinates are normalized by dividing by the scale factor $\lambda = Z_c = \tilde{w}$ which yields the location in pixel coordinates.

$$p = \begin{pmatrix} u \\ v \end{pmatrix} \Rightarrow \tilde{p} = \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

So the equations above can be expressed in matrix form:

$$\begin{bmatrix} \lambda u \\ \lambda v \\ \lambda \end{bmatrix} = \underbrace{\begin{bmatrix} k_u f & 0 & u_0 \\ 0 & k_v f & v_0 \\ 0 & 0 & 1 \end{bmatrix}}_{K} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$$

The K matrix consists of the location of the intersection of the optical axis with the image plane (u_0, v_0) and the transformation to pixel coordinates. After that the normalization is still necessary.

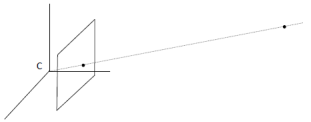
4.5.1 Undistoring and image

1. Define all pixel coordinates of the destination image (undistorted image).
2. Distort those coordinate location.
3. Measure the image intensity of the source image at the calculated locations.
4. Map the measured intensities back to the destination image.

4.6 Pose determination from n Points (PnP)

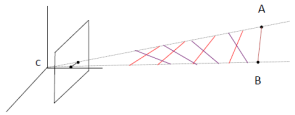
How many points are enough?

1. A single point



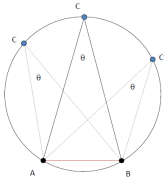
The camera can move along the line of projection. Thus we have infinitely many solutions.

2. Two Points

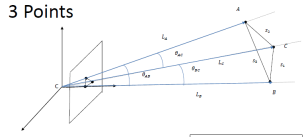


Since we don't know size and orientation of the line between the two points, the camera position is still undetermined.

Differently formulated the knowing of an angle between two points does not fix the location of the camera:



3. Three Points (P3P Problem)



From Carnot's theorem:

$$
\begin{aligned}
s_1^2 &= L_B^2 + L_C^2 - 2L_B L_C \cos(\theta_{BC}) \\
s_2^2 &= L_A^2 + L_C^2 - 2L_A L_C \cos(\theta_{AC}) \\
s_3^2 &= L_A^2 + L_B^2 - 2L_A L_B \cos(\theta_{AB})
\end{aligned}
$$

A system of polynomial equations in n unknowns can have no more solutions than the product of their respective degrees[1]. In this case: 8. Since all terms are constant or quadratic half of the solutions are negative and thus invalid. This leaves us with a total of four valid solutions. **A forth point disambiguates these solutions.**

4.7 Camera Calibration: Direct Linear Transform (DLT)

Estimate **intrinsic** and **extrinsic** parameters.
Find the projection matrix m as

$$M = K[R|T]$$

Which can be used as follows:

$$
\begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} m_1^T \\ m_2^T \\ m_3^T \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}
$$

where m_i^T are the rows of the projection matrix M.
Now assume that $P = \begin{bmatrix} X_w & Y_w & Z_w & 1 \end{bmatrix}^T$
And project the whole equation to pixel coordinates using

$$
\begin{aligned}
u = \frac{\tilde{u}}{\tilde{w}} &= \frac{m_1^T \cdot P}{m_3^T \cdot P} & (m_1^T - u_i m_3^T) \cdot P_i = 0 \\
v = \frac{\tilde{v}}{\tilde{w}} &= \frac{m_2^T \cdot P}{m_3^T \cdot P} & (m_2^T - v_i m_3^T) \cdot P_i = 0
\end{aligned}
$$

By re-arranging and writing in matrix form:

$$
\underbrace{\begin{bmatrix} P_1^T & 0^T & -u_1 P_1^T \\ 0^T & P_1^T & -v_1 P_1^T \\ \vdots & \dots & \vdots \\ P_n^T & 0^T & -u_n P_n^T \\ 0^T & P_n^T & -v_n P_n^T \end{bmatrix}}_{Q} \underbrace{\begin{bmatrix} m_1 \\ m_2 \\ m_3 \end{bmatrix}}_{M} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}
$$

where Q can be written out as

$$
Q = \begin{bmatrix}
X_w^1 & Y_w^1 & Z_w^1 & 1 & 0 & 0 & 0 & 0 & -u_1 X_w^1 & -u_1 Y_w^1 & -u_1 Z_w^1 & -u_1 \\
0 & 0 & 0 & 0 & X_w^1 & Y_w^1 & Z_w^1 & 1 & -v_1 X_w^1 & -v_1 Y_w^1 & -v_1 Z_w^1 & -v_1 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
X_w^n & Y_w^n & Z_w^n & 1 & 0 & 0 & 0 & 0 & -u_n X_w^n & -u_n Y_w^n & -u_n Z_w^n & -u_n \\
0 & 0 & 0 & 0 & X_w^n & Y_w^n & Z_w^n & 1 & -v_n X_w^n & -v_n Y_w^n & -v_n Z_w^n & -v_n
\end{bmatrix}
$$

and M can be written out as

$$
M = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} & m_{21} & m_{22} & m_{23} & m_{24} & m_{31} & m_{32} & m_{33} & m_{34} & m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix}^T
$$

For a minimal solution, the matrix $Q_{(2nx12)}$ should have at least rank 11 in order to have a unique (up to scale), non-trivial solution M. Since each 3D-to-2D correspondence provides 2 independent equations we need at least $5 + \frac{1}{2}$ point correspondences, thus 6.
For an over-determined solution we can minimize the squarred error QM subject to $||M||^2 = 1$.

[1] Intuition: Think of a decoupled system of polynomial equations. Then the solution of one equation can be interpreted as finding zeros of that polynomial. The maximum number of zeros found equals the degree of the polynomial. Thus for multiple equations the number of solutions multiplies. Also for a general case coupling terms may be added.

```
[U,S,V] = svd(Q);
M = V(:,12);
```

Once M is known the intrinsic and extrinsic parameters can be calculated using:

$$M = K(R|T)$$

Degenerate configurations:
There are certain combinations of 3D-2D correspondences which are degenerate and do not deliver additional information:

1. Points lying on a plane and/or along a single line passing through the projection center.
2. Camera and points on a twisted cubic (i.e. smooth curve in 3D space of degree 3) why?

4.7.1 Camera Calibration from 3D objects

Given the M matrix we can recover the extrinsic and intrinsic parameters based on:

$$\begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix} = \begin{bmatrix} \alpha_u r_{11} + u_0 r_{31} & \alpha_u r_{12} + u_0 r_{32} & \alpha_u r_{13} + u_0 r_{33} & \alpha_u t_1 + u_0 t_3 \\ \alpha_v r_{21} + v_0 r_{31} & \alpha_v t_{22} + v_0 r_{32} & \alpha_v r_{23} + v_0 r_{33} & \alpha t_2 + v_0 t_3 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix}$$

To enforce that $R \cdot R^T = I$ we can use QR factorization of M, which decomposes M into a R(orthogonal), T, and upper triangular matrix i.e. K.

A 3D calibration example is $Tsai's1987$:

1. Edge detection
2. Straight line fitting to the detected edges
3. Intersection of the edges to find the corners
4. Using more than 6 points, not all in a plane!

The parameters describing the resulting calibration are:

f_y	f_x/f_y	skew	x_0	y_0	residual
1673.3	1.0063	1.39	379.96	305.78	0.365

- The ratio f_x/f_y is not zero since the pixels were not squares.
- The skew indicates that the pixels were parallelograms.
- Today it can be mostly assumed that $\frac{\alpha_u}{\alpha_v} = 1$ and $K_{12} = 0$, where K_{12} is the skewness factor. If it is non-zero the pixels are parallelograms instead of rectangles.
- The residual is the average reprojection error. Today algorithms are expected to deliver errors about around 0.2.

Definition 14. *The **reprojection error** is computed as the distance (in pixels) between the observed pixel point and the camera-reprojected 3D point.*

4.7.2 Camera Calibration from 2D objects

A 2D calibration example, which in contrast requires the points to lie on a plane is Zhang 1999.

1. Neglect radial distortion.
2. All points lie on a plane $\Rightarrow Z_w = 0$

$$\begin{aligned} \tilde{p} = \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} &= \lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K[R|t] \begin{bmatrix} X_w \\ Y_w \\ 0 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ 0 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & t_1 \\ r_{21} & r_{22} & t_2 \\ r_{31} & r_{32} & t_3 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ 1 \end{bmatrix} \\ &= \underbrace{\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}}_{\text{Homography}} \begin{bmatrix} X_w \\ Y_w \\ 1 \end{bmatrix} \end{aligned}$$

$$\begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} h_1^T \\ h_2^T \\ h_3^T \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ 1 \end{bmatrix}$$

3. Conversion to pixel coordinates yields:

$$u = \frac{\tilde{u}}{\tilde{w}} = \frac{h_1^T \cdot P}{h_3^T \cdot P}$$
$$v = \frac{\tilde{v}}{\tilde{v}} = \frac{h_2^T \cdot P}{h_3^T \cdot P}$$

where $P = (X_w, Y_w, 1)^T$, which leads to the two equations of interest:

$$(h_1^T - u_i h_3^T) \cdot P_i = 0$$
$$(h_2^t - v_i h_3^T) \cdot P_i = 0$$

4. By rearranging

$$\underbrace{\begin{bmatrix} P_1^T & 0^T & -u_1 P_1^T \\ 0^T & P_1^T & -v_1 P_1^T \\ \vdots & \vdots & \vdots \\ P_n^t & 0^T & -u_n P_n^T \\ 0^T & P_n^T & -v_n P_n^T \end{bmatrix}}_{Q \text{ is known}} \underbrace{\begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix}}_{H \text{ is unkown}} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

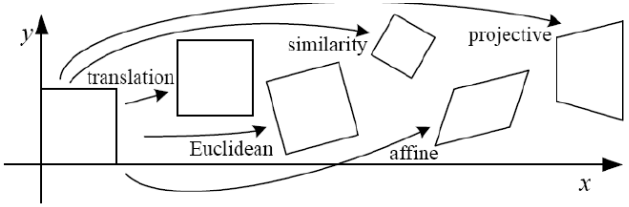
Minimal solution:

- $Q_{(2n\times 0)}$ should have rank 8 to have a unique (up to scale) non-trivial solution H.
- Each point correspondence provides 2 independent solutions, thus at least 4 **non-collinear points** is required.
- For an overdetermined solution $n > 4$ points SVD can deliver a solution.

Having found H, K and $[Rt]$ can be found making a QR decomposition such that:

$$\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix}$$

4.7.3 Types of 2D Transformations



Name	Matrix	# D.O.F.	Preserves:	Icon
translation	$\begin{bmatrix} \boldsymbol{I} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	2	orientation + ···	
rigid (Euclidean)	$\begin{bmatrix} \boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	3	lengths + ···	
similarity	$\begin{bmatrix} s\boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	4	angles +	
affine	$\begin{bmatrix} \boldsymbol{A} \end{bmatrix}_{2\times 3}$	6	parallelism + ···	
projective	$\begin{bmatrix} \tilde{\boldsymbol{H}} \end{bmatrix}_{3\times 3}$	8	straight lines	

This transformation is called Homography

4.7.4 Exercise Summary: Find the camera pose from 2D-3D correspondences (DLT algorithm)

- Goal: Calculation of $[R|t]$ that satisfy the perspective projection equation:

$$\begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{W} \end{bmatrix} = \lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K[R|t] \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

- This leads to calculating the solution of $Q \cdot M = 0$, as described in section 4.7. The following caveats are to be expected:
  1. If M solves $Q \cdot M = 0$, αM does as well, thus M is recovered up to an unknown scaling factor.
  2. Remember that Q is built using normalized coordinates, not pixel coordinates!
  3. To solve the overdetermined system of equations the least squares approach is implemented using singular value decomposition. It can be shown that the solution of this problem is the eigenvector corresponding to the smallest eigenvalue of Q^TQ, which simply corresponds to the last column of V if S, where $Q = USV^T$, has its diagonal entries sorted in descending order. The `svd` function in matlab provides that.

```
[~,~,V] = svd(Q);
M = reshape(V(:,12),4,3)';
```

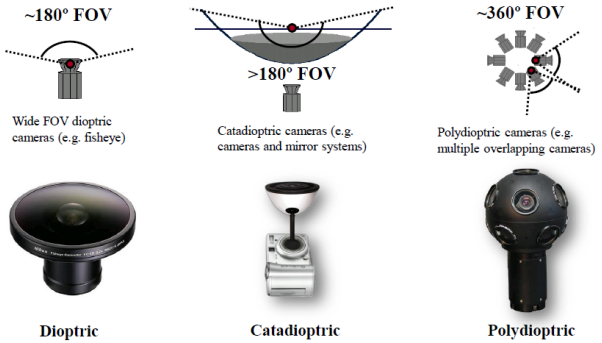
  4. To enforce R actually is a unitary matrix, we have to ensure that the M matrix, being a transformation matrix, actually makes a shift in positive z direction, thus $t_z = M_{34} > 0$.

```
if M(3,4)<0
    M=-M;
end
```

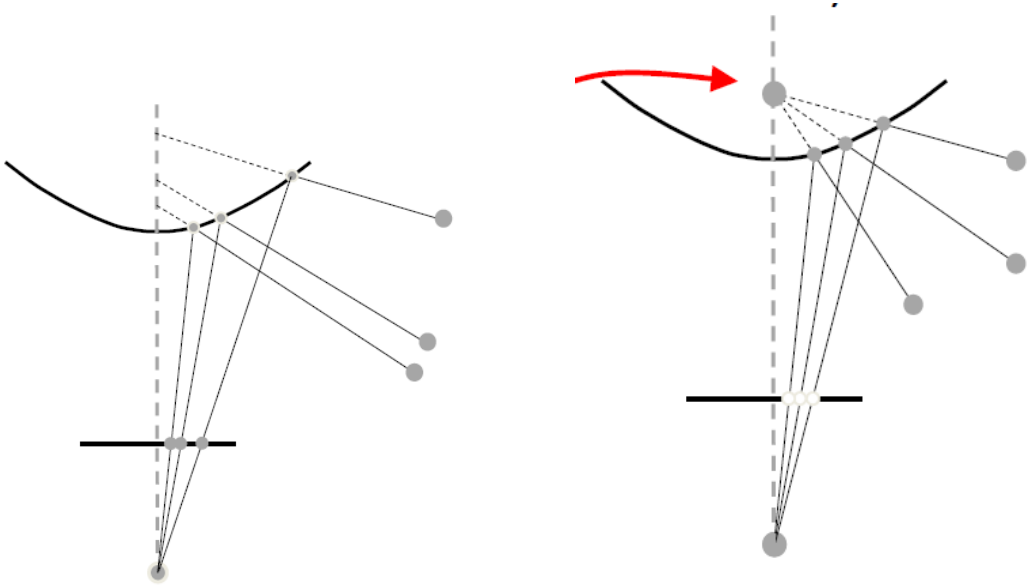
  5. Since the solution of the least squares problem only delivers the approximation of a homogeneous transformation matrix, we cannot be sure that R is actually a rotation matrix in $SO(3)$. To extract the actual rotation matrix from M, that is closest to the original estimated R, singular value decompositions is used. Hereby $R = USV^T$, which can be made unitary by removing the scaling factors in S. Thus $\tilde{R} = UV^T$.
  6. Since our solution M might actually be a scaled version αM. Now, having found the correct rotation matrix $\tilde{R}$, we can compare it to the original R to find the scale between the two. Thus $\alpha = \frac{||\tilde{R}||}{||R||}$. And the correct homogeneous transformation matrix is found as:

$$\tilde{M} = [\tilde{R}|\alpha t]$$

4.8 Omnidirectional Cameras



Catadioptric cameras come in different versions:

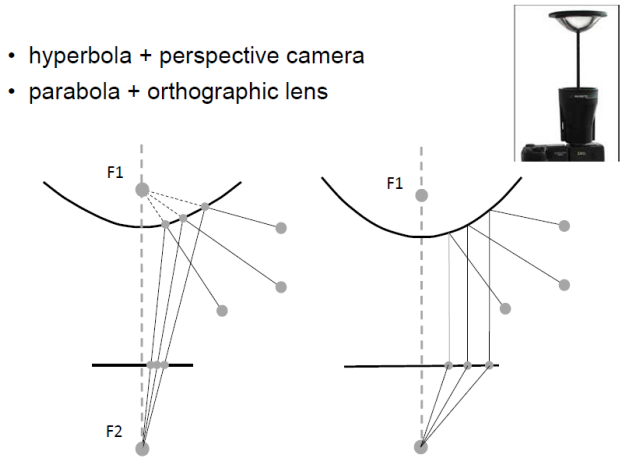


Definition 15. *A camera is a* ***central catadioptric camera*** *if the projection is such that there is a single effective viewpoint.*

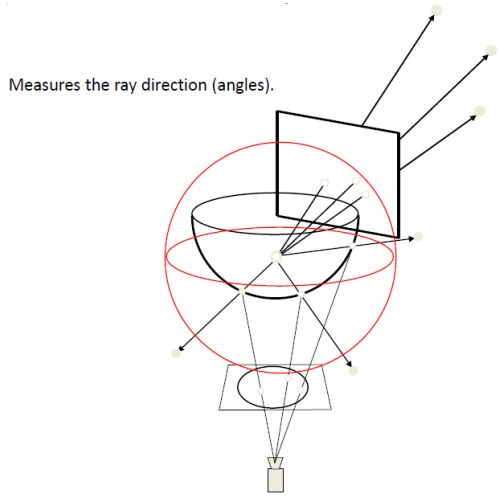
It is important to have a single view point to enable:

- Unwrapping an omnidirectional image into a perspective image.
- Transforming image points into normalized vectors on the unit sphere.
- Applying standard algorithms valid for perspective geometry.

4.8.1 Example for central catadioptric lenses



4.8.2 Equivalence between Perspective and Omnidirectional Model



4.9 Digital Images

Definition 16. ***Matlab coordinates:*** $[rows, cols]$
C/C++ coordinates: $[cols, rows]$

5 Filtering

- A smoothing filter has **positive values**, always **sums up to 1** to **preserve the overall brightness** of the picture and **removes high-frequency** contents, is thus a **low-pass filter**.

- A derivative filter has **opposite signs**, used to get high **responses in regions of high contrast**, **sums to 0** and highlights **high frequency components**.

5.1 Types of noise

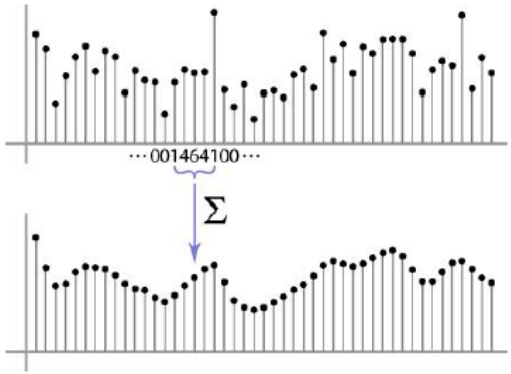
Definition 17. ***Salt and pepper noise****: random occurences of black and white pixels. Resulting from electromagnetive waves.*

Definition 18. ***Impulse noise****: random occurences of white pixels.*

Definition 19. ***Gaussian noise****: variations in intensity drawn from a Gaussian distribution. Very useful model for real world sensor noise.*

5.2 Noise removal

- Moving average filter. Based on the assumption of likeness of close pixels and the assumption of location-independent noise. It is possible to weigh pixels non-uniformly.



Definition 20. ***Convolution*** *defines the operation needed for the implementation of a moving average filter. One of the sequences is flipped and then slid over the other, multiplying each element with each other element and adding them up. A convolution is noted as $a_\star b$.*
Properties: linearity, associativity, commutativity

A convolution in 2D requires flipping the filter in both dimensions, which is equivalent to a 180° rotation. Then the convolution is defined as:

$$G[x,y] = \sum_{u=-k}^{k}\sum_{v=-k}^{k} F[x-u, y-v]H[u,v] \qquad G = F * H$$

Definition 21. *The* ***cross-correlation*** *is almost equivalent to a convolution, but does not include a flipping of the filter.*
Properties: linearity

$$G[x,y] = \sum_{u=-k}^{k}\sum_{v=-k}^{k} F[x+u, y+v]H[u,v]$$

5.3 Box filters

5.3.1 Moving Average Filter

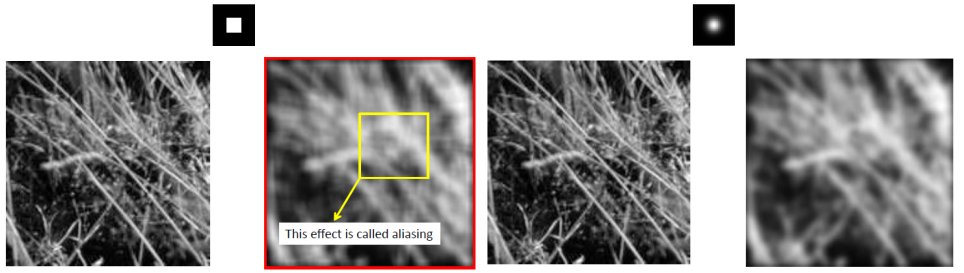
$$\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}\cdot\frac{1}{9} \quad \text{Unweighted}$$

$$\begin{bmatrix}1&2&1\\2&4&2\\1&2&1\end{bmatrix}\frac{1}{16} \quad \text{Weighted}$$

5.3.2 Gaussian Filter

$$H[u,v] = \frac{1}{2\pi\sigma^2}e^{-\frac{u^2+v^2}{2\sigma^2}}$$

A finely resolving Gaussian filter will not introduce aliasing as a box filter would.



Filter with sharp edges will cause aliasing, since they introduce high frequency contributions. A gaussian filter has the property of being smooth in the image domain as well as in the frequency domain. It does not introduce high frequency contributions.
Important parameters:

- Size of the kernel (How large is the box?)

  Good choice: 3σ, since 90% of the information is contained within that span.

- Variance of the filter (How large is sigma σ) The larger the image the more intense the blur.

```
gausswindow = fspecial('gaussian', [16, 16], 16 * 1.5);
```

5.4 Boundary issues

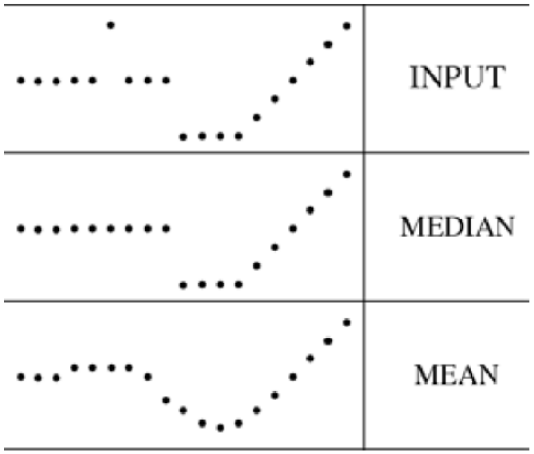
Ho to treat boundaries when filtering an image?

- zero padding (black)
- wrap around

- copy edge
- reflect across edge

5.5 Median Filter

A linear smoothing filter will not remove salt and pepper noise, but lead to more corruption of the image. For this reason a median filter is applied.



A median filter preserves sharp transitions but removes small brightness variations.

5.6 High-Pass Filtering

To accomplish edge detection it makes sense to consider the first order derivatives of the image. For a 2D function $F(x, y)$ the partial derivative is:

$$\frac{\partial F(x,y)}{\partial x} = \lim_{\epsilon \to 0} \frac{F(x+\epsilon,y) - F(x,y)}{\epsilon}$$

For discrete data ϵ is set to 1.

$$\frac{\partial F(x,y)}{\partial x} \approx \frac{F(x,+1,y) - F(x,y)}{1}$$

Possibilities of filters implementing that are:

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \text{ and } G_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \quad \text{Prewitt filter}$$

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \text{ and } G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad \text{Sobel filter}$$

```
%Apply Sobel Filter to Image
im = imread('lion.jpg');
h = fspecial('sobel');
outim = imfilter(double(im),h);
imagesc(outim);
colormap gray;
```

Since both direction are needed to represent all intensity changes within the image the gradient is used:

$$\nabla F = \left[\frac{\partial F}{\partial x}, \quad \frac{\partial F}{\partial y} \right]$$

$$\theta = \tan^{-1} \left(\frac{\frac{\partial F}{\partial y}}{\frac{\partial F}{\partial x}} \right) \quad \text{Gradient Direction}$$

$$||\nabla F|| = \sqrt{\left(\frac{\partial F}{\partial x}\right)^2 + \left(\frac{\partial F}{\partial y}\right)^2} \quad \text{Edge strength}$$

In order for the high pass filter not to pick up noise instead of the edge to be detected, the image needs smoothing before the edge detection. Convolution allows convolution of the two filters which, if using a gaussian for smoothing, is equivalent to filtering by the derivative of a gaussian filter.

This is implemented by the **Canny edge-detection algorithm (1986)**.

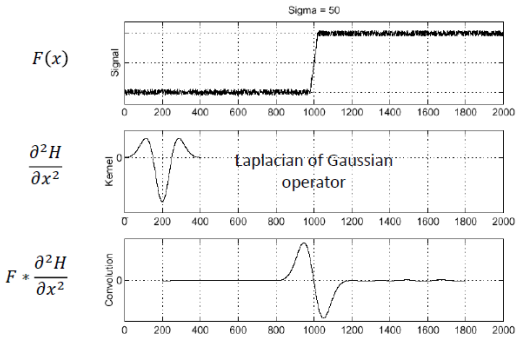
1. Conversion to grayscale.
2. Application of smoothing and gradient filter. (Derivatives of Gaussian)
3. Plotting of the edge strength.
4. Thresholding of the image. Setting all pixels below threshold to zero.
5. Non-maxima suppression (local-maxima detection) along edge direction.

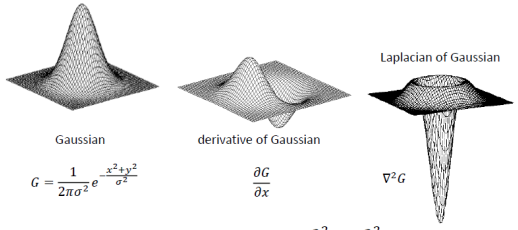
```
I_x_filter =...
    [-1 0 1;...
    -2 0 2;...
    -1 0 1];

%Calculate the gradient
I_x = conv2(img,I_x_filter,'valid');
%'valid' => only include domain in which the whole filter
   fits on the image
```

5.7 Laplacian of the Gaussian

Instead of analysing the first derivative (searching for maxima to find edges) we can take the second derivative and detect the same maximum by finding the zero crossing of the second derivative.





- ∇^2 is the Laplacian operator: $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$

```
gausswindow = fspecial('gaussian', [16, 16], 16 * 1.5);
```

5.8 Canny edge-detection

1. Compute gradient of smoothed image in both directions.
2. Discard Pixels whose gradient is below a certain threshold.
3. Non-maxima suppression: identify local maxima along gradient direction.

6 Point Feature Detection and Matching

6.1 Template Matching

The correlation of a filter (a template) and the image can be used to find the location of the template, as the maximum of the filtered image. The matching will only work though if scale, orientation, illumination and the in general the appearance of the object and the template are similar.

6.1.1 Similarity Measures

Definition 22. *The **Sum of Absolute Differences (SAD)** is defined as*

$$SAD = \sum_{u=-k}^{k} \sum_{v=-k}^{k} |H(u,v) - F(u,v)|$$

Definition 23. *The **Sum of Squared Differences (SSD)** is defined as*

$$SSD = \sum_{u=-k}^{k} \sum_{v=-k}^{k} (H(u,v) - F(u,v))^2$$

SSD is computationally expensive.

Definition 24. *The **Normalized Cross Correlation (NCC)** takes values between -1 and $+1$ where ($+1$ is taken if the images are identical). It is defined as*

$$NCC = \frac{\sum_{u=-k}^{k} \sum_{v=-k}^{k} H(u,v)F(u,v)}{\sqrt{\sum_{u=-k}^{k} \sum_{v=-k}^{k} H(u,v)^2} \sqrt{\sum_{u=-k}^{k} \sum_{v=-k}^{k} F(u,v)^2}}$$

To account for illumination differences the mean of each image is subtracted before calculating similarity. This leads to the definition of

Definition 25. ***Zero-mean SAD, SSD, NCC***

$$\mu_H = \frac{\sum_{u=-k}^{k} \sum_{v=-k}^{k} H(u,v)}{(2N+1)^2}$$

$$ZSAD = \sum_{u=-k}^{k} \sum_{v=-k}^{k} |(H(u,v) - \mu_H) - (F(u,v) - \mu_F)|$$

$$ZSSD = \sum_{u=-k}^{k} \sum_{v=-k}^{k} ((H(u,v) - \mu_H) - (F(u,v) - \mu_F))^2$$

The above are note invariant the affine illumination changes, ZNCC is.

$$ZNCC = \frac{\sum_{u=-k}^{k} \sum_{v=-k}^{k} (H(u,v)-\mu_H)(F(u,v)-\mu_F)}{\sqrt{\sum_{u=-k}^{k} \sum_{v=-k}^{k} (H(u,v)-\mu_H)^2} \sqrt{\sum_{u=-k}^{k} \sum_{v=-k}^{k} (F(u,v)-\mu_F)^2}}$$

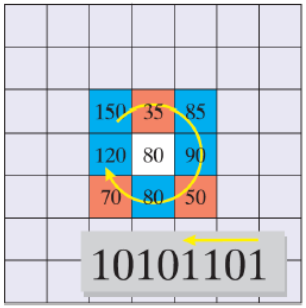
Definition 26. ***Affine intensity changes** are defined as*

$$I'(x,y) = \alpha I(x,y) + \beta$$

6.1.2 Census Transform

Definition 27. *The **Hamming distance** of two strings is the number of bits that are different.*

The census transform maps an image patch to a bit string (value larger than center pixel $\rightarrow$ 1 otherwise $\rightarrow$ 0) and compares strings using the Hamming distance.



Advantages

- More robust to object-background problem (same object, different background yields less similarity).
- No square roots or division required, thus very efficient, especially on FPGA.
- Intensities are considered relative to the center pixel of the patch making it invariant to monotonic intensity changes.

Definition 28. ***FPGA*** *is a field programmable gate array. Thus an integrated circuit designed to be configured by a customer or a designer after manufacturing.*

6.2 Feature Matching

Challenges

- Find distinctive features.
- Account for rotations, translations, distortions, color changes, illumination changes, lens-imperfections.
- Find the same feature in both images (repeatability).
- Match the corresponding points.

Definition 29. *A **corner** is defined as the intersection of one or more edges.*

\+ A corner has high localization accuracy (very good for VO)

\- Less distinctive than a blob.

E Harris, Shi-Tomasi, SUSAN, FAST

Definition 30. *A **blob** is any other image pattern, which is not a corner, that differs significantly from its neighbours in intensity and texture.*

\- Has less localization accuracy than a corner.

\+ Blob detectors are better for place recognition.

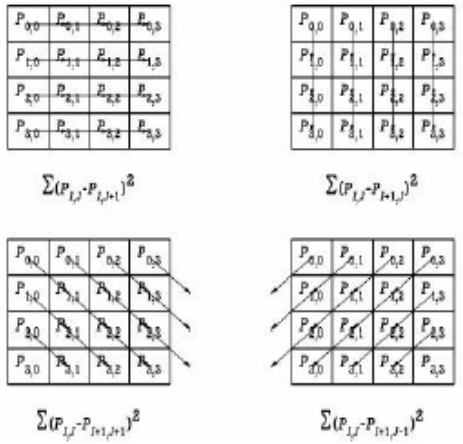
\+ More distinctive than a corner.

E MSER, LOG, DOG (SIFT), SURF, CenSurE

6.2.1 The Moravec Corner Detector

1. A corner has a significant change in SSD in at least 2 directions and is thus repeatable and distinctive.

2. Sums of squares of differences of pixels adjacent in each of four directions (horizontal, vertical and two diagonals) over each window are calculated, and the window's interest measure is the minimum of these four sums. This makes sense since a corner is indicated by changes in all directions.



6.2.2 Harris Corner Detector

1. Consider the reference patch centered at (x,y) and the shifted windows centered at $(x + \Delta x, y + \delta y)$. The patch has size P.

2. Calculate the sum of squared differences (SSD):

   a) $I_x = \frac{\partial I(x,y)}{\partial x}$, $I_y = \frac{\partial I(x,y)}{\partial y}$, approximated with first order Taylor:

   $$I(x + \Delta x, y + \Delta y) \approx I(x, y) + I_x(x, y)\Delta x + I_y(x, y)\Delta y$$

   b) Thus SSD can be approximated as

   $$SSD(\Delta x, \Delta y) \approx \sum_{x,y \in P} (I_x(x, y)\Delta x + I - y(x, y)\Delta y)^2$$

   c) This can be written in matrix form:

   $$SSD(\Delta x, \Delta y) \approx \sum_{x,y \in P} \begin{bmatrix}\Delta x & \Delta y\end{bmatrix} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \begin{bmatrix}\Delta x \\ \Delta y\end{bmatrix} \Rightarrow SSD(\Delta x, \Delta y) \approx \begin{bmatrix}\Delta x & \Delta y\end{bmatrix} M \begin{bmatrix}\Delta x \\ \Delta y\end{bmatrix}$$

   where $M = \sum_{x,y \in P} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$.

   Notice that I_x^2 for example is note a matrix product but a pixel-wise product!

3. For any corner rotated with ϕ: $M = \begin{bmatrix}\cos(\phi) & -\sin(\phi) \\ \sin(\phi) & \cos(\phi)\end{bmatrix} \begin{bmatrix}\lambda_1 & 0 \\ 0 & \lambda_2\end{bmatrix} \begin{bmatrix}\cos(\phi) & \sin(\phi) \\ -\sin(\phi) & \cos(\phi)\end{bmatrix}$

   If any λ is close to zero there is no corner.

4. Singular value decomposition can be used to find λ_i directly.

5. λ_i can be used to identify a corner. A corner has been found if the minimum of the two eigenvalues is larger than a certain threshold.

6. **Cornerness function** $R = \min(\lambda_1, \lambda_2)$

7. The corner detector using this criterion is called **Shi-Tomasi** detector.

8. Alternatively, to avoid the calculation of the eigenvalues, another cornerness function can be defined: $R = \lambda_1 - \lambda_2 - k(\lambda_1 + \lambda_2)^2 = \det(M) - k\text{trace}^2(M)$, where $k \in (0.04, 0.15)$.

\+ Invariant to image rotation.

\- Not invariant to image scale.

6.3 Scale changes

- Scale changes are difficult to detect using a patch of a defined size.
- A possible solution would be rescaling the patch in order to match a feature with itself after a scale change.
- To remedy to computational intensity included in testing all n patches for s different scales a possible solution would be to assign a scale to each feature.

6.4 Automatic Scale Selection

- Idea: Define a function that is invariant to scale changes.
- $f(x, y, patch - size)$ would be the candidate function that has a value for each patch size. Now this function has a maximum at a certain patch size, which allows defining the scale of the detected feature.
- **This has to be done for each feature individually!**
- After finding the scale of a picture it can be normalized, this reduces the computational effort of finding matches of features in different images can be reduced significantly, since the iteration over patch sizes is not needed anymore.
- If there are multiple local maxima, the feature is replicated multiple times at the corresponding scales. The computational effort is still reduced that way.

How to find that function?

- Convolution with a kernel that highlights edges:

  $$f = Kernel * Image$$

- The Laplacian of Gaussian kernel is most effective under certain conditions.

  $$L \circ G = \nabla G(x, y)$$

- The $L \circ G$ Kernel already includes smoothing.

How to describe features?

- Idea: Find a feature descriptor that is **invariant** to geometric and photometric changes.

How to achieve invariance with Patch descriptors?

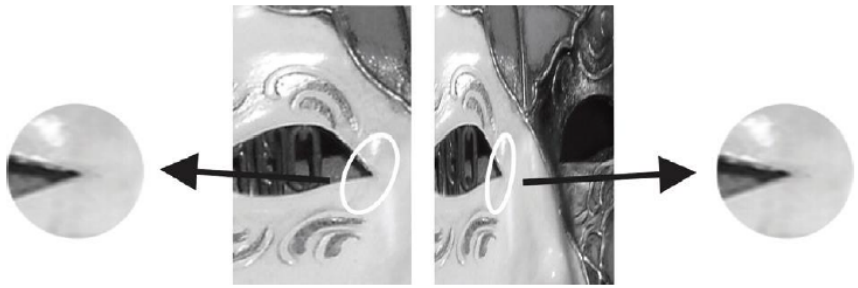
1. Re-scaling and De-rotation:
   a) Find correct scale using LoG operator.
   b) Rescale the patch.
   c) Find local orientation (Dominant direction of gradient (Harris eigenvectors)).

d) De-rotate patch through patch warping. → Canonical orientation.

- Start with an empty canonical patch.
- For each pixel in the destination patch find the corresponding location in the source patch as defined by the **warping function**.
- Interpolate in the source frame to find the intensity in the destination frame.
- **Roto-Translational Warping:**

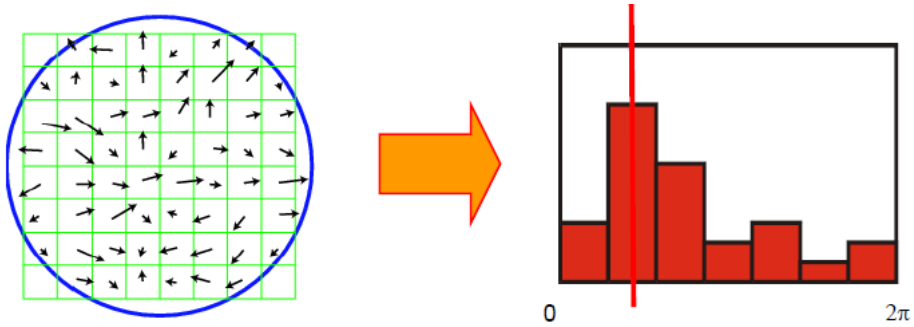
$$x' = x\cos\theta - y\sin\theta + a$$
$$y' = x\sin\theta + y\cos\theta + b$$

- **Affine Warping:** The second moment matrix M can be used to identify the two directions of fastest and slowest change of intensity, which can be used to define an elliptic patch, that is then normalized to a circular patch.



2. **Disadvantages:**
   - If warping not accurate the matching score decreases significantly.
   - Computationally expensive.

3. HOG descriptor (Histogram of Oriented Gradients)
   a) Multiply the patch by a Gaussian kernel.
   b) Compute gradients vectors at each pixe.
   c) Build a histogram of gradient orientations, weighted by the gradient magnitudes.
   d) Extract all local maxima and make a descriptor (HOG) for each.
   e) Apply a circular shift to the descriptor such that the detected maximum corresponds with 0 degrees.



6.5 SIFT Descriptor

1. Multiply the patch by a Gaussian filter
2. Divide the patch into 4x4 sub-patches
3. Compute HOG (8bins, i.e. 8 directions) for all pixels inside each sub-patch.
4. Concatenate all HOGs into a single 1D vector. $4x4x8 = 128$ values.
5. The descriptor vector v is then normalized such that:

$$\bar{v} = \frac{v}{\sqrt{\sum_i^n v_i^2}}$$

   This guarantees invariance to linear illumination changes. Overall the SIFT descriptor is invariant to affine illumination changes.

\+ Can handle severe viewpoint changes (up to 50°.

\+ Can handle even non affine changes in illumination (low to bright scenes).

\- Computationally expensive: 10 frames per second on an i7.

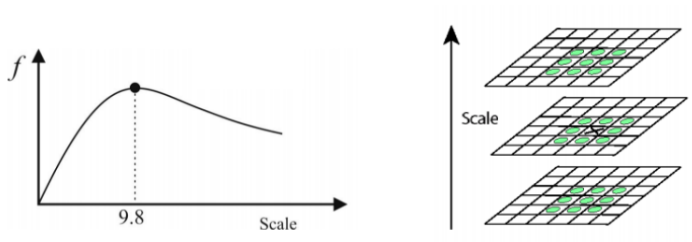
6.6 SIFT detector

Idea: Detect keypoints as local extrema over the image as well as over the patch scale, using a Difference of Gaussian (DoG) kernel.

1. Incrementally convolve the initial image with Gaussians $G(k^i\sigma)$ to produce blurred images separated by a constant factor k in scale space.
   a) Initial Gaussian: $\sigma = 1.6$.
   b) k is chosen. $k = 2^{1/s}$ where s is the number of intervals into which each octave of scale space is divided.
   c) For efficiency reasons, when k^i equals 2, the images is downsampled by a factor of 2 and the procedure is repeated again up to 5 octaves. Downsampling and reusing the gaussian kernels is equivalent to increasing i further, but much more computationally efficient.

2. Ajdacent blurred images are then subtracted to produce the Difference-of-Gaussian (DoG) images.

3. Local Maxima a found in the resulting scales (scale = 3 DoGs).



An efficient approach to that problem is using an image dilation algorithm for detecting maxima. `imdilate(image,mask)`

For a twodimensional peak-search the algorithm using imdilate would do the following:

a) Define the mask as: $\begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$.

b) Then the dilation serves to calculate the neighbouring maximum for each pixel.

c) Finally the resulting dilated image is compared to the original image. The local maxima are found as `original>filtered` which in Matlab results in a logical array, where 1s indicate a local maximum. In other words this comparison yields `true` if the pixel is larger than the maximum of its neighbours.

In this case the used mask is the 3D equivalent of the mask used above.

6.6.1 Summary

- Based on the property of the LoG allowing to recognize features of a certain radius, and on the idea that that radius may be changed by changing the standard deviation of the gaussian curve, one can vary the peak-finding function over the scale by varying σ. In addition, instead of increasing the radius of the filter, the image size is decreased for the next octave of Gaussians.

Number of parameters used:

- **Descriptor:** 4x4x8 = 128-element 1D vector
- **Location:** (pixel coordinates of the center of the patch): 2D vector
- **Scale:** (i.e. size) of the patch: 1 scalar value
- **Orientation:** (i.e. angle of the patch): 1 scalar value

```
%use of imdilate
%Finding local maxima
%For example in a DoG pyramid
DoG_dilated = imdilate(DoG, true(3,3,3));
%Including all values
DoG_ind_max = DoG_dilated == DoG_dilated;
%Excluding nonzero values
DoG_ind_max = DoG_dilated == DoG_dilated & DoG;
%Dilate multiplies the given pattern with each position in
    an array (pixelwise) and sets the corresponding position
    to the maximum of the multiplication
```

6.7 Feature matching

How to find the best match for patch in I_1 with patch in I_2?

1. Define distance function for comparison ((Z)SSD, SAD, NCC or Hamming distance for binary descriptors (Census, BRIEF, BRISK)).

2. Brute-force matchin

   a) Test all the features in I_2.

   b) Take the one a min distance.

The closest descriptor can give very good scores to ambiguous matches. The solution is to **compute the ratio of distances** between first and second match.

$$d(f_1)/d(f_2) < Threshold \text{ (usually 0.8)}$$

Motivation for distance ratio

- Features in SIFT are matched based on euclidean distance, but there might be many features that appear in one image but not in the other.
- Comparison of the distances between closest neighbour and second closest neighbour performs well, since for a match to be true the closest neighbour needs to be significantly closer than the second closest.
- For false matches there will likely be a number of additional false matches, which will make $d(f_1)/d(f_2)$ go closer to 1. It can be thought of as an estimate of the density of false matches in a region.

6.8 Overview

Find the corresponding papers in section 2.2.

Detector	Localization Accuracy of the detector	Descriptor that can be used	Efficiency	Relocalization & Loop closing
Harris	++++	Patch	+++	+
		SIFT	+	+++++
		BRIEF	++++	+++
		ORB	++++	++++
		BRISK	+++	+++
		FREAK	++++	++++
Shi-Tomasi	++++	Patch	++	+
		SIFT	+	+++++
		BRIEF	++++	+++
		ORB	++++	++++
		BRISK	+++	+++
		FREAK	++++	++++
FAST	++++	Patch	++++	+++
		SIFT	+	+++++
		BRIEF	++++	+++
		ORB	++++	++++
		BRISK	+++	+++
		FREAK	++++	++++
SIFT	+++	SIFT	+	++++
SURF	+++	SURF	++	++++

7 Multi View Geometry

7.1 3D Reconstruction from Multiple Views

	3D Reconstruction	Structure from Motion
K, T, R	known	unknown
Goal	Find 3D structure	Simultaneous recovery of 3D scene and camera pose graph (up to scale).

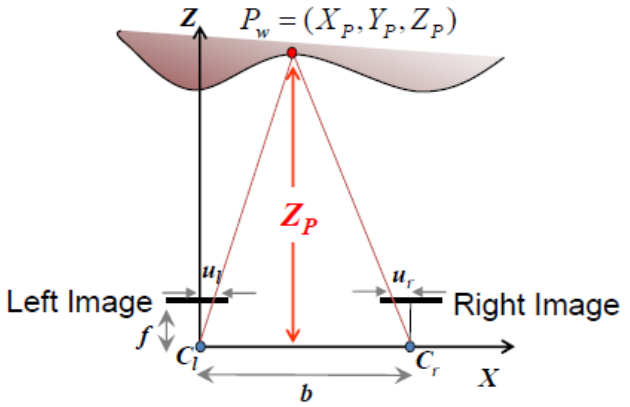
7.2 Triangulation

- From a single camera only the ray on which each image point lies can be calculated.
- With a stereo camera (binocular) the intersection of rays delivers the 3D structure.

Definition 31. ***Stereopsys*** *is the brains ability to reproduce a single 3D image from the two retinal 2D images.*

Definition 32. *The process of flipping retinal images upside down and the removal of radial distortion is called* ***rectification****.*

Definition 33. *The* ***disparity*** *between two images describes the distance between the projections of a single 3D point on the two images. The closer the object the larger the disparity.*

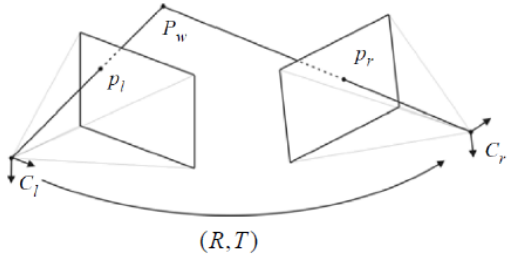


$$\frac{f}{Z_P} = \frac{u_l}{X_P}$$
$$\frac{f}{Z_P} = \frac{-u_r}{b - X_P}$$
$$Z_P = \frac{bf}{u_l - u_r}$$

b	Baseline	[mm]
$u_l - u_r$	Disparity	[mm]
f	focal length	[mm]
Z_P	Depth	[mm]

- Maximum disparity: Sensor width.
- Disparity of a point at infinity: 0.
- The uncertainty of the disparity depends on the distance of the viewed object. The further away the more uncertain the disparity estimation.
- How to increase the accuracy of a stereo system: Optimize the baseline **b**.
  - ↓ **b** Large depth error.
  - ↑ **b** Minimum measurable distance increases.
  - ↑ **b** Difficult search problem for close objects.

7.2.1 General Case



1. Parallel alignment of cameras is hard.
2. For the use of a stereocamera the following are needed:
   - Extrinsic parameters (relative rotation and translation)
   - Intrinsic parameter (focal length, optical center, radial distortion of each camera)

$$\tilde{p}_l = \lambda_l \begin{bmatrix} u_l \\ v_l \\ 1 \end{bmatrix} = K_l \begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix} \qquad \tilde{p}_r = \lambda_r \begin{bmatrix} u_r \\ v_r \\ 1 \end{bmatrix} = K_r R \begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix} + T$$

which can be formulated as

Left camera	Right camera
$\lambda_1 \begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix} = \underbrace{K[I\|0]}_{M_1} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$	$\lambda_2 \begin{bmatrix} u_2 \\ v_2 \\ 1 \end{bmatrix} = \underbrace{K[R\|T]}_{M_2} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 2 \end{bmatrix}$
$p_1 \times M_1 \cdot P = 0$	$p_2 \times M_2 \cdot P = 0$
$[p_{1\times}]\, M_1 \cdot P = 0$	$[p_{2\times}]\, M_2 \cdot P = 0$

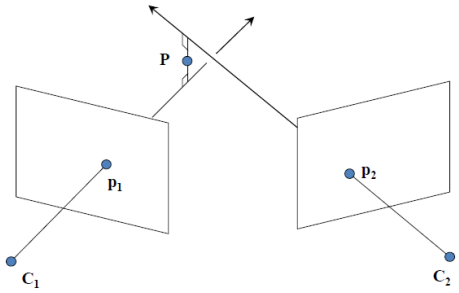
where the cross product can be interpreted as a matrix vector product:

$$a \times b = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = [a_\times]b$$

The two matrix equations above deliver an overdetermined set of equations that can be solved using singular value decomposition.

Geometric interpretation and Nonlinear Approach of the triangulation:

Even for the best possible solution the two rays corresponding to the same object might not have an intersection in 3D. Thus the challenge is to minimize the smallest distance between two corresponding rays.



- Find P that minimizes the **Sum of Squared Reprojection Error**:

  $$SSRE = d^2(p_1, \pi_1(P)) + d^2(p_2, \pi_2(P))$$

  where $d(p_1, \pi_1(P)) = ||p_1 - \pi_1(P)||$ is called **Reprojection Error**.

- Use Gauss-Newton or Levenberg-Marquardt for minimization of the SSRE.

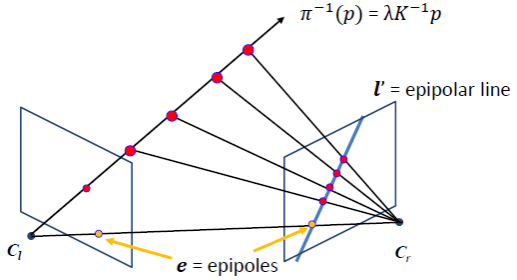
7.3 Epipolar Geometry

7.3.1 Correspondence Problem

Problem: Identify corresponding points in 2 images. **Solution:** Use epipolar geometry to reduce the number of possibly matching points by one dimension (!), thus omitting an exhaustive search.

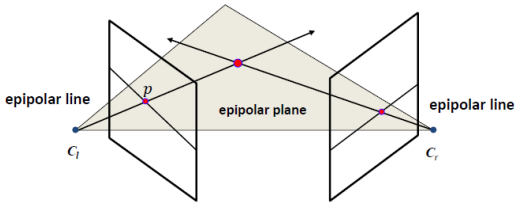
Definition 34. *The **epipolar** line is the projection of the infinite ray $\pi^{-1}(p)$ corresponding to p in the other camera image.*

Definition 35. *The **epipole** is the projection of the optical center on the other camera image.*



Definition 36. *The **epipolar plane** is uniquely defined by the two optical centers C_l, C_r and one image point p.*

Definition 37. *The **epipolar constraint** constrains the location, in the second view of the corresponding point to a given point in the first view.*



7.4 Stereo rectification

Since even commercial stereocameras are never perfectly aligned and for computational reasons, it is practical to have the scanlines aligned with the epipolar lines a stereo rectification is performed to make for such an alignment.

Idea: Project both images on a common plane parallel to the baseline.

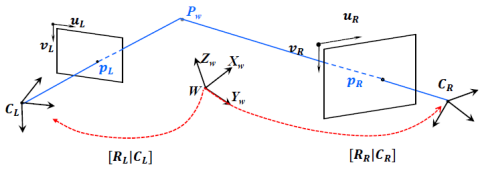
Approach

1. Rotate the optical planes around their optical centers until they are coplanar.
2. Thus the epipoles are at infinity and the epipolar lines are parallel.
3. To have horizontal epipolar lines the baseline must be parallel to the new X axis of both cameras.
4. Further, the corresponding points must have the same y coordinate. Thus both camera must have the same intrinsic parameters.

7.4.1 Mathematical solution

1. $\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = KR^{-1} \left[\begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} - C \right]$

2. Apply the perspective equation to both images.



3. Mapping of both cameras to coplanar planes.

$$\lambda_L \begin{bmatrix} u_L \\ v_L \\ 1 \end{bmatrix} = K_L R_L^{-1} \left[\begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix} - C_L \right] \rightarrow \tilde{\lambda}_L \begin{bmatrix} \tilde{u}_L \\ \tilde{v}_L \\ 1 \end{bmatrix} = \tilde{K}\tilde{R}^{-1} \left[\begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix} - C_L \right]$$
$$\lambda_R \begin{bmatrix} u_R \\ v_R \\ 1 \end{bmatrix} = K_R R_R^{-1} \left[\begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix} - C_R \right] \rightarrow \tilde{\lambda}_R \begin{bmatrix} \tilde{u}_R \\ \tilde{v}_R \\ 1 \end{bmatrix} = \tilde{K}\tilde{R}^{-1} \left[\begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix} - C_R \right]$$

4. By solving for $\begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix}$ for each camera (between new and old mapping) we can find the homography that has to be applied to each camera for rectification.

$$\bar{\lambda}_L \begin{bmatrix} \bar{u}_L \\ \bar{v}_L \\ 1 \end{bmatrix} = \lambda_L \bar{K}\bar{R}^{-1} R_L K_L^{-1} \begin{bmatrix} u_L \\ v_L \\ 1 \end{bmatrix} \qquad \bar{\lambda}_R \begin{bmatrix} \bar{u}_R \\ \bar{v}_R \\ 1 \end{bmatrix} = \lambda_R \bar{K}\bar{R}^{-1} R_R K_R^{-1} \begin{bmatrix} u_R \\ v_R \\ 1 \end{bmatrix}$$

5. How to choose the new $\bar{K}$ and $\bar{R}$:

$$\bar{K} = (K_L + K_R)/2$$
$$\bar{R} = \begin{bmatrix} \bar{r}_1 & \bar{r}_2 & \bar{r}_3 \end{bmatrix}$$

with $\bar{r}_1, \bar{r}_2, \bar{r}_3$ being the column vectors of $\bar{R}$, where

$$\bar{r}_1 = \frac{C_2 - C_1}{||C_2 - C_1||} \qquad \bar{r}_2 = r_3 \times \bar{r}_1 \qquad \bar{r}_3 = \bar{r}_1 \times \bar{r}_2$$

7.5 Correspondence Search

After rectification the correspondence search can be done along the same lines. The complexity has been reduced by one dimension.
To improve on ambiguity in textureless regions, the window size is increased.

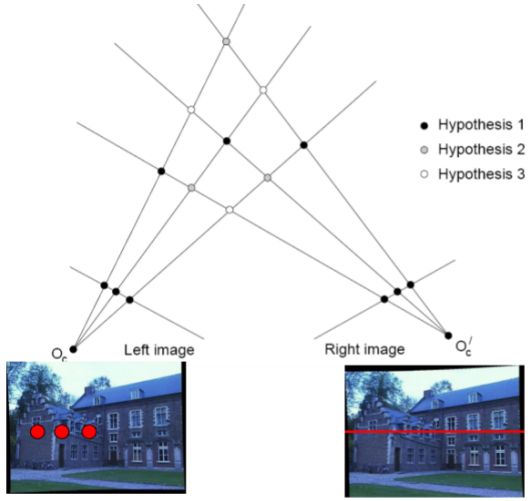
- Smaller Window
  - \+ More detail
  - \- More noise
- Larger Window
  - \+ Smoother disparity maps
  - \- Less detail

7.5.1 Disparity Map

1. For each pixel on the left image, find its corresponding point on the right image.
2. Compute the disparity for each pair of correspondences.
3. Visualize it in gray-scale or color-coded image.

7.5.2 Correspondence Problems

Multiple Matches



To improve on that introduce soft constraints:

- Uniqueness: Only one match in right image for every point in left image.
- Ordering: Points on same surface will be in same order in both views.
- Disparity gradient: Disparity changes smoothly between points on the same surface.

8 Two-View Structure from Motion

Given n point correspondence between two images $\{p_1^i = (u_1^i, v_1^i),\ p_2^i = (u_2^i, v_2^i)\}$, simultaneously estimate the 3D points $\underline{P}^i$, the camera relative motion parameters $(\underline{R}, \underline{T})$ and the camera intrinsics $\underline{K}_1, \underline{K}_2$ that satisfy

$$\lambda_1 \begin{bmatrix} u_1^i \\ v_1^i \\ 1 \end{bmatrix} = K_1[I|0] \begin{bmatrix} X_W^i \\ Y_W^i \\ Z_W^i \\ 1 \end{bmatrix}$$

$$\lambda_2 \begin{bmatrix} u_2^i \\ v_2^i \\ 1 \end{bmatrix} = K_2[I|0] \begin{bmatrix} X_W^i \\ Y_W^i \\ Z_W^i \\ 1 \end{bmatrix}$$

For the calibrated case the intrinsics are know.
Thus the equations are normalized:

$$\lambda_1 \begin{bmatrix} \bar{u}_1^i \\ \bar{v}_1^i \\ 1 \end{bmatrix} = [I|0] \begin{bmatrix} X_W^i \\ Y_W^i \\ Z_W^i \\ 1 \end{bmatrix}$$

$$\lambda_2 \begin{bmatrix} \bar{u}_2^i \\ \bar{v}_2^i \\ 1 \end{bmatrix} = [I|0] \begin{bmatrix} X_W^i \\ Y_W^i \\ Z_W^i \\ 1 \end{bmatrix}$$

8.1 Scale Ambiguity

Rescaling the entire scene (camera views and object) will not have an effect on the projection of the scene. Thus the scale is ambiguous. In monocular vision it is **not possible** to recover the absolute scale of the scene.

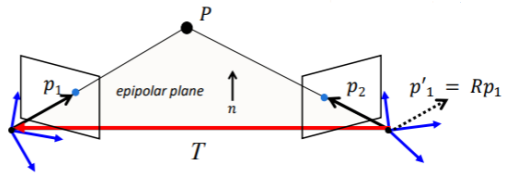
Thus only 5 degrees of freedom are measurable.

- 3 parameters to describe the rotation.
- 2 parameters for the translation up to scale.
- $4n$ knowns from n correspondences.
- $5{+}3n$ unknowns, 5 for motion up to scale, $3n$ = number of the coordinates of the n 3D points.

A solution exists only if

$$4n \geq 5 + 3n \Rightarrow n \geq 5$$

8.2 Solution through Epipolar Geometry



Since p_1, p_2 and T are coplanar the epipolar constraint can be established. It basically states that the crossproduct of p_1' (p_1 described in the frame of the second camera) and T will deliver the vector normal to the plane. The dot product of that vector with p_2 has to be zero since they are perpendicular. Using R to describe $p_1' = Rp_1$, the constraint is:

$$\boxed{p_2^T E p_1 = 0 \qquad E = [T]_\times R}$$

$$\bar{p}_2^T E \bar{p}_1 = 0$$

Each pair of correspondences $\bar{p}_1 = (\bar{u}, \bar{v}, 1)^T$, $\bar{p}_2 = (\bar{u}_2, \bar{v}_2, 1)^T$ provides a linear equation. For n points:

$$\begin{bmatrix} \bar{u}_2^1\bar{u}_1^1 & \bar{u}_2^1\bar{v}_1^1 & \bar{u}_2^1 & \bar{v}_2^1\bar{u}_1^1 & \bar{v}_2^1\bar{v}_1^1 & \bar{v}_2^1 & \bar{u}_1^1 & \bar{v}_1^1 & 1 \\ \bar{u}_2^2\bar{u}_1^2 & \bar{u}_2^2\bar{v}_1^2 & \bar{u}_2^2 & \bar{v}_2^2\bar{u}_1^2 & \bar{v}_2^2\bar{v}_1^2 & \bar{v}_2^2 & \bar{u}_1^2 & \bar{v}_1^2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \bar{u}_2^n\bar{u}_1^n & \bar{u}_2^n\bar{v}_1^n & \bar{u}_2^n & \bar{v}_2^n\bar{u}_1^n & \bar{v}_2^n\bar{v}_1^n & \bar{v}_2^n & \bar{u}_1^n & \bar{v}_1^n & 1 \end{bmatrix} \begin{bmatrix} e_{11} \\ e_{12} \\ e_{13} \\ e_{21} \\ e_{22} \\ e_{23} \\ e_{31} \\ e_{32} \\ e_{33} \end{bmatrix} = 0$$

$$\boxed{Q \cdot \bar{E} = 0}$$

Minimal solution:

- $Q_{n\times 9}$ should have rank 8 to have a unique (up to scale) non-trivial solution.
- Each p2p correspondence provides one independent equation.
- Thus 8 points are required at least.

Over-determined solution:

- $n > 8$
- A solution is to minimize $||Q\bar{E}||^2$ subject to the constraint $||\bar{E}||^2 = 1$. The solution is the eigenvector corresponding to the smallest eigenvalue of Q^TQ since it is the solution of the minimization $||Qx||^2 = x^TQ^TQx$.

```
[U,S,V] = svd(Q);
Ev = V(:,9);
E = reshape(Ev,3,3)';
```

- **Degenerate Configurations** The solution is degenerate if the 3D points are coplanar.

8.3.1 Implementation of Q

```
%First version according to lecture slides:
p1_T = p1.';
p2_T = p2.';

Q = [p1_T(:,1).*p2_T(:,1),...
    p1_T(:,2).*p2_T(:,1),...
    p1_T(:,3).*p2_T(:,1),...
    p1_T(:,1).*p2_T(:,2),...
    p1_T(:,2).*p2_T(:,2),...
    p1_T(:,3).*p2_T(:,2),...
    p1_T(:,1).*p2_T(:,3),...
    p1_T(:,2).*p2_T(:,3),...
    p1_T(:,3).*p2_T(:,3)];

[~,~,V] = svd(Q);
Eh = V(:,end);

E = reshape(Eh,3,3)';

[U,S,V] = svd(E);
S(3,3) = 0;
F = U*S*V';
```

```matlab
%Second Version according to exercises
N = size(p1,2);
Q = zeros(N,9);

% building Q
for i = 1:N
    Q(i,:) = kron(p1(:,i),p2(:,i))';
end

% solve Q*vec(F)=0
[~,~,VQ] = svd(Q);
vecF = VQ(:,end);
F = reshape(vecF,3,3);

% ensure that det(F) = 0
[U,S,V] = svd(F);
S(3,3) = 0;
F = U * S * V';
```

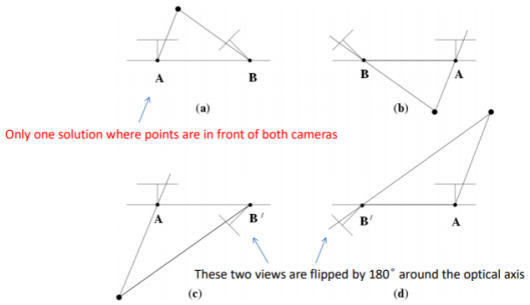
8.4 The Fundamental Matrix

For the uncalibrated case the intrinsics are unknow.
By substituting the definition of normalized coordinates

$$\begin{bmatrix}\bar{u}\\ \bar{v}\\ 1\end{bmatrix} = K_1^{-1}\begin{bmatrix}u\\ v\\ 1\end{bmatrix}$$

into the epipolar constraint we get the epipolar constraint for uncalibrated cameras:

$$\begin{bmatrix}u_2^i\\ v_2^i\\ 1\end{bmatrix}^T \underbrace{K_2^{-T}EK_1^{-1}}_{\text{Fundamental Matrix : } F} \begin{bmatrix}u_1^i\\ v_1^i\\ 1\end{bmatrix} = 0$$



There exist 4 different ways for finding R and T from E.

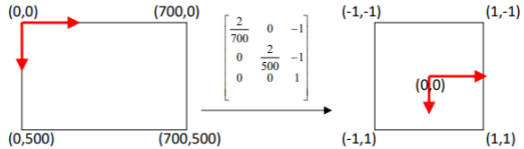
+ We work in pixel coordinates.

- This algorithm assumes 0 lens distortion. If that is not given it has to be corrected for.

- Numerically unstable.

8.5 Normalized 8-Point Algorithm

Poor numerical conditioning in the 8-Point algorithm motivates normalization. The idea is to transform the image coordinates such that they are in the range $[-1,1]\times[-1,1]$.



To achieve such a transformation the points are transformed such that the coordinate origin is set in the centroid of the image and the average point is located at $[1,1,1]^T$ (in homogeneous coordinates).

$$\hat{p}^i = \frac{\sqrt{2}\sigma}{(p^i-\mu)}$$

This can be formulated in a matrix:

$$\hat{p}^i = \begin{bmatrix}\frac{\sqrt{2}}{\sigma} & 0 & -\frac{\sqrt{2}}{\sigma}\mu_x\\ 0 & \frac{\sqrt{2}}{\sigma} & -\frac{\sqrt{2}}{\sigma}\mu_y\\ 0 & 0 & 1\end{bmatrix} p^i$$

1. Normalize point correspondences: $\hat{p}_1 = B_1 p_1$, $\hat{p}_2 = B_2 p_2$
2. Estimate normalized $\hat{F}$
3. Compute unnormalized F from $\hat{F}$:
$$F = B_2^T \hat{F} B_1$$

- R, T, K_1, K_2 can in general not be extracted from F.

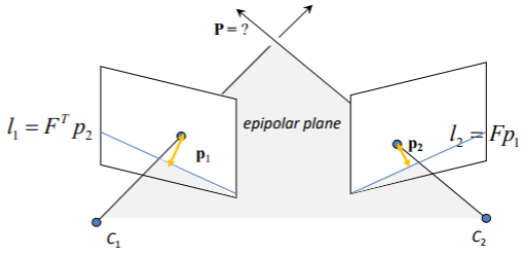
+ However, if the coordinates of the principal points of each camera are known and the two cameras have the same focal length f in pixels, then R, T, f can be determined uniquely.

8.6 Error measures

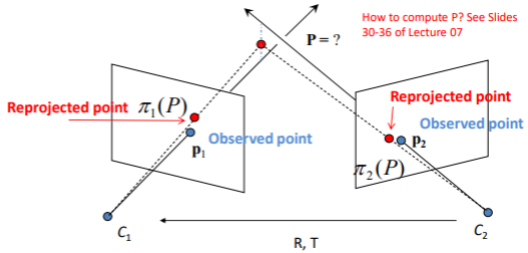
$$\boxed{\sum_{i=1}^{N}(\bar{p}_{2i}^{T}E\bar{p}_{1}^{i})^{2}} \text{ Algebraic Error}$$

$$\boxed{\sum_{i}(cos(\theta_i))^2 \text{ where } \cos(\theta) = \frac{p_2^T \cdot Ep_1}{||p_2||||Ep_1||}} \text{ Directional Error}$$

$$\boxed{\sum_{i=1}^{N} d^2(p_1^i, l_1^i) + d^2(p_2^i, l_2^i)} \text{ Epipolar Line Distance}$$



$$\boxed{\sum_{i=1}^{N} ||p_1^i - \pi_1(P^i)||^2 + ||p_2^i - \pi_2(P^i, R, T)||^2} \text{ Reprojection Error}$$



+ Most popular since it is very accurate.

- Computationally expensive because it requires point triangulation.

8.7 RANSAC (RAndom SAmple Consensus)

Standard method for model fitting in the presence of outliers (very noisy data).

- Choose 2 Points, fit a line.
- Define threshold and count the number of points that support the current hypothesis.
- Iterate and find the maximum number of inliers.
- Computationally very expensive: $N(N-1)/2$.
- Given a percentage of outliers it is enough to only check a subset of all possibilities. The number of iterations needed under that assumption is found as follows:
  - $w :=$ number of inliers/N where N is the number of data points
  - w^2 is the probability that both selected points are inliers.
  - $1 - w^2$ is the probability that both selected points are outliers.
  - k is the number of iterations performed so far.
  - $(1 - w^2)^k$ is the probability that RANSAC has never selected 2 inliers.
  - $p = 1 - (1 - w^2)^k$ is the probability of success. Thus the needed number of iterations based on the needed probability of success is calculated as follows:

$$\boxed{k = \frac{\log(1-p)}{\log(1-w^2)}}$$

- The percentage of outliers can also be adaptively updated.
- RANSAC applied to general model fitting:
  1. Initial: let A be a set of N points.
  2. **repeat**
     a) Randomly select a sample of s points from A.
     b) Fit a model from the s points
     c) Compute the distances of all other points from this model.
     d) Construct the inlier set.
     e) Store the inliers.
  3. **until** maximum number of iterations is reached.
  4. The set with the maximum number of inliers is chosen as a solution to the problem.

$$\boxed{k = \frac{\log(1-p)}{log(1-w^2)}}$$

8.7.1 Application to SFM

1. What is the model?
   - The Essential Matrix (for calibrated cameras) or the Fundamental Matrix (for uncalibrated cameras).
   - Alternatively R and T
2. What is the minimum number of points?
   - We know that 5 points is the theoretical minimum number of points.
   - When using the 8-point algorithm 8 is the chosen minimum.
3. How to compute the distance of a point from the model?
   - Algebraic error ($\bar{p}_2^T E\bar{p}_1 = 0$ or $p_2^T Fp_1 = 0$)

- Directional error
- Epipolar line distance
- Reprojection error

4. The number of iterations needed k increases exponentially with the number of outliers.

$$\boxed{k = \frac{\log(1-p)}{\log(1-(1-\epsilon)^8)}}$$

where ϵ is the fraction of outliers.

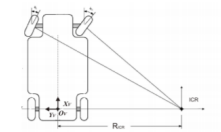
8.7.2 Ackerman's Steering Principle

$$R = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad T = \begin{bmatrix} \rho\cos\phi \\ \rho\sin\phi \\ 0 \end{bmatrix} \quad \text{Planar motion}$$

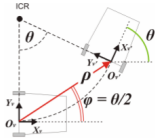
Calculating the essential matrix delivers:

$$\boxed{E = [T]_\times R = \begin{bmatrix} 0 & 0 & \rho\sin\phi \\ 0 & 0 & -\rho\cos\phi \\ -\rho\sin(\phi-\theta) & \rho\cos(\phi-\theta) & 0 \end{bmatrix}}$$

- ρ is a scale factor and can be divided out of the epipolar constraint unless it is zero.
- When applying Ackerman's steering principle ρ, ϕ and θ are determined as follows:



Example of Ackerman steering principle



Locally-planar circular motion

In this case only one point correspondence is needed! This is only valid if the camera lies on the rear axis of the vehicle. Other wise more (2) points are needed.

$$\boxed{E = \begin{bmatrix} 0 & 0 & \rho\sin\frac{\theta}{2} \\ 0 & 0 & \rho\cos\frac{\theta}{2} \\ \rho\sin\frac{\theta}{2} & -\rho\cos\frac{\theta}{2} & \end{bmatrix}}$$

Then using the epipolar constraint for a single correspondence:

$$p_2 T^T E p_1 = 0 \Rightarrow \sin\tfrac{\theta}{2}\cdot(u_2+u_1) + \cos\tfrac{\theta}{2}\cdot(v_2-v_1) = 0$$

which leads to

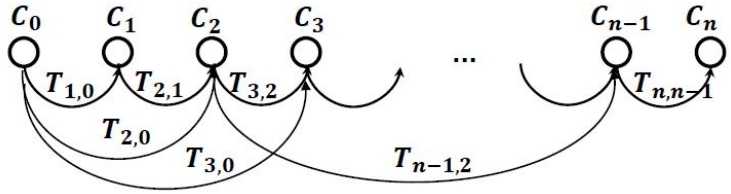
$$\boxed{\theta = -2\tan^{-1}\left(\frac{v_2-v_1}{u_2+u_1}\right)}$$

9 Refinement

9.1 Pose-Graph Optimization

Non-linear refinement of the motion:

$$\boxed{C_k = \arg\min_{C_k} \sum_i \sum_j ||C_i - C_j T_{ij}||^2}$$

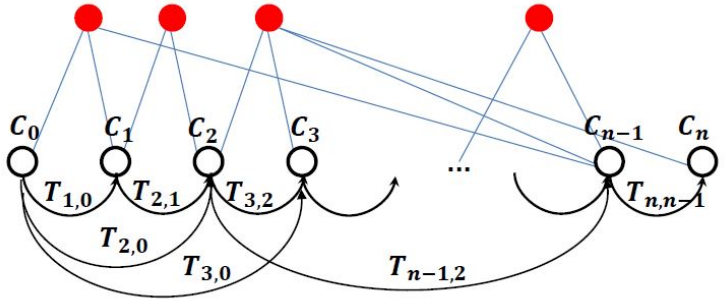


9.2 Bundle Adjustment

Non-linear, simultaneous refinement of structure P^i and motion $C = R, T$.

$$\boxed{(P^i, C_1, C_2) = \arg\min_{P^i, C_1, C_2} \sum_{i=1}^{N} ||p_1^i - \pi_1(P^i, C_1)||^2 + ||p_2^i - \pi_2(P^i, C_2)||^2}$$

where C_i denote the camera poses and π denotes the reprojection point. Use **Levenberg-Marquardt** for minimization, initialize close to the minimum. (Minimizing the sum of squared reprojection errors over each view k).



9.3 BA vs. PGO

- BA is **more precise** than PGO since it uses additional landmark constraints.
- BA is more costly. A possible workaround lies in a smaller window size or motion-only BA.

10 Dense 3D Reconstruction

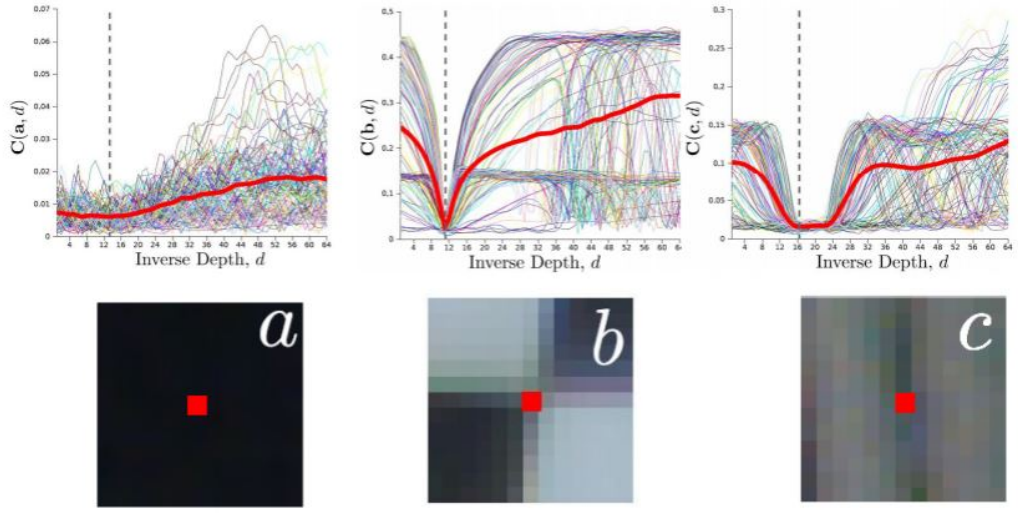
Definition 38. *In constrast to **sparse reconstruction** where a structure is estimated from a sparse set of features, **dense reconstruction** estimates the structure from a dense region of pixels.*

1. Local methods: Estimate depth for every pixel independently.
2. Global methods: Refine the depth surface as a whole by enforcing smoothness constraint.

10.1 Photometric Error (SSD)

Project a ray through the first image and minimize the reprojection error (from projecting the resulting 3D point back into all other pictures) along a varying depth („going along the ray“).

- Minimum patch size for SSD: 1 pixel.
- Patch/Window-size
  - Smaller window
    - \+ More detail
    - \- More noise
  - Larger window
    - \+ Smoother disparity map
    - \- Less detail
- Not all pixels can be matched reliably (Viewpoint and illumination changes, occlusions)
- Take advantage of many small-baseline views where high quality matching is possible.



- Flat regions (a), Edges parallel to the epipolar line (c)
- Distinctive features/blobs show a clear minimum.
- Non-distinctive features will show multiple minima.

10.2 Disparity Space Image (DSI)

Definition 39. *The **disparity space image** is a volumetric 3D grid, that saves the photometric error w.r.t. the reference image for discrete depth hypotheses.*

$$C(u,v,d) = \sum_{k=R+1}^{R+n-1} \rho(I_R(u,v) - I_k(u',v',d))$$

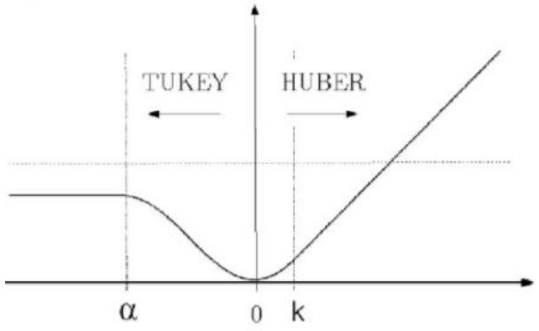
n	number of images considered
$I_k(u',v',d)$	patch of intensity values in k-th image
$\rho(\cdot)$	photometric error (L_1, L_2, Tuckey, Huber)

$$I_k(u',v',d) = I_k(\pi(T_{k,R}(\pi^{-1}(u,v)\cdot d)))$$

The solution to the depth estimation problem thus lies in the finding of a function $d(u,v)$ that minimizes the aggregated photometric error (local methods) and is piecewise smooth (global methods).

10.2.1 Tuckey and Huber

Goal: Penalize the influence of wrong matches.



$$\rho(x) = \begin{cases} x^2 & \text{if } |x| \le k \\ k(2|x| - k) & \text{if } |x| \ge k \end{cases}$$ Huber norm

$$\rho(x) = \begin{cases} \alpha^2 & \text{if } |x| \ge \alpha \\ \alpha^2 \left(1 - \left(1 - \left(\frac{x}{\alpha}\right)^2\right)^3\right) & \text{if } |x| \le \alpha \end{cases}$$ Tuckey norm

10.3 Global Methods

The objective is to find a surface $d(u, v)$ that minimizes a global energy

$$E(d) = \underbrace{\sum_{(u,v)} C(u, v, d(u, v))}_{E_d(d)\text{: Data term}} + \underbrace{\lambda \sum_{(u,v)} \left(\frac{\partial d(u,v)}{\partial u}\right) + \left(\frac{\partial d(u,v)}{\partial v}\right)}_{\lambda E_S(d)\text{: Regularization term}}$$

where λ controls the tradeoff data/regularization.

10.3.1 Scene depth discontinuities

Assumption: Depth discontinuities coincide with intensity discontinuities (i.e. image gradients)
Solution: Control/weigh regularization term according to image gradient:

$$E_s(d) = \sum_{(u,v)} \left(\frac{\partial d(u,v)}{\partial u}\right)^2 \rho_I \left(\frac{\partial I(u,v)}{\partial u}\right)^2 + \left(\frac{\partial d(u,v)}{\partial v}\right)^2 \rho_I \left(\frac{\partial I(u,v)}{\partial v}\right)^2$$

where ρ_I is some monotically decreasing function of image gradients. High for small image gradients and low for high image gradients, such that the data term dominates in regions where the image has discontinuities.

10.3.2 Baseline

The smaller the baseline the larger the depth error, the larger the baseline the harder the search problem, due to wide view point changes.
Solution: Obtain depth map from small baselines, when baseline becomes large (e.g. $>$ 10% of the average scene depth) **create a new reference frame** (keyframe) and start a new depth map computation. The multiple depth maps can then be fused into one.

10.4 GPU

Definition 40. *A **GPU** performs calculations in parallel on thousands of cores where a **CPU** has only a few cores that are optimized for serial processing.*

GPU capabilities

- Fast fixel processing (Ray tracing, draw textures, shaded triangles)
- Fast matrix/vector operations (Transform vertices)
- Programmable (Shading, bump mapping)
- Floating-point support (Accurate computations)
- Deep Learning

GPU capabilities suitable for 3D Dense Reconstruction:

- Image processing
  - Filtering and Feature extraction (convolutions)
  - Warping (epipolar rectification, homography)
- Multiple-View Geometry
  - Search for dense correspondences
    - Pixel-wise operations (SAD, SSD, NCC)
    - Matrix and vector operations (epipolar geometry)
  - Aggregated Photometric Error for multi-view stereo
- Global Optimization
  - Variational methods (regularization)
    - Parallel, in place operations for gradient / divergence computation.

11 Tracking

11.1 Point Tracking

11.1.1 Block Matching

\+ Works well if the motion is large

\- Can become computationally demanding if the motion is large

E Differential Methods

Assumptions

1. **Photo consistency** The intensity of the pixels around the point to track in image I_0 should be the same of its corresponding pixels in image I_1
2. **Temporal persistency**
   The motion between the two frames must be small (1-2 pixels at the most).
3. **Spatial coherency**
   Neighbouring pixels that belong to the same surface have similar motion.

Approach

1. We want to find the motion vector (u, v) that mimizes the SSD.
   $$E = SSD = \sum(\Delta I - I_x u - I_y v)^2$$
2. Minimize the E
   $$\frac{\partial E}{\partial u} = 0 \Rightarrow -2 \sum I_x(\Delta I - I_x u - I_y v) = 0$$
   $$\frac{\partial E}{\partial v} = 0 \Rightarrow -2 \sum I_y(\Delta I - I_x u - I_y v) = 0$$
3. This system can be written in matrix form

$$\begin{bmatrix}\sum I_xI_x & \sum I_xI_y \\ \sum I_xI_y & I_yI_y\end{bmatrix}\begin{bmatrix}u\\v\end{bmatrix} = \begin{bmatrix}\sum I_x\Delta I \\ \sum I_y\Delta I\end{bmatrix} \Rightarrow \begin{bmatrix}u\\v\end{bmatrix} = \underbrace{\begin{bmatrix}\sum I_xI_x & \sum I_xI_y \\ \sum I_xI_y & I_yI_y\end{bmatrix}^{-1}}_{M}\begin{bmatrix}\sum I_x\Delta I \\ \sum I_y\Delta I\end{bmatrix}$$

4. For M to be invertible $\det(M)$ should be non-zero, which is given for not-flat regions of the image. For this reason good features should be selected!

Definition 41. ***Optical flow*** *is a vector field describing the motion of pixels from one frame to the next.*

Definition 42. *The* ***Aperture Problem*** *refers to the fact that a small aperture, when for example using block-based methods, cannot always determine the motion of an edge, since multiple solutions exist. A possible solution in the enlargement of the aperture.*

11.1.2 Block-based vs. Differential Methods

- Block-based methods
  - \+ Robust to large motions.
  - \- Can be computationally expensive (D × D validations need to be made for a single point to track)
- Differential methods
  - \- Works only for small motions (e.g. high frame rate). For larger motion, multi-scale implementations are used but are more expensive.
  - \+ Much more efficient than block-based methods

11.2 Common 2D Transformations

Transformation	Projection	2D Warping
Translation	$x' = x + a_1$ $y' = y + a_2$	$W(x,p) = \begin{bmatrix}1 & 0 & a_1\\ 0 & 1 & a_2\end{bmatrix}\begin{bmatrix}x\\y\\1\end{bmatrix}$
Euclidean	$x' = x\cos(a_3) - y\sin(a_3) + a_1$ $y' = x\sin(a_3) + y\cos(a_3) + a_2$	$W(x,p) = \begin{bmatrix}\cos(a_3) & -\sin(a_3) & a_1\\ \sin(a_3) & \cos(a_3) & a_2\end{bmatrix}\begin{bmatrix}x\\y\\1\end{bmatrix}$
Affine	$x' = a_1x + a_3y + a_5$ $y' = a_2x + a_4y + a_6$	$W(x,p) = \begin{bmatrix}a_1 & a_3 & a_5\\ a_2 & a_4 & a_6\end{bmatrix}\begin{bmatrix}x\\y\\1\end{bmatrix}$
Projective	$x' = \frac{a_1x + a_2y + a_3}{a_7x + a_8y + 1}$ $y' = \frac{a_4x + a_5y + a_6}{a_7x + a_8y + 1}$	$W(x,p) = \begin{bmatrix}a_1 & a_2 & a_3\\ a_4 & a_5 & a_6\\ a_7 & a_8 & 1\end{bmatrix}\begin{bmatrix}x\\y\\1\end{bmatrix}$

For the Lucas-Kanade tracker we need the gradients of some of the above:

$$\nabla W_p = \begin{bmatrix}\frac{\partial W_1}{\partial a_1} & \frac{\partial W_1}{\partial a_2}\\ \frac{\partial W_2}{\partial a_1} & \frac{\partial W_2}{\partial a_2}\end{bmatrix} = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} \text{ (Translation)}$$

$$\nabla W_p = \begin{bmatrix}1 & 0 & -x\sin(a_3) & -y\cos(a_3)\\ 0 & 1 & x\cos(a_3) & -y\sin(a_3)\end{bmatrix} \text{ (Euclidean)}$$

$$\nabla W_p = \begin{bmatrix}x & 0 & y & 0 & 1 & 0\\ 0 & x & 0 & y & 0 & 1\end{bmatrix} \text{ (Affine)}$$

11.2.1 Template Tracking

Definition 43. *Following a template image in a video sequence by estimating the warp to enable matching is called* ***template tracking****.*

Thus we'd like to find the set of warp parameters $\underline{p}$ such that

$$I(W(\underline{x},\underline{p})) = T(\underline{x})$$

This is solved by determining $\underline{p}$ that minimizes the SSD.

$$\boxed{E = SSD = \sum_{\underline{x}\in T}\left[I(W(\underline{x},\underline{p})) - T(x)\right]^2}$$

Assumptions

- No errors in the template image boundaries.
- No occlusion
- Brightness constancy: The pixel intensity should not change much from one frame to the next.
- Temporal consistency: The motion between two frames should be small.
- Spatial coherency: Pixels belonging to the same surface should have a similar motion.

11.3 Lucas-Kanade Tracker

$$E = \sum_{\underline{x}\in \underline{T}}\left[I(W(\underline{x},\underline{p})) - T(\underline{x})\right]^2$$

Idea:

- Assume that an initial estimate of $\underline{p}$ is known. From that we'd like to find the increment $\Delta\underline{p}$ that minimizes the objective above.
- Thus a first order Taylor approximation is made:

$$I(W(\underline{x},\underline{p}+\Delta\underline{p})) \approx I(W(\underline{x},\underline{p})) + \nabla I\frac{\partial W}{\partial \underline{p}}\Delta\underline{p}$$

- The minimum is then found by differentiating w.r.t. $\Delta\underline{p}$ and equating to zero:

$$0 = \frac{\partial E}{\partial \Delta \underline{p}}$$

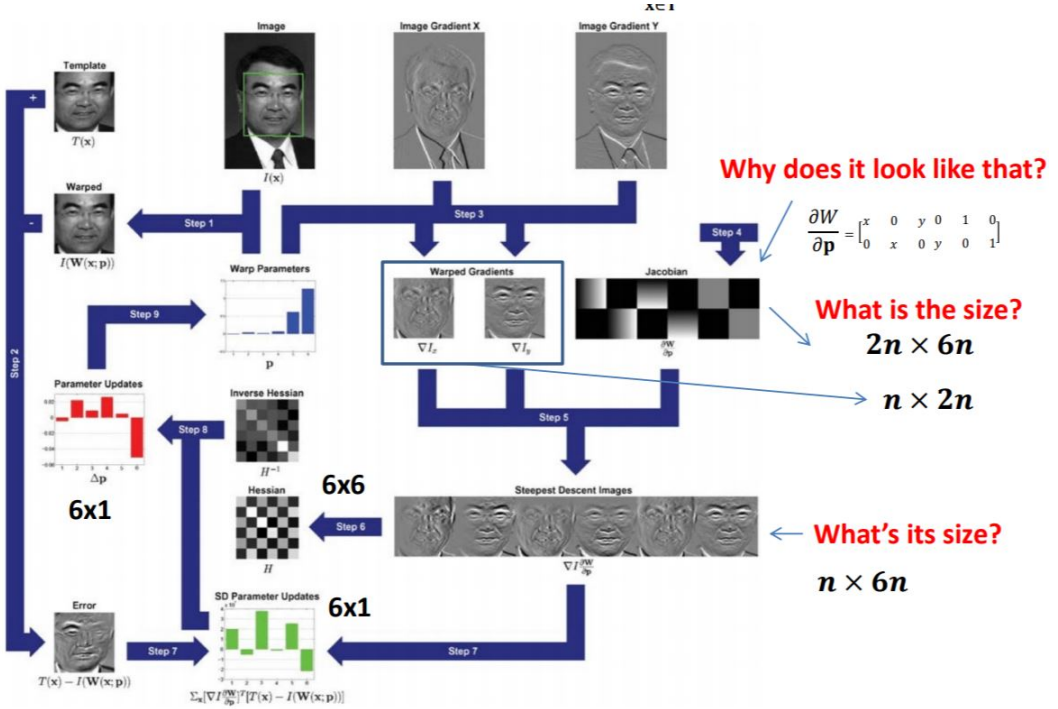
$$0 = 2 \sum_{\underline{x} \in \underline{T}} \left[\nabla I \frac{\partial W}{\partial \underline{p}} \right]^T \left[I(W(\underline{x}, \underline{p})) + \nabla I \frac{\partial W}{\partial \underline{p}} \Delta \underline{p} - T(\underline{x}) \right]$$

$$\boxed{\Delta \underline{p} = H^{-1} \sum_{\underline{x} \in \underline{T}} \left[\nabla I \cdot \frac{\partial W}{\partial \underline{p}} \right]^T \left[T(\underline{x}) - I(W(\underline{x}, \underline{p})) \right] \qquad H = \sum_{\underline{x} \in \underline{T}} \left[\nabla I \cdot \frac{\partial W}{\partial \underline{p}} \right]^T \left[\nabla I \cdot \frac{\partial W}{\partial \underline{p}} \right]}$$

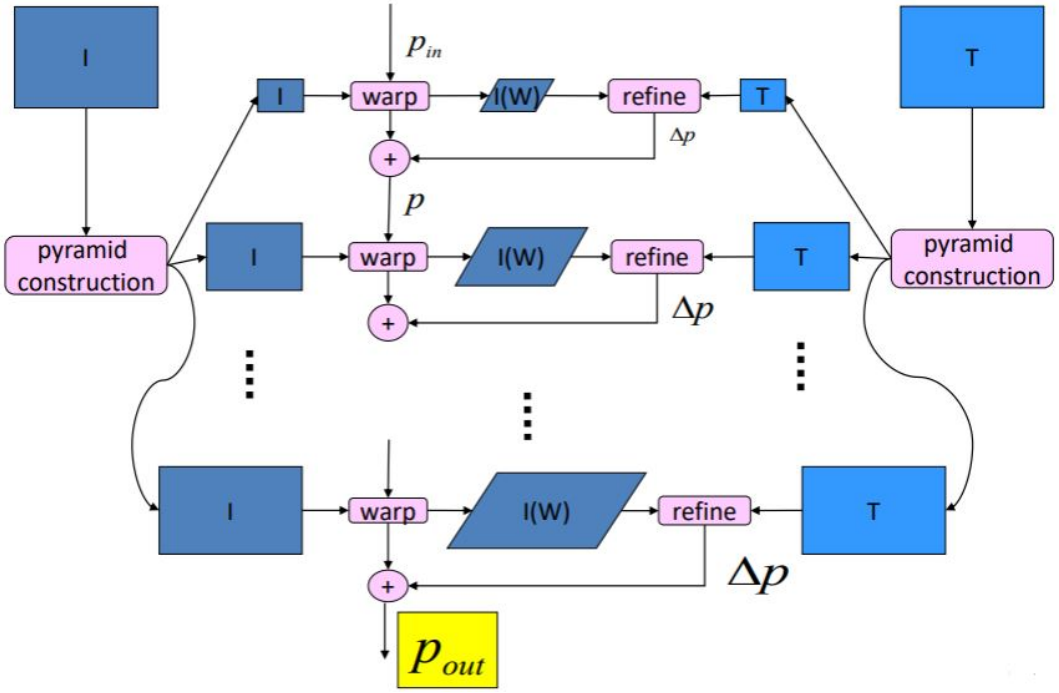
where $\cdot$ denotes a pixel-wise product! H is called the second moment matrix (Hessian) of the warped image.

Algorithm:

1. Warp $I(\underline{x})$ with $W(\underline{x}, \underline{p}) \rightarrow I(W(\underline{x}, \underline{p}))$.
2. Compute the error: $T(x) - I(W(\underline{x}, \underline{p}))$.
3. Compute warped gradients: $\nabla I = [I_x, I_y]$ evaluated at $W(\underline{x}, \underline{p})$.
4. Evaluate the Jacobian of the warping $\frac{\partial W}{\partial \underline{p}}$.
5. Compute the steepest descent $\nabla I \frac{\partial W}{\partial \underline{p}}$.
6. Compute inverse Hessian $H^{-1} = \left[\sum_{\underline{x} \in \underline{T}} \left[\nabla I \frac{\partial W}{\partial \underline{p}} \right]^T \left[\nabla I \frac{\partial W}{\partial \underline{p}} \right] \right]^{-1}$.
7. Multiply steepest descent with error: $\sum_{\underline{x} \in \underline{T}} \left[\nabla I \frac{\partial W}{\partial \underline{p}} \right]^T \left[T(\underline{x}) - I(W(\underline{x}, \underline{p})) \right]$
8. Compute $\Delta \underline{p}$.
9. Update parameters $\underline{p} \leftarrow \underline{p} + \Delta \underline{p}$.
10. Repeat until $\Delta \underline{p} < \underline{\epsilon}$.



To get an estimate of the warp parameters the image is downsampled to form an image pyramid, for which at lower levels Δp can be found more easily.



11.3.1 Failure Cases

- Initial estimate too far from the minimum, thus the approximation no longer holds. The image pyramid is the solution. **Coarse-To-Fine**
- The template changes over time, thus we update it with every newly tracked image.

11.3.2 Generalization

The concept can be generalized to 3D templates. In this case the transformation to be estimated would be a rigid body movement of the considered template model, which for matching would be projected onto 2D and varied around the estimated position to increase the chance of matching (Particle Filter).

11.4 Tracking by detection of local image features

1. Keypoint detection and matching (invariant to scale, rotation or perspective)
2. Geometric verification (RANSAC)

11.5 Tracking Issues

- How to segment the object to track from the background?
- How to initialize the warping?
- How to handle occlusions?
- How to handle illumination changes and non modelled effects?

12 Recognition

The complexity of finding image in a database of N images when comparing M features is NM^2.

To reduce the complexity an inverted file index is used. Thus for every „word“the images that contain that word are registered.

Definition 44. *An **inverted file index** lists all occurrences of a word for each word.*

How to define a visual „word“?

1. Collect a large enough dataset that is representative of all possible features.
2. Extract features and descriptors from each image and map them into the same descriptor space. (For SIFT map to a 128-dimensional space).
3. Cluster the descriptor space into K clusters.
4. The centroid of each cluster is a visual word.

12.1 K-means clustering

- Initialize k cluster centers.
- Minimize $D(X, M) = \sum_{i=1}^{k} \sum_{x \in S_i} (x - m_i)^2$
  - Assign each data point x_j to the nearest center m_i.
  - REcompute each cluster center as the mean of all points assigned to it.

12.2 Applying k-means to Image Retrieval

1. Inverted File Index lists all visual words in the vocabulary (extracted at training time)
2. Voting array: has as many cells as the images in the DB. Each word in the query image votes for all images registred in the IFI.

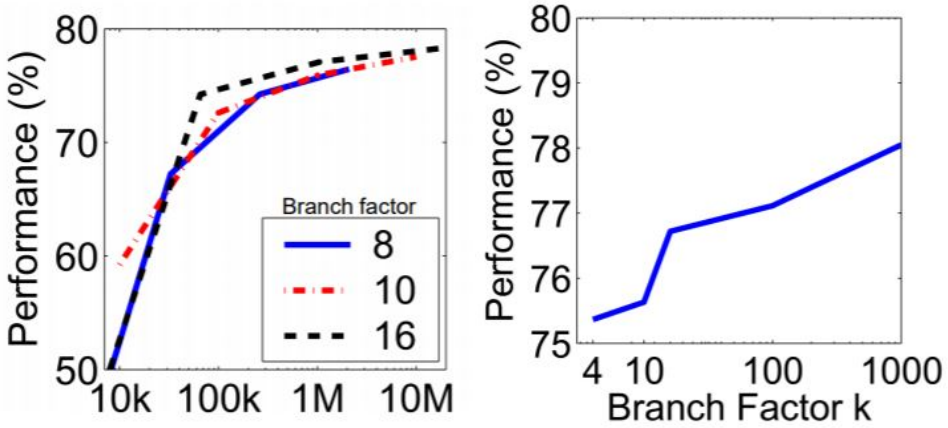
12.3 Hierarchical Clustering

- Build a hierarchical means-tree, defining clusters and sub-clusters.
- Search in super-clusters first, then refine the search.
- This reduces the number of needed comparisons greatly.

12.4 Robust Object/Scene Recognition

- Visual Vocabulary discards the spatial relationships between features.
- Thus retain the h most similar images and use a 5- or 8-point algorithm to geometrically verify the image, searching for the smallest reprojection error.

12.5 Performance Analysis

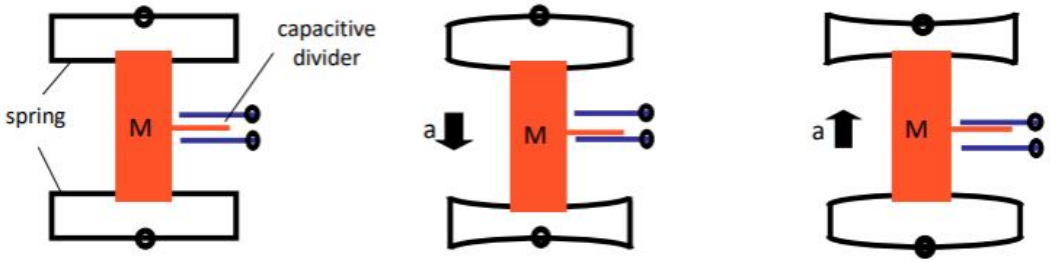


- The higher the word count the better the performance.
- The higher the branch factor, the better the performance (but slower).

13 Visual Inertial Fusion

Definition 45. *An **Inertial Measurement Unit (IMU)** measures angular velocities (gyroscope) and linear accelerations (accelerometer).*

MEMS (Micro Electro-Mechanical Systems



- The MEMS Accelerometer works with a mass in a capacitive divider.
- The MEMS Gyroscope is using a principle similar to halteres of flies (which are two small oscillating organs which change their relative phase when the flies' body rotates). In practice this is implemented using piezoelectric crystals.
- IMU integration leads to large errors since there is always drift.
  - The error in velocity (integrated from accelerometer) is proportional to t.
  - The error in position is proportional to t^2.
  - In addition the position also depends on orientation!

Cameras	IMU
Precise in slow motion	Robust
Rich information for other purposes	High output rate ($\sim 1000\,\text{Hz}$)
	Accurate at high acceleration
Limited output rate ($\sim 100\,\text{Hz}$)	
Scale ambiguity in monocular setup	Large relative uncertainty when at low acceleration/angular velocity
Lack of robustness	Ambiguity in gravity/acceleration

Both do dead-reckoning, which suffers from drifting. **Solution: loop detection and loop closure.**

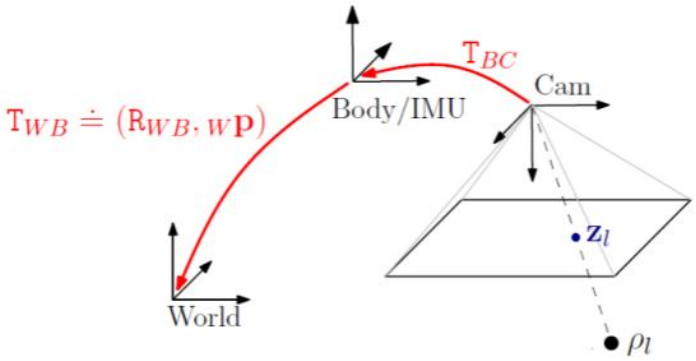
13.1 IMU model

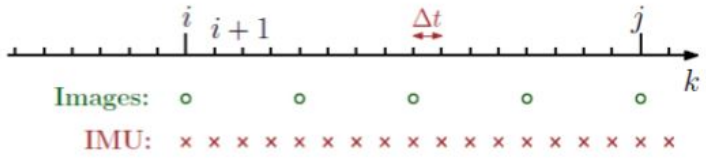
$$
\begin{aligned}
{}_B\tilde{\underline{\omega}}_{WB}(t) &= {}_B\underline{\omega}_{WB}(t) + \underline{b}^g(t) + \underline{n}^g(t) \\
{}_B\tilde{\underline{a}}_{WB}(t) &= \underline{R}_{BW}(t)({}_W\underline{a}_{WB}(t) - {}_w\underline{g}) + \underline{b}^q(t) + n^a(t)
\end{aligned}
$$

where superscript g stands for gyroscope and a stands for accelerometer.

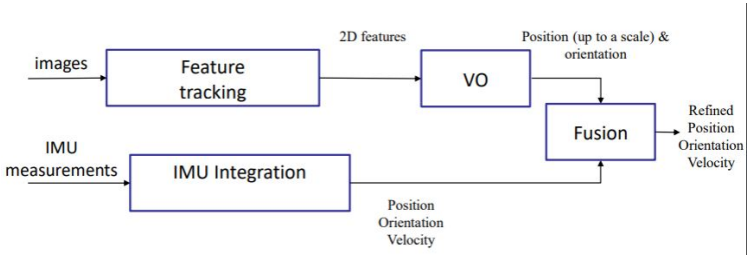
$$
\boxed{\underline{p}_{W_{t_1}} = \underline{p}_{W_{t_1}} + (t_2 - t_1)\underline{v}_{W_{t_1}} + \iint_{t_1}^{t_2} \left(\underline{R}_{W_t}(t)\left(\tilde{\underline{a}}(t) - \underline{b}^a(t)\right) + {}_w\underline{g}\right) dt^2} \quad \text{IMU Integration}
$$

13.2 Camera IMU System





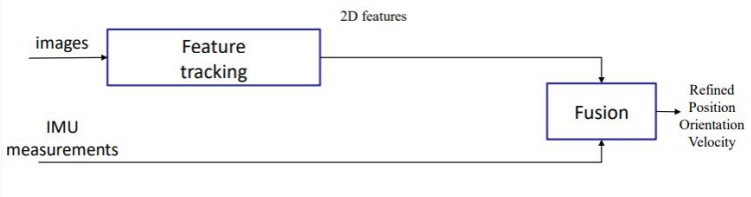
13.3 Loosely Coupled Approach



- The loosely coupled approach fails to model cross-correlations between internal states of the different measurement devices.

$$\boxed{\underline{X} = \left[{}_w\underline{p}(t);\ \underline{q}_{WB}(t);\ {}_w\underline{v}(t);\ \underline{b}^a(t);\ \underline{b}^g(t)\right]} \text{ System states}$$

13.4 Tightly Coupled Approach



- Much less drift than the loosely coupled approach.

$$\boxed{\underline{X} = \left[{}_w\underline{p}(t);\ \underline{q}_{WB}(t);\ {}_w\underline{v}(t);\ \underline{b}^a(t);\ \underline{b}^g(t);\ {}_w\underline{L}_1;\ {}_w\underline{L}_2;\ \ldots;\ {}_w\underline{L}_k\right]} \text{ System states}$$

${}_w\underline{p}(t)$	position of the IMU in world coordinates
$\underline{q}_{WB}(t)$	attitude of the IMU
${}_w\underline{v}(t)$	velocity of the IMU in world coordinates
$\underline{b}^a(t)$	bias of the accelerometer
$\underline{b}^g(t)$	bias of the gyroscope
${}_w\underline{L}_i$	Landmarks

13.5 Closed Form Solution

1. Absolute pose x is known up to scale: $x = s\tilde{x}$
2. Equate that with the information from the IMU:

$$s\tilde{x} = x_0 + v_0(t_1 - t_0) + \iint_{t_0}^{t_1} a(t)dt^2$$

3. For 6DOF (Martinelli 14), both s and v_0 can be determined in closed form from a single feature observation and 3 views. This can be generalized for N features.

13.6 Filtering

Filtering	Fixed-lag Smoothing	Full smoothing
Only updates the most recent state • (e.g., extended Kalman filter)	Optimizes window of states • Marginalization • Nonlinear least squares optimization	Optimize all states • Nonlinear Least squares optimization
×1 Linearization ×Accumulation of linearization errors ×Gaussian approximation of marginalized states ✓Fastest	✓Re-Linearize ×Accumulation of linearization errors ×Gaussian approximation of marginalized states ✓Fast	✓Re-Linearize ✓Sparse Matrices ✓Highest Accuracy ×Slow (but fast with GTSAM)

Problems:

- Wrong linearization point: Linearization depends on the current estimates of states which might be erroneous.
- Complexity of the EKF grows quadratically in the number of estimated landmarks.

An alternative is MSCKF which keeps a window of recent states and updates them using EKF. Visual obersvations are incorporated without including point positions into the states.

13.6.1 Smoothing

VIO can be solved as a graph optimization over a set of robot states $X = \{x_1, \ldots, x_N\}$ and 3D Landmarks $L = \{l_1, \ldots, l_N\}$.

$$x_k = f(x_{k-1}, u) \quad \text{State Transition Function}$$

$$z_{i_k} = \pi(x_k, l_i) \quad \text{Reprojection of the Landmark}$$

Then the optimization problem to be solved is:

$$\boxed{\{X, L\} = \text{argmin}_{\{X,L\}} \left\{ \sum_{k=1}^{N} ||f(x_{k-1}, u) - x_k||^2_{\Lambda_k} + \sum_{k=1}^{N} \sum_{i=1}^{M} ||\pi(x_k, l_i) - z_{i_k}||^2_{\Sigma_{i_k}} \right\}}$$

where Λ_k is the covariance from the IMU integration and Σ_{i_k} is the covariance from the noise 2D measurements.

13.6.2 Full smoothing

Solves the same optimization problem as smoothing, but **keeps all the frames**.
To make the optimization more efficient:

- Only keyframes are used, making the graph sparser.
- IMU data is pre-integrated between keyframes.
- Factor Graphs: Only frames affected by a new observations are optimized.

13.7 Unsolved Problems

- Filters
  - Linearization around different values of the same variable may lead to error.
- Smoothing methods
  - May get stuck in local minima.

13.8 Camera-IMU Calibration

The goal is to estimate the rigid body transformation T_{BC} and the delay t_d between a camera and a rigidly attached IMU. Assume that the intrinsics of the camera are known.

Available data:

- Image point of detected calibration pattern (checkerboard)
- IMU measurements: accelerometer $\{a_k\}$ and gyroscope $\{\omega_k\}$.

Approach: Minimization

$$J(\theta) := J_{feat} + J_{acc} + J_{gyro} + J_{bias_{acc}} + J_{bias_{gyro}}$$

(Feature reprojection error)

$$\sum_k (a_{IMU}(t_k - t_d) - a_{cam}(t_k))^2$$

$$\sum_k (w_{IMU}(t_k - t_d) - w_{cam}(t_k))^2$$

$$\int ||\frac{db_{acc}}{dt}(u)||^2 du$$

$$\int ||\frac{db_{gyro}}{dt}(u)||^2 du$$

Where the quantities we'd like to identify are: $T_{BC}, t_d, g_w, T_{WB}, b_{acc}8t), b_gyro(t)$.
Also continuous time can be modelled using spline interpolation.

14 Event Based Vision

Challenges in computer vision:

- Latency
- Motion Blur
- Dynamic Range

Event cameras do not suffer from these issues!

+ Low latency ($\sim 1\,\mu s$)
+ High dynamic range (HRD) (140 dB instead of 60 dB)
+ High update rate (1 MHz)
+ Low power (10 mW instead of 1 W)

- Paradigm shift requires fundamentally new vision algorithms:
  - Asynchronous pixels
  - No intensity information

$$\boxed{\left\langle t; \langle x, y\rangle, \text{sign}\left(\frac{dI(x,y)}{dt}\right)\right\rangle} \text{ Event}$$

t	timestamp
$\langle x, y\rangle$	Pixel coordinates
$\text{sign}\left(\frac{dI(x,y)}{dt}\right)$	Pixel polarity, increase or decrease in brightness

Note that an event is sampled when a certain level is crossed, in other words a logarithmic brightness changes has happened.

$$\boxed{|\Delta \log I| = |\log I(t + \Delta t) - \log I(t)| = C} \text{ Logarithmic Brightness Increment}$$

$C \in [0.15, 0.2]$	Contrast sensitivity
$\Delta \log I = C$	ON event
$\Delta \log I = -C$	OFF event

14.1 Comparison to existing high-speed imaging technology

	Photron Fastcam SA5	Matrix Vision Bluefox	DVS
Max fps or measurement rate	1MHz	90 Hz	**1MHz**
Resolution at max fps	64x16 pixels	752x480 pixels	**346x260 pixels**
Bits per pixels	12 bits	8-10	**1 bits**
Weight	6.2 Kg	30 g	**30 g**
Active cooling	yes	No cooling	**No cooling**
Data rate	1.5 GB/s	32MB/s	**~1MB/s on average**
Power consumption	150 W + llighting	1.4 W	**10 mW**
Dynamic range	n.a.	60 dB	**140 dB**

35

14.2 Calibration

+ Pinhole camera model still works

- Passive calibration patterns cannot be used
  - – Camera motion or blinking patterns required.

14.3 Event Processing

- Event-by-event processing
  - + Low latency (in the order of microseconds)
  - - High speed motion provokes high amount of data.
- Event-packet processing
  - + N can be tuned to allow real-time performance on a CPU.
  - - No longer microsecond resolution.

14.4 Event Generation Model

- Assume $I(x, y, t) = \log(I(x, y, t))$
- Consider a given pixel $p(x, y)$ with gradient $\nabla I(x, y)$ undergoing the motion $\underline{u} = (u, v)$ in pixels, induced by a moving 3D point $\underline{P}$.
- An event is generated if the scalar product between the gradient vector $\nabla I(x, y)$ and the apparent motion vector $\underline{u}(u, v)$ is equal to C.

$$\boxed{-\nabla I \cdot \underline{u} = C}$$ Event Generation Condition

The above can be derived based on the assumption that the brightness change of a pixel corresponding to a 3D point that performs a small motion remains unchanged. Then the intensities of the two pixels representing that same 3D point at times t and $t + \Delta t$ can be equated:

$$I(x, y, t) = I(x + u, y + v, t + \Delta t)$$
$$I(x, y, t) = I(x, y, t + \Delta t) + \frac{\partial I}{\partial x} u + \frac{\partial I}{\partial y} v$$
$$\Delta I = C = -\nabla I \cdot \underline{u}$$

14.5 DAVIS Sensor

Dynamic and Active-pixel VIsion Sensor

- Combines an event sensor (DVS) and a standard camera in the same pixel array.
- Output: frames (30 Hz) and events (asynchronous)

14.6 Applications

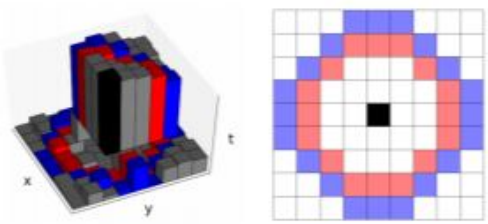
14.6.1 Image Reconstruction

1. Challenge: Recover absolute brightness based on events and camera motion.
2. Relate the event-creating edges to their positions in the real world.
3. Integrate the resulting gradient map to recover the absolute brightness of the scene. (Poisson Equation)

14.6.2 6-DoF Pose Tracking from a Photometric Depth Map

1. Probabilistic approach (Bayesian filter) $p(s|e) = p(e|s)p(s)$
2. State vector: $s = (R, T, C, \sigma_C, \rho)$
   - pose (R, T)
   - contrast mean value C
   - uncertainty σ_C
   - inlier ratio ρ
3. Motion model: random walk.
4. Robust sensor model (likelihood)

14.6.3 Event-based Corner Detection



1. Operates on surface of active events.
2. The event is considered a corner if
   - 3-6 contiguous pixels on red ring are newer than all other pixels on the same ring.
   - 4-6 contiguous pixels on the blue ring are newer than all other pixels on the same ring

15 Matlab

15.1 Apply Gauss Filter to Image

```
hsize = 20;
sigma = 5;
h = fspecial('gaussian',hsize,sigma);
mesh(h); %To show the filter in 3D
imagesc(h); %To show the filter in 2D
im = imread('panda.jpg');
outim = imfilter(im,h);
imshow(outim);
```

Understanding Checks

Definition **Derivation** **Comparison** **Application**

Basics

Visual Odometry

- The process of **incrementally** estimating the **pose** of the vehicle by examining the changes that motion induces on the images of its **onboard cameras** in **real time**.

VO, VSLAM, SFM

- Structure from motion describes the **3D reconstruction** and **6DOF pose estimation** from **unordered image sets**. By this it is a **superset of VSLAM** which concentrates on localization and mapping based on ordered images sets. It basically performs visual odometry enhanced with **loop detection** and **graph optimization** which serves to build an accurate **map**.

Needed Assumptions for VO

- Sufficient illumination
- Dominance of static scene
- Enough texture
- Sufficient scene overlap

Working Principle / Building Blocks of VO

- Working Principle
  1. Compute the **relative motion** from the last to the current image.
  2. Concatenate pose changes over time to **recover the trajectory**.
  3. Optimize over a set of poses to **refine the trajectory** locally.
- Building Blocks
  1. Recorded **image sequence**
  2. Feature **detection**
  3. Feature **matching** across frames
  4. **Motion estimation** from 2D-2D, 3D-3D or 2D-3D correspondences
  5. Local **optimization**

Dense, Semi-Dense, Sparse Methods

- Dense methods use **every pixel** in the frame and typically do not rely on features.
- Semi-Dense methods only consider **part of the pixels**. In the case of LSD-SLAM the pixels are selected based on a **variance threshold**. Alternatively also a **threshold on the gradient** could be used.
- Sparse methods rely on the direct comparison of certain sparsely chosen areas on the frame, for example areas **corresponding to features**, as done in SVO.

Image Formation

Blur circle

- The blur circle describes the **projection of a point light source** when the lens is **not focused**.

Thin lens equation including the Pinhole Approximation

- Sketch object-, lens- and image-plane including the corresponding distances and the focal point.
- Derive the formula using **similar triangles**.
- The pinhole approximation assumes that the **object is far away from the camera**. Therefore $z \gg f$ and $z \gg L$. Thus $f \approx e$.

Vanishing Points and Lines

- Parallel lines are at the **same distance along their course**. If however they are not parallel to the image plane, and thus their depth changes over their course, the perceived **distance between the lines decreases proportionally with the depth** until the two lines intersect in the vanishing point. Two **parallel planes intersect in their vanishing line**. This can be easily shown choosing pairs of parallel lines in those planes.
- This can be shown using the **equations of two parallel lines** and applying the **perspective projection**.

Ames Room

- An Ames room is a Room which allows introspection only through a **single defined peephole** (only one eye!), from which it seems to be perfect cuboid (no depth perception due to **monocular vision**). Unexpectedly though it is built such that **one corner extends further back**. If a person now walks along the back side of that room they appear to remain at constant distance while changing size, even though they actually move away or towards the viewer.
- The room has to have a **trapezoidal back wall**, **one corner further back** and a **sloped floor**.

Relation between FOV and Focal Length

- Sketch image plane in the context of the pinhole approximation and use the tangent of the field-of-view-angle to establish a relation between the focal length and the width of the image plane.

Perspective Projection with Homogeneous Transform and Lens Distortion Removal

- Perspective Projection
  1. Use a homogeneous transformation to calculate the the **camera-frame coordinates** of a desired point in world coordinates.
  2. Use the **K-matrix** to find the corresponding **image plane coordinates** and transform them to **pixel coordinates** in one go.
  3. Remember that k_u and k_v establish the connection between the number of pixels and the focal length in meters.
- Lens Distortion Removal
  1. Asses the lens distortion and calculate the **warping** that describes where every pixel of the unwarped picture goes when affected by the warping in this case.
  2. Allocate an unwarped (**destination**) image and for every pixel in it calculate where the corresponding (**source**) warped pixel must lie. Then interpolate that pixel in the source image using **next neighbour** or **bilinear interpolation**.

Image Formation 2

PnP Problem, Derivation of the Behaviour of the Solutions

- Problem Description: **Find the pose** of the camera from n **2D-3D-correspondences**.
- For a **single point** correspondence neither position nor orientation of the camera are determined.
- For **two points** the camera still has infinitely many solutions, since size and orientation of the line are unknown. (**Thales' Circle**)
- For **three points Carnot's theorem** can be used to establish a set of equations connecting the bearing angles between points to their relative distances and their distances from the camera. The resulting system of equations is of eighth order and reduces to fourth order when negative solutions are neglected. Then a **forth point is needed to disambiguate** the remaining solutions.

DLT

- The idea of DLT is to use a set of 2D-3D-correspondences to determine **extrinsic and intrinsic parameters** of a camera.
- Approach

1. Define the **projection matrix** as the multiplication of the K matrix and the homogeneous transform which in combination describe the projection of a 3D point to 2D image coordinates.
2. Rearrange the resulting equation **solving for image coordinates** u and v.
3. Put all terms to the right side and **write in matrix form**.
4. The resulting system has **12 unknowns** and can thus be solved up to scale with 11 given correspondences, thus **at least 6 points** are needed.

- Degenerate configurations are **collinear**, **coplanar** points, **passing through the projection center** or if the camera and the points are on a **twisted cubic**.

Central and Non Central Omnidirectional Cameras

- A central omnidirectional camera has a **single effective viewpoint**, thus the rays recorded in the camera do effectively meet in a single point.

Equivalence between perspective and omnidirectional model

- Both, the omnidirectional camera view and the perspective camera view can be projected onto a unit sphere, allowing the use of unified algorithms.

Which Mirrors ensure Central Projection

- Mirrors resulting from the **revolution of a conic** can be used for central omnidirectional cameras. The advantage of such a camera is that the image can actually be unwarped to give a perspective view. Or alternatively project the view onto a unit sphere making an interface for standardized algorithms.

Photometric Calibration

- **Radiometric response function:** What is the **brightness recorded depending on the irradiance** of the scene?
- **Vignetting:** How much darker do the **image corners** appear?
- **Point spread function:** Describes how the imaging system **responds to a point source** or a point object. Can be associated with the **impulse response**, returns the image when convoluted with the input.

Superposition of a virtual object on an image given camera pose

- Given the 3D coordinates of the object, find the corresponding camera frame 3D coordinates.
- Multiply with the K-matrix to find pixel coordinates and plot.

Filtering

Convolution and Cross-correlation

- The convolution of two sequences is calculated by sliding the flipped filter over the target sequence and multiplying all coinciding elements. In contrast the **cross-correlation does not include flipping the filter** first. For a symmetric filter the two are equivalent.
- Convolution: Linear, **Associative**, **Commutative**
- Correlation: Linear

Box and Gauss filter

- Both are **smoothing filters**. A box filter has **sharp edges** and will introduce **high frequency contributions** which can cause aliasing. In contrast a gauss filter slowly approaches zero, for which reason, if finely resolved a gauss filter will not lead to **aliasing**.

Determination of kernel size based on filter size

- Choose the kernel size such that the **kernel is sufficiently close to zero at its boundaries** to avoid aliasing. A width of 3σ is a good choice.

Application of a Median Filter

- Effective removal of salt and pepper noise.

Handling of Boundary Issues

- At the boundary a filter cannot be applied since depending on the size of the kernel a certain number of additional pixels are needed. To provide those one can add **zero padding**, **wrap the image around**, **copy the edge** or **reflect across the edge**.

1D Edge Detection

- **Smoothing** of the image before taking **first derivative**, otherwise we have large **noise induced peaks** in the derivative. The 1D edge can then be found as the **local maximum or minimum** of the derivative. Alternatively the **zero crossing of the second derivative** can give the position of the edge.

Explanation of Differential Property of the Convolution

- When edge detection is combined with smoothing the differential property of the convolution comes in handy, since it allows to change the order of operations (**Commutativity**). Instead of differentiating the image itself, this operation can be applied to the filter and thus needs to be performed **only once**, and also on a smaller array, since the filters are usually smaller than the filtered images.

Computation of the first derivative of an image in both directions

- Convolution of a **Prewitt** or a **Sobel** filter in the respective directions results in the gradients of the image, two image-sized arrays.

Explain use of LoG

- Having identified the use of the second derivative as highlighting an edge with its **passage through zero**, we can say that the second derivative is useful to identify edges. Doing this in both directions and linearly combining the two yields a single 2D map which highlights corners with clear zero passings, if combined with a **noise-removing filter**, which in this case is a gaussian. Using the **derivative property of the convolution** needed for filtering we arrive at the laplacian of gaussian filter.

Properties of smoothing and derivative filters

- **Smoothing** filters have **positive signs** and **sum up to 1** in order not to change the overall luminosity of the image. They remove high frequency content and are thus **low pass filters**.
- Derivative filters have **opposite signs** in order to highlight changes in the image. The **sum up to 0** such that for flat regions there is no response of the filter. These filters are **high pass filters** and remove low frequency content.

Canny Edge Detector

- Compute **gradient** of the image.
- **Thresholding** of the gradient images.
- **Non-maxima suppression**, identifying the maximum **along the gradient direction**.
- **Non-maxima suppression:** Evaluate gradient direction of the current pixel and check whether it is a maximum in the corresponding direction (discretize possible directions). Preserve the value if it is a maximum in its direction otherwise suppress.

Point Feature Detection

Template Matching

- Use the **correlation** to detect a template in the image. No flipping, otherwise template would not fit. If the template matches a location in the image there will be a maximum in the correlation.
- **Limitations: Not size-invariant**, **not rotation-invariant**, and template and object have to be very similar. **Background** of the template might mess up a match. Template matching has no context awareness, a car cannot match another if they are not very similar.
- Alternatively use the NCC.

Similarity Metrics (SSD, SAD, NCC

- **NCC** Normalized Cross Correlation: Consider patches/images as **vectors** in $\mathbb{R}^n$ and calculate the **cosine of the angle** between the two using the scalar product of the normalized vectors.
- **SAD** Sum of Absolute Differences. Direct comparison of corresponding pixel values.
- **SSD** Sum of Squared Differences, can be interpreted as a measure for the length of the connecting vector between the two images understood as vectors in $\mathbb{R}^n$.
- **Subtract the mean of the images to account for overall illumination changes!**

Good Features for Tracking

- Distinctiveness
- Rotation, Scale, Illumination, Distortion Invariance
- Repeatability

Corners and Blobs

- Corners:
  + High localization accuracy
  - Less distinctive
- Blobs

+ More distinctive

+ Better for **place recognition** (for example SIFT can handle large viewpoint, scale and illumination changes.

- Less localization accuracy.

Moravec Definition of Corner, Edge and Flat Region (HARRIS)

- Define a window size.
- Consider the **SSD** of a single **pixel shift** in one of four directions.
- The **minimum of the four SSDs** is the measure of interest, which defines how much of a corner is within the window.

Second moment matrix (HARRIS)

- Consider a patch and a shifted version of it. Calculate the **SSD between the two** and approximate it with a **first order approximation** using the image gradients.
- Write it as a **quadratic form** to receive the second moment matrix. (Pixel wise products)
- The 2D quadratic form can be analysed by its eigenvalues which indicate if the currently considered window is a flat region, a corner or an edge. These eigenvalues can be visualized using ellipses.

Properties of the M matrix for corners/edges/flat/90-degree-corner/non-axis-aligned 90-degree-corner regions (HARRIS)

- **Corner**: Eigenvalues are **both nonzero**.
- **Edge**: Only **one eigenvalue** is significantly **larger than zero**.
- For any corner diagonalize the second moment matrix ($R^{-1}ER$) to find the orientation of the directions of fastest and slowest change of SSD.
- The minimum of the Eigenvalues of the M matrix define the **Cornerness-Function**.

Harris VS Shi-Tomasi

- Shi-Tomasi uses the Cornerness-Function as defined by the minimum of the eigenvalues.

$$\boxed{R = \min(\lambda_1, \lambda_2)}$$

- Harris uses an approximation by the difference of the determinant of the matrix and the squared trace times k(0.04-0.15).

$$\boxed{R = \det(M) - k \cdot \text{trace}^2(M)}$$

Invariance to Illumation Changes and Scale changes (HARRIS)

- Invariant to rotation (reverted when finding the eigenvalues)
- Not invariant to large scale changes.
- As long as the illumination change does not diminish the contrast too much corners should still be detected.

Repeatability of Harris Detector after increasing scale by 2.

- Only 18%.

Point Feature Detector Part 2

Working principle of automatic scale detection.

- Rescaling patches individually is expensive.
- Find a function over the patch size that has a **maximum at a scale at which the feature of interest is represented best**, which in addition is **independent of the scale of the image** considered. After the scale has been found the patch size can be **normalized**. In addition the function should have **single and sharp peaks**.
- Convolution of image and a kernel is a possible solution. Kernel of choice: LoG, since we'd like to find corners.

Efficient implementation of automatic scale detection?

- Not calculation of LoGs but **DoGs**

Feature Descriptor

- A feature descriptor is information that aims to **describe the feature** in consideration as **uniquely** and as **concisely** as possible.
- Patch descriptors: For rotation and scale invariance: **Rescale and de-rotate** (normalisation by dominant direction of gradient) (**warping** necessary)
- Better solution: **HoGs** (Histograms of Oriented Gradients).
- For matching use comparison function and find lowest difference.

Keypoint Detection SIFT vs. Harris

- Harris: **Corners**! Eigenvalues of Second Moment matrix.
- SIFT: **Blobs**! Identified as maxima of the DoGs in **space** and **scale**.

Orientation Invariance for SIFT

- Assign each keypoint a **canonical orientation**. Strongly smoothed HoG peak to make stable for lighting and contrast changes. This does not originate in the use of HoGs for the subpatches!

Working Principle: SIFT

- **Detector $\neq$ Descriptor!**
- Make **image pyramid**, progressively smaller resolution for larger scales.
- Apply **3 versions of blurring** to each scale, increasing the variance of the applied Gauss filter by increments of $2^{1/s}$ where is is the number of intervals per scale space.
- Calcualte the **DoG pyramid**.
- Identify the blobs as **maxima** of the DoGs.
- Establish features descriptors for each blob:
  1. 4x4 HOGs (4x4 patches), 8 directions, Location (2D), Scale, Orientation
  2. Gauss-Filter patch
  3. Divide patch into subpatches
  4. Compute HOGs
  5. Concatenate HOGs
  6. Normalize the descriptor

Robustness of SIFT

- Scale change of 2: No problem, just shifts feature to the next octave of the pyramid.
- Viewpoint change of 50 degrees: Can handle yes, example mars rover.

Illustrate the 1st to 2nd closest ratio of SIFT detection: what's the intuitive reasoning behind it? Where does the 0.8 factor come from?

- Intuition: The closest match for a candidate blob with the database **should be significantly closer than the next best guess**, because if not there is no certainty about the matching - a small stochastic change could tip the balance. So if we found a mismatch there are potentially many other mismatches around. The threshold of 0.8 includes many matches while discarding a large proportion of the mismatches. Optimized over a large data set.

2 Ways for Tracking Features

- KLT
- Detection, Description and Matching.

Popular Detectors and Descriptors

- **Detectors**
  - **Harris:** Approximation to minimal eigenvalue of second moment matrix.
  - **Shi-Tomasi:** Minimal eigenvalue of the second moment matrix.
  - **FAST:** Intensity of pixels on **circle** around candidate pixel C. Is a corner if N **contiguous pixels are either brighter or darker** than the intensity of C + a threshold.
  - **SIFT:** Maxima of DoGs in scale and space.
  - **SURF:** Similar to SIFT, but faster due to approximations.
- **Descriptors**
  - **Patch:** Direct comparison via similarity measure.
  - **SIFT:** Oriented and scaled subpatch HoGs.
  - **SURF:** Shorter descriptors but similar to SIFT.
  - **BRIEF:** Random or machine-learning description pattern, evaluated as binary descriptor which allows fast comparison.
  - **ORB:** Oriented and Rotated BRIEF descriptors
  - **BRISK:** Predefined pattern, binary descriptor much faster than SURF.
  - **FREAK:** Pattern inspired by the human retina, fast, compact, robust.

SFM vs. 3D reconstruction

- Structure from motion assumes no knowledge of the camera intrinsics/extrinsics.

Disparity (simplified and general)

- In stereo vision objects that are not infinitely far away do **not appear in the same position on both screens**. That difference is called disparity. For a simplified case where the cameras are aligned it can be easily measures, else we have to perform stereo rectification first.
- Mathematical Expression

$$\boxed{u_l - u_r = \frac{bf}{Z_P}} \text{ Simplified}$$

- Depth Uncertainty: Decreases with an increasing base line and increases with increasing depth.
- How to improve on uncertainty?

  Increase the baseline.
- Mathematical description of the uncertainty: Check how a small change in the **measured disparity affects the estimated depth**, thus express ΔZ_p by assessing the **partial derivative of** $Z_P(D)$.
- **General case:** For the general case of unaligned and different cameras **stereo rectification** has to be performed before the disparity can be assessed. Alternatively **triangulation** can deliver an estimate of the position and thus the camera-depending depths.

Effects of large/small baseline

- Large: Minimum measurable distance increases and for close object the search problem gets harder.
- Small: Large depth error.

Closest measurable depth?

- Depends in the **screen width**. The „flattest rays“of the two cameras that can be recorded limit the minimum measurable distance, also depending on the **baseline**.

How to compute intersection of 2 lines?

- **Linear:** Write the **projection equations for both cameras** and then find the 3D point by solving the resulting **system of equations** using svd.
- **Nonlinear:** Find that point that will **minimize reprojection errors**. The solution of the corresponding minimization problem is the nonlinear approach. (Minimization of the sum of squared reprojection errors)

Geometric interpretation of linear and nonlinear approach? What error is minimized?

- in the nonlinear case the **reprojection error** is minimized. In the linear case the **algebraic erro**r is minimized.

Epipole, Epipolar Line, Epipolar Plane

- The **epipoles** are the projections of the line connecting the two optical centers.
- The **epipolar line** is the projection of all point on the ray corresponding to p in the other camera.
- The **epipolar plane** is the plane defined by the connecting line of the two optical centers and the epipolar line/a combination of the two rays to P.

Epipolar lines for

- Converging Cameras
- Forward motion: Epipole has same coordinates in all frames. Points move along lines radiating from e. **Focus of expansion.**
- Side-moving camera: Perfect case: Epipolar lines are parallel to pixel rows. (if cameras are rotated accordingly) which simplifies the search problem a lot! Epipoles at infinity!

Stereo Rectification

- Mathematical Derivation of the rectifying homographies
  1. Write **perspective camera equation** for both images.
  2. Rewrite them in non-homogeneous coordinates (Not with $[R|t]$) and **invert the homogeneous transform**, now calculating the transformation that yields world coordinates.
  3. Formulate a **new projection** for both cameras featuring the **same rotation and intrinsics** (for both new views) but **different translations**. Solve old and new view for world coordinates to establish a system of equations for the new intrinsics and extrinsics.
  4. Define the combined intrinsics as

$$\boxed{\bar{K} = (K_L + K_R)/2 \qquad \bar{R} = [\bar{r}_1, \bar{r}_2, \bar{r}_3]}$$

where

$$\bar{r}_1 = \frac{C_2 - C_1}{||C_2 - C_1||}$$
$$\bar{r}_2 = r_3 \times \bar{r}_1 \; r_3 \text{ is the 3rd column of the left rotation matrix } R_L$$
$$\bar{r}_3 = \bar{r}_1 \times \bar{r}_2$$

- The new rotation matrix aligns one axis with the connection vector of the two cameras and makes the second axis perpendicular to that line and one direction of the orientation of the left camera.

Computation of the disparity map?

- Establish pixel correspondences and **subtract their u-coordinates**. Or actually in practice when shifting patches for comparison the optimal shift delivers the disparity of the current pixel.

Establishment of stereo correspondences with subpixel accuracy

- When comparing possibly comparing patches we can find the match by searching the **minimum of the applied similarity measure**. To achieve subpixel accuracy that function can be **interpolated** before searching the minimum.

Rejection of outliers in stereo correspondences

- Rejection of ambiguous matches if more than two possible matches are detected (similar score to optimal). More than 2 since 2 close candidates can still be justified as the two **encompassing the minimum in the similarity metric**.
- Reject match when at the **boundary of the disparity range** since that might indicate a minimum outside the range.
- Epipolar geometry
- Soft constraints: Uniqueness, same surface → same order, smooth disparity changes along same surface.

Alternatives to stereo vision for gaining depth information

- Monocular vision with **deep learning** combined with shape recognition/use of contextual information.
- Visual-Inertial Fusion.

Multiple View Geometry 2

Minimum Number of correspondences for calibrated SFM

- Number of knowns: $4n$ **established correspondences**
- $5 + 3n$ unknowns: **Pose up to scale** and n **3D points**.
- Minimum number of points needed: **5 or more**. Less with more restrictive assumptions (Steering Principle).

Epipolar Constraint

- If two cameras view the same point the two recorded points must lie on the **epipolar plane** by definition. Thus the epipolar constraint can be established by saying that p_1 and p_2 must be **coplanar**!

Essential Matrix

- The essential matrix is defined as the **cross-matrix of T multiplied with R** and results directly from the **epipolar constraint**.

8-point algorithm

- Formulate the **epipolar constraint for 8 points** to receive a unique (up-to-scale) solution delivering the essential matrix. Degenerate if the 3D points are **coplanar**. It minimizes the projection of p_1 onto the normal direction n.

Number of Decompositions of the essential matrix

- 4 decompositions. And there is only one solution where the points are front of both cameras.

Relation between essential and fundamental matrix.

- The essential matrix formulates the epipolar constraint for already calibrated cameras. When a calibration needs to be done in addition the system of equations can be reformulated such that the matrix in question becomes the calibration-including **fundamental matrix**.

Importance of normalization of the point coordinates for the 8-point algorithm

- **Large range of pixel values** makes for different scaling of the different equations which leads to **poor numerical conditioning** of the resulting system of equations.
- Transformation of the image coordinates such that they are **between 1 and -1**.
- Or transformation with **shift by the centroid** of the distribution and **scaling to a standard deviation of** $\sqrt{2}$.

Normalized 8-point algorithm

- Normalize point correspondences
- Compute normalized $\hat{F}$
- Transform back: $F = B_2^T \hat{F} B_1$
- **Possible if both camera image centers are known and both have the same focal length.**

Quality metrics for the essential matrix estimation

- Evaluate the **epipolar constraint**.
- Directional error evaluating the **squared cosines of the angles between** p_2 **and the normal** n. Find cosine from dot product $p_2 \cdot Ep_1$.
- Squared epipolar line to point distance
- **Squared reprojection errors**, expensive computation but accurate.

Why do we need RANSAC?

- Because there are outliers because of image noise, blur, moving objects and occlusions.

Theoretical maximum number of combinations to explore

- Cardinality of the set: 2^N

Number of iterations for a given success probability?

- $\boxed{k = \frac{\log(1-p)}{\log(1-w^s)}}$
- Calculate probability that RANSAC never selects s inliers in k tries.
- This is not the probability for sucess. If the inliers are too noisy the result might still be unusable.

Trend of RANSAC iterations vs. fraction of outliers, vs. number of points?

- Fraction of outliers: exponential increase of needed number of tries.
- Number of points: exponential increase of needed number of tries.

How to apply RANSAC to

- 8-point algorithm: Randomly select 8 points, calculate the pose change, apply to points and count inliers.
- DLT: Randomly select 6 points, calculate projection matrix, apply to all points and calculate projection error.
- P3P: Select 4 points, set up and solve carnots theorem, disambiguate, calculate projection error.

How to reduce the number of RANSAC iterations for the SFM problem?

- Yes we can reduce the number of points below 5 by the inclusion of motion constraints. **Planar motion** or **Ackermans steering principle**.

Multiple View Geometry 3

Bundle Adjustment

- Mathematical Expression

$$\boxed{(P^i, C_k) = \arg\min_{p^i, C_k} \sum_k \sum_i ||p_k^i - \pi_k(P^i, C_k)||^2}$$

- Bundle adjustment optimizes the estimated camera poses and the triangulated landmarks, **minimizing the sum of squared reprojection errors**.
- Pose-Graph optimization **only optimizes the poses**, BA optimizes the structure as well. BA is more precise but also more costly.

Hierarchical and Sequential SFM for monocular VO

- **Hierachical:** Extract and match features between nearby frames, **establish groups** of three and compute SFM for those by doing it first for the first two images and then **merging the third view** using 3-point RANSAC. **Merge clusters** pairwise and **refine** both structure and motion.

- **Sequential:** Initialize structure and motion from 2 views (**bootstrapping**). **Sequentially add new** views by determining pose, then extending the structure, then **refine** pose and structure.

What are Keyframes?

- Why needed? Since the triangulation through two frames that are to close will be inaccurate it is advisable to select two frames with a large enough baseline.
- How to select? Based no the ratio between baseline and average depth.

Loop closure detection

- Realizing when a place is seen for the second time.
- Detecting loop closures allows introducing **additional constraints removing drifts** in the map. It allows helps to **avoid map duplication**.

Most popular open source VO and VSLAM?

VSLAM) PTAM
  – Splits tracking and mapping
  – Allows more efficient computation of usually too expensive optimizations in real time.

VSLAM) ORB-SLAM
  – Three Threads: Tracking, Local Mapping and Loop Closing, Real Time

VSLAM) LSD-SLAM
  – Direct Method, Semi-Dense, Real Time

(VO) DSO
  – Direct Method, Sparse, Real Time

(VO) SVO
  – Semi-Direct, Real Time, Two Threads, Sparse

Features-based vs direct methods

- Feature based methods rely on **distinct image points** which are tracked throughout frames. They minimize the reprojection error.
  + Large frame-to-frame motions
  + Accuracy: Efficient optimization of structure an motion possible
  - Slow due to costly feature detection
  - Robustness needed
- Direct methods do not bother identifying features but track the motion by matching the whole frame and minimizing the **photometric error**.
  + All information in the image used.
  + Increasing camera frame-rate reduces computational cost per frame.
  + Subpixel accuracy.
  - Limited frame-to-frame motion.
  - Joint optimization of dense structure and motion to expensive.

Dense 3D Reconstruction

Multi-View stereo working principle

- Start with a series of calibrated images with known extrinsics.
- First **estimate the depth of every pixel**. The idea is to minimize the photometric error in all images as a function of depth in the first image. Do this for all image combinations and **aggregate the error** (SSD between corresponding patches, pros/cons windowsize). Then find the **miniumum**.
- Consider limiting to **small baseline** to avoid occlusions.
- Second apply **global methods** on the recovered 3D structure to **make it smoother**.

Aggregated Photometric Error for corners, flat regions, edges?

- Corners: Clear minimum.
- Flat: No recognizable minimum.
- Edges: Extended minimum.

Disparity Space Image

- Calculate the **photometric error for every pixel and all depths** for the reference image I_R and sum up over all other images.

How to extract depth from DSI

- Find the **minimal photometric error for every pixel** to find the **most probable depth** for the corresponding point in 3D.

How to enforce smoothness (regularization) and how to incorporate depth discontinuities

- To enforce smoothness the cost function for the minimization is extended by a **term penalizing fast depth changes**.
- This however does not incorporate **depth discontinuities** that exist in the 3D world. The solution is a **weighting of the regularization** with the **gradient of the actual image**. The stronger the gradient in the image the weaker the influence of the smoothing term.

What happens if we increase lambda? What if lambda = 0, what if lambda is too big?

- The smaller lambda the weaker the effect of the regularization. If lambda = 0, no regularization. If lambda is to big the depth information is lost as the image is flattened.

What is the optimal baseline for multi-view stereo?

- The larger the baseline the more difficult the search problem becomes, as corresponding pixels move further apart. Thus the depth map is obtained from small baselines until the baseline is larger than about 10% of the average scene depth, then a new reference frame is chosen and the individual depth maps are merged later.

What are advantages of GPU's?

- Much faster at **parallel computation**. Vector-Matrix Calculations, Raytracing, any pixel-wise operations.

Tracking

Illustrate Tracking with Block matching

- Define a corner, define a patch around it and **try to find that patch again** in the next image. Works well even for large motions, but can become computationally demanding.

Differential Methods

- Assumptions
  1. **Photo consistency**: The patch around the point to be tracked should remain similar to its original.
  2. **Temporal persistence**: The motion between two images must be small.
  3. **Spatial coherency**: Neighbouring pixels belonging to the same surface have similar motion.
- Derive mathematical expression
  1. Want to f**ind the motion vector that minimizes the SSD**: $\sum(I_0(x,y) - I_1(x+u, y+v))^2$ which is **approximated using the image gradient** $E = \sum(\Delta I - I_x u - I_y v)^2$
  2. To minimize the error E we set the **derivative equal to zero**.
  $$\frac{\partial E}{\partial u} = 0 \Rightarrow -2\sum I_x(\Delta I - I_x u - I_y v) = 0$$
  $$\frac{\partial E}{\partial v} = 0 \Rightarrow -2\sum I_y(\Delta I - I_x u - I_y v) = 0$$
  3. Write in matrix form to recover the M matrix. We can now invert it to find u, v from the local image gradient and the illumination change.
- Meaning of the M matrix: Same interpretation as for harris.

Invertibility of the M matrix

- Invertible for non-flat, non-edge regions.

Aperture Problem

- Definition: When only part of a structure is visible its **motion can not be determined uniquely** if it does not contain a corner. There exist infinitely many solutions exist.
- Solution: Increase the aperture size.

Optical Flow

- Describes **how pixels move** when the **viewpoint changes** or the **viewed objects move**. Tracking of the motion of every pixel between two frames.

Block-Based vs. Differential Methods for tracking

- Block based
  + Works for large motions
  - Becomes computationally demanding for large motions
- Differential methods
  + Tracks every pixel

+ No search problem to be solved, thus much more efficient.
- Does not work for large motions unless multi scale methods are used.

Working principle of KLT

- Idea: Template Tracking with general twist instead of translation.
- Find the minimum by **incrementing the initial guess** $\underline{p}$ with $\Delta\underline{p}$ locally. Thus a first order approximation is made.
- Differentiate and set equal to zero.
- Iterate over $\underline{p} \leftarrow \underline{p} + \Delta\underline{p}$.

Hessian Matrix

- The Hessian matrix connects the **steepest descent parameter updates** to the **actual parameter updates** needed to achieve a step in the steepest direction. It coincides with the point tracking Hessian for a simple translation.

Lukas-Kanade failure cases

- **Large motion**, or rather if the initial guess is bad, the linearization no longer holds, thus we make a coarse-to-fine-approach.
- **Template changes over time**, update it with the last image.

How to get the initial guess?

- **Pyramidal implementation**. Down-sampling makes the apparent motion smaller and the search problem easier. Propagation of the initial guess from coarse to fine.

Alternative tracking procedures using point features

- Detecting features.
- Matching features.
- Geometric verification (RANSAC).

Recognition

Inverted File Index

- Locations to content is an index. **Content to locations** is an inverted index.

Visual Word

- Collect a large enough dataset (book).
- Extract features and descriptors (letters) and map them into the same **descriptor space**.
- **Cluster the descriptor space**, make words from letters.
- The **centroid of each cluster** is a word.

K-means clustering

- **Divide a space into K clusters** with assigned means / centroids.
- How: **Minimize the Euclidean distance** between points and the nearest cluster centers.
  1. **Reassign each datapoint** to the nearest center.
  2. **Recompute each cluster center** as the mean of all points assigned to it.

Purpose of Hierarchical clustering

- **Reduce the number of word comparison** by **clustering similar words** into a common subject / same initial letter.

Image retrieval using Bag of Words

- Establish **inverted file index** and cluster / **cluster** hierarchically.
- Find **features in query** image.
- Search the **corresponding words** in the inverted index and **establish the voting array** for all database images, counting the number of matching words per DB image.
- The matching image is the one with the **highest voting**.

Open challenges on place recognition, what proposed solutions

- Images with **same words but shuffled** will return a 100% match even though the images are different. Use **geometric verification**, for example RANSAC and 8-point algorithm. Use reprojection error and number of inliers as quality metric.

Visual Inertial Fusion

Usefulness of IMU for VO

- **Scale ambiguity** for monocular vision, recover scale with IMU measurements.
- **Increase robustness** in general. **Low texture, High dynamic range, High speed motion.**

Working principle of a MEMS IMU

- Accelerometer: Mass attached to two springs, acceleration-induced movement leads to a change of the electric signal generated by a capacity divider.
- Gyroscope: Measures the Coriolis force acting on the structure. Similar to **haltere of flies.**

Why not just an IMU?

- Because IMUs have the tendency to drift.
- Integration of the angular velocities thus error increases with t.
- Double integration of the acceleration thus error increases with t^2.

Drift of Industrial IMU

- Thousands of Kilometers per hour for cheap IMU. Much better for extremely expensive ones, but not perfect.

What causes bias in IMU

- Temperature changes, mechanical pressure, new startup.

Definition IMU measurement model

- The rotational velocity results from a biased and noisy measurement of the gyroscope.
- The acceleration in world coordinates results from the de-rotated difference of the measured acceleration and the gravitational acceleration, combined with bias and noise.

How to model the bias?

- The derivative of the bias is white gaussian noise (random walk).
- ?

How to integrate acceleration?

- Simply integrate, do not forget initial conditions and gravity.

Definition of loosely coupled and tightly coupled visual inertial fusion?

- The loosely coupled approach treats IMU and VO as separate black boxes and fuses the resulting position, orientation (and velocity for IMU) estimations.
- In contrary the closely coupled approach allows for correlations between the internal states of the different measurement devices. Thus it is more accurate but requires more implementation effort.

How can we use non-linear optimization-based approaches to solve for visual inertial fusion?

- Use closed form solution to initialize filters and smoothers.
- For sensor calibration by energy minimization techniques.
- Fixed-lag of full smoothing, optimization of states by minimizing the IMU residuals and the reprojection residuals.

Event based Vision

Working principle: DVS

- Inspired by the human vision.
- Detects intensity changes in pixels asynchronously.
- Detection by level-crossing-sampling.
- Reacts to logarithmic brightness changes.

$$\boxed{C = |\Delta \log(I)| = |\log(I(t) + \Delta t) - \log(I(t))|}$$

Pros and Cons vs Standard Camera

+ Low latency ($\sim 1\,\mu s$)
+ HDR (140 dB instead of 60 dB)
+ High update rate (1 MHz)

+ Low power (10 mW instead of 1 W)

- Paradigm shift, new algorithms required

Are standard calibration techniques applicable?

- No, a still image of a checkerboard does not produce any events. Either the camera or the calibration image have to move. Blinking works as well.

How to compute optical flow with a DVS?

- Visualization using a single vertical black line in horizontal movement.
- Will trigger a series of events aligned vertically, travelling horizontally.
- The speed of the edge can be found as the the space difference of two events divided by their trigger-time difference.

Intuitive explanation of intensity reconstruction

- The basic idea is to integrate the intensity changes a pixel records to recover the intensity signal.
- This knowledge combined with the camera motion allows the recovery of the absolute brightness.
- Practice the recovery of the camera motion allows building a gradient map, which is then integrated to yield a brightness map.

Definition: Generative Model of a DVS

- Assume $I(x, y, t) = \log(I(x, y, t))$
- Pixel $p(x, y)$ with gradient $\nabla I(x, y)$ undergoing a motion $\underline{u} = (u, v)$ in pixels, induced by a moving 3D point $\underline{P}$
- An event is generated if $-\nabla I \cdot \underline{u} = C$

Definition: Davis Sensor

- Combines conventional image sensor with an event sensor in the same pixel array.
- Output images at certain framerate and events asynchronously.

Equation: Event generation model and Proof

- Assume Brightness Constancy: Pixel value remains the same during motion.
- Make a first order approximation in space.
- Calculate the pixel difference over time.