Autonomous Mobile Robot Design
Topic: Feature Extraction and Matching
Dr. Kostas Alexis (CSE)
Motivation

- **Goal 1**: we need to detect the same point independently within both images.
- **Goal 2**: For each point correctly recognize the corresponding one.
Motivation

- **Goal 1**: we need to detect the same point independently within both images.
- **Goal 2**: For each point correctly recognize the corresponding one.
Keypoint Detection and Feature Description

- **Keypoint** is an *image location* at which a descriptor is computed.
- The feature **descriptor** summarizes the *local structure around the keypoint*.
Popular Features

- **SIFT**: Scale Invariant Feature Transform
- **SURF**: Speeded-Up Robust Features
- **HOG**: Histogram of Oriented Gradients
- **GLOH**: Gradient Location and Orientation Histogram
- **BRIEF**: Binary Robust Independent Elementary Features
- **BRISK**: Binary Robust Invariant Scalable Keypoints
- ...
Keypoints

- **Task**: “Look for locally distinct points”

- **Procedure**
  - *Over different image pyramid levels*
  - **Step 1**: Gaussian smoothing
  - **Step 2**: Difference-of-Gaussians - find extrema (over smoothing scales)
  - **Step 3**: maxima suppression at edges
Visualization

differently
blurred images

differently
sized images
Difference of Gaussians

- Subtract differently blurred images from each other
- Increases visibility of corners, edges, and other detail present in the image
Scale-Space Representation

\[ i = 0, 1, 4, 16, 64, 265 \]
Difference of Gaussians

- Subtract differently blurred images from each other
- Only keeps the frequencies that lie between the blur level of both images
- Filters out high-frequencies (noise) but maintains local differences
- Acts as a band-pass filter
Difference of Gaussians

- Keypoints are extrema in the DoG over different (smoothing) scales
Visualization

Scale (next octave)

Scale (first octave)

Gaussian

Difference of Gaussian (DOG)

Scale
Extrema Suppression

- The DoG finds **blob-like** and **corner-like** image structures but also leads to strong responses along edges.
- SIFT uses a criterion based on the **ratio between the eigenvalues of the Hessian**.
Hessian Matrix

- In mathematics, the Hessian matrix or Hessian is a square matrix of second-order partial derivatives of a scalar-valued function, or scalar field.
- It describes the local curvature of a function of many variables.

\[
H = \begin{bmatrix}
\frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\
\frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\
\vdots & \vdots & \ddots & \vdots \\
\frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2}
\end{bmatrix}.
\]
Hessian Matrix for Vector-valued functions

- If \( f \) is a vector field: \( f : \mathbb{R}^n \rightarrow \mathbb{R}^m \)

- i.e.: \( f(x) = (f_1(x), f_2(x), \ldots, f_m(x)) \)

- Then the collection of second partial derivatives is not a \( n \times n \) matrix, but rather a third order tensor. This can be thought of as an array of \( m \) Hessian matrices, one for each component of \( f \):

\[
H(f) = (H(f_1), H(f_2), \ldots, H(f_m))
\]
What about the Descriptor?

keypoint

descriptor at the keypoint

\[ f = \begin{bmatrix} 0.02 \\ 0.04 \\ 0.1 \\ 0.03 \\ 0 \\ \ldots \end{bmatrix} \]
SIFT Matching
Extrema Suppression

- Image content is transformed into features that are **invariant to**
  - image translation,
  - rotation, and
  - scale
- They are **partially invariant to**
  - illumination changes and
  - affine or 3D projection
- Suitable for mobile robots in order to detect visual landmarks
  - from different angles and distances
  - with a different illumination.
SIFT Features

- A SIFT feature is given by a vector computed at a local extreme point in the scale space

\[ \langle p, s, r, f \rangle \]

- Location in the image: viewpoint dependent
- Scale
- Orientation

128-dim. descriptor, generated from local image gradients

- Mainly independent
SIFT Features

- Compute image gradients in local 16x16 area at the selected scale
- Create an array of orientation histograms
- 8 orientations x 4x4 histogram array = 128 dimensions (yields best results)

Example using an 8x8 area
Keypoint and Descriptor done!

keypoint

descriptor at the keypoint

\[ f = \begin{bmatrix} 0.02 \\ 0.04 \\ 0.1 \\ 0.03 \\ 0 \\ \ldots \end{bmatrix} \]
How to match them?
Based on Descriptor Difference?
Based on Descriptor Difference?
Solving the Correspondence Problem

- Choosing correspondences only based on the descriptor difference will lead to (some) wrong matches!
Solving the Correspondence Problem

- Solving the Correspondence Problem is Essential!
- Identifying outliers is key to robustly solving a large set of problems

**RANSAC**

RANdom SAmple Consensus
RANdom SAmpLe Consensus

- Proposed by Fischler & Bolles in 1981
- Approach to deal with high fractions of outliers in the data
- Try-and-error approach
- **Key idea:** Find the best partition of points in inlier set and outlier and estimate the model from the inlier set
- **Standard approach** for dealing with outliers
RANSAC Algorithm

- **Step 1:** Sample the number of data points required to fit the model
- **Step 2:** Compute model parameters using the sampled data points
- **Step 3:** Score by the fraction of inliers within a preset threshold of the model

**Repeat** 1-3 until the best model is found with high confidence
RANSAC: Line Fitting Example

- **Step 1: Sample** the number of data points required to fit the model
- **Step 2: Compute** model parameters using the sampled data points
- **Step 3: Score** by the fraction of inliers within a preset threshold of the model
- **Repeat** 1-3 until the best model is found with high confidence
RANSAC: Line Fitting Example

- **Step 1: Sample** the number of data points required to fit the model (here 2 points)
- **Step 2: Compute** model parameters using the sampled data points
- **Step 3: Score** by the fraction of inliers within a preset threshold of the model
- **Repeat** 1-3 until the best model is found with high confidence
RANSAC: Line Fitting Example

- **Step 1: Sample** the number of data points required to fit the model (here 2 points)
- **Step 2: Compute** model parameters using the sampled data points
- **Step 3: Score** by the fraction of inliers within a preset threshold of the model
- **Repeat** 1-3 until the best model is found with high confidence
RANSAC: Line Fitting Example

- **Step 1: Sample** the number of data points required to fit the model (here 2 points)
- **Step 2: Compute** model parameters using the sampled data points
- **Step 3: Score** by the fraction of inliers within a preset threshold of the model
- **Repeat** 1-3 until the best model is found with high confidence

#inliers: 4
RANSAC: Line Fitting Example

- **Step 1: Sample** the number of data points required to fit the model (here 2 points)
- **Step 2: Compute** model parameters using the sampled data points
- **Step 3: Score** by the fraction of inliers within a preset threshold of the model

- **Repeat** 1-3 until the best model is found with high confidence

#inliers: 12
RANSAC Example: Translation

select one match
RANSAC Example: Translation

count inliers (0)
RANSAC Example: Translation

select another match
RANSAC Example: Translation

count inliers (4)
RANSAC Example: Translation

Repeat: select one match, count inliers
RANSAC Example: Translation

select solution with most inliers
Feature-based Alignment
Feature-based Alignment

- Extract Features
Feature-based Alignment

- Extract Features
- Compute putative matches
Feature-based Alignment

- Extract Features
- Compute putative matches
- **Loop:**
  - Hypothetize transformation $T$
Feature-based Alignment

- Extract Features
- Compute putative matches
- Loop:
  - Hypothetize transformation $T$
  - Verify transformation (search for other matches consistent with $T$)
Feature-based Alignment

- Extract Features
- Compute putative matches
- Loop:
  - Hypothetize transformation $T$
  - Verify transformation (search for other matches consistent with $T$)
- Extract best solution and apply
RANSAC: Pros and Cons

- **Pros**
  - Robust to outliers
  - Applicable for number of objective function parameters of up to ~10

- **Cons**
  - Computational time grows quickly with fraction of outliers and number of parameters needed to fit the model
  - Not good for getting multiple fits
Common RANSAC Applications

- Finding point correspondences
- Estimating fundamental matrix (relating two views)
- Visual odometry
- Computing a homography (e.g., image stitching)
- Laser scan matching
- ...

Summary

- Interest points can be found automatically
- Several descriptors (SIFT, SURF, …)
- Just comparing feature descriptors is not enough
- Outlier rejection is key to robust operation
- RANSAC – the standard tool for model fitting with outliers
Code Examples and Tasks

- Feature detection and matching & application to SLAM
  - https://github.com/unr-arl/autonomous_mobile_robot_design_course/tree/master/matlab/image-processing
  - https://github.com/unr-arl/autonomous_mobile_robot_design_course/tree/master/matlab/localization-mapping
  - SIFT Example in class
How does this apply to my project?

- Enable camera-based localization and mapping
- Detect objects and areas of interest
- Track areas and objects of interest
Find out more

- https://en.wikipedia.org/wiki/Feature_(computer_vision)
- http://docs.opencv.org/trunk/index.html
- http://crcv.ucf.edu/REU/2013/p2_SIFT.pdf
Thank you!

Please ask your question!