Image Stitching

Morten R. Hannemose, mohan@dtu.dk

April 12, 2024

02504 Computer vision course lectures,
DTU Compute, Kgs. Lyngby 2800, Denmark

This lecture is being livestreamed and recorded

(hopefully)

Two feedback persons

Learning objectives

After this lecture you should be able to:

- explain and implement RANSAC to fit homographies
- understand explain the challenges involved in stitching panoramas

Presentation topics

Image stitching

Finding inliers

Transforming images

Multiple images

Nodal point

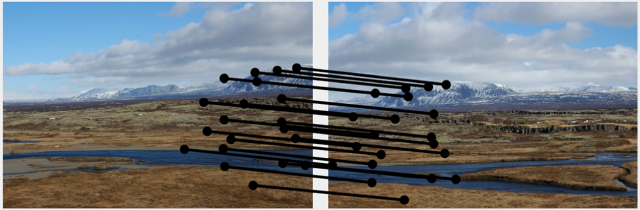
Setting the scene

Today you will be using:

- Homographies (hest)
- (SIFT) Features
- RANSAC

Image stitching

Image stitching – Panorama



About homographies

- What does a homography describe?

About homographies

- What does a homography describe?
  - The mapping of points between to images viewing the same plane.
- BUT! the mountains on the previous slide are not a plane!?
  - Why can we still use a homography?

Homographies for panoramas

- When the camera doesn't move but only rotates there are no perspective deformations
- It is equivalent to looking at a painting of the world
- Therefore we can assume the world is flat and use a homography!

Homographies for panoramas

- When the camera doesn't move but only rotates there are no perspective deformations
- It is equivalent to looking at a painting of the world
- Therefore we can assume the world is flat and use a homography!
- What happens if the camera moves?

Finding inliers

Measuring the error of a match

Let $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ be matching points without errors, such that

$$\begin{aligned}
\boldsymbol{p}_1 &= \boldsymbol{H}\boldsymbol{p}_2 \\
\boldsymbol{p}_1 &= \Pi^{-1}(\boldsymbol{q}_1), \quad \boldsymbol{p}_2 = \Pi^{-1}(\boldsymbol{q}_2) \\
\boldsymbol{q}_1 &= \Pi\left(\boldsymbol{H}\Pi^{-1}(\boldsymbol{q}_2)\right)
\end{aligned}$$

Measuring the error of a match

We have some 2D error in our detection of the points

$$\tilde{\boldsymbol{q}}_1 = \boldsymbol{q}_1 + \boldsymbol{\epsilon}_1, \quad \tilde{\boldsymbol{q}}_2 = \boldsymbol{q}_2 + \boldsymbol{\epsilon}_2$$
$$\tilde{\boldsymbol{p}}_1 = \Pi^{-1}(\tilde{\boldsymbol{q}}_1), \quad \tilde{\boldsymbol{p}}_2 = \Pi^{-1}(\tilde{\boldsymbol{q}}_2)$$

Measuring the error of a match

We have some 2D error in our detection of the points

$$\tilde{\boldsymbol{q}}_1 = \boldsymbol{q}_1 + \boldsymbol{\epsilon}_1, \quad \tilde{\boldsymbol{q}}_2 = \boldsymbol{q}_2 + \boldsymbol{\epsilon}_2$$
$$\tilde{\boldsymbol{p}}_1 = \Pi^{-1}(\tilde{\boldsymbol{q}}_1), \quad \tilde{\boldsymbol{p}}_2 = \Pi^{-1}(\tilde{\boldsymbol{q}}_2)$$

The error is the distance to the observed point in both images.

$$\left\|\tilde{\boldsymbol{q}}_1 - \Pi\left(\boldsymbol{H}\Pi^{-1}(\boldsymbol{q}_2)\right)\right\|_2^2 + \|\tilde{\boldsymbol{q}}_2 - \boldsymbol{q}_2\|_2^2$$

Do we know $\boldsymbol{q}_2$?

Measuring the error of a match

We have some 2D error in our detection of the points

$$\tilde{\boldsymbol{q}}_1 = \boldsymbol{q}_1 + \boldsymbol{\epsilon}_1, \quad \tilde{\boldsymbol{q}}_2 = \boldsymbol{q}_2 + \boldsymbol{\epsilon}_2$$
$$\tilde{\boldsymbol{p}}_1 = \Pi^{-1}(\tilde{\boldsymbol{q}}_1), \quad \tilde{\boldsymbol{p}}_2 = \Pi^{-1}(\tilde{\boldsymbol{q}}_2)$$

The error is the distance to the observed point in both images.

$$\left\|\tilde{\boldsymbol{q}}_1 - \Pi\left(\boldsymbol{H}\Pi^{-1}(\boldsymbol{q}_2)\right)\right\|_2^2 + \|\tilde{\boldsymbol{q}}_2 - \boldsymbol{q}_2\|_2^2$$

Do we know $\boldsymbol{q}_2$?

No!

Measuring the error of a match

We can use optimization to estimate the error free point

$$\text{dist}^2_{\mathsf{true}} = \min_{\boldsymbol{q}_2} \left\| \tilde{\boldsymbol{q}}_1 - \Pi\left(\boldsymbol{H}\Pi^{-1}(\boldsymbol{q}_2)\right) \right\|_2^2 + \left\| \tilde{\boldsymbol{q}}_2 - \boldsymbol{q}_2 \right\|_2^2$$

Requires solving a least squares problem for each distance computation.

Impractical!

Practical approximation

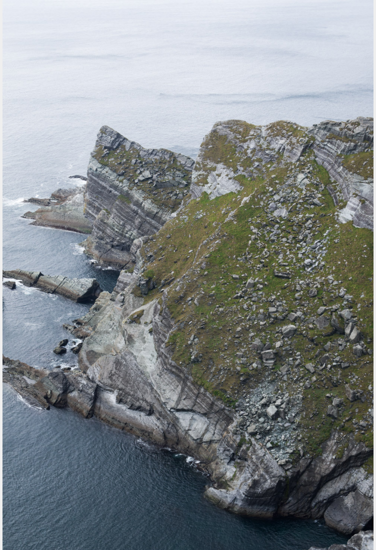
Map the observed points back and forth, and compute the distance

$$\mathsf{dist}^2_{\mathsf{approx}} = \|\tilde{\boldsymbol{q}}_1 - \Pi\left(\boldsymbol{H}\tilde{\boldsymbol{p}}_2\right)\|_2^2 + \left\|\tilde{\boldsymbol{q}}_2 - \Pi\left(\boldsymbol{H}^{-1}\tilde{\boldsymbol{p}}_1\right)\right\|_2^2$$

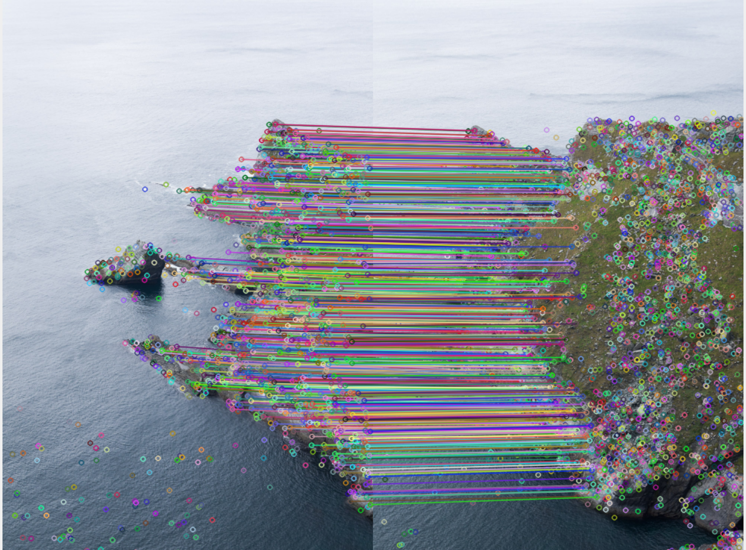
Use this distance to measure if a pair of points are an inliers with respect to a homography.

Transforming images

Example – Images



Example – Matches



How to use the homography?

- We have estimated the homography
- How do we use it?
- Same approach as image undistortion.

Transforming images

- We can use the homography to warp an image to our current field of view.
- Generate all x, y coordinates for all pixels in the reference image
- Map these to the other image using the homography
- Use bilinear interpolation to compute the value at the transformed pixel locations.
  - Code is provided in the exercise.

Warping

- You have to decide in which area to evaluate the homography
- Simple, evaluate all homographies in the same area
    - Wasteful computation
    - Simple to implement
- Warp only the valid part of each image and move the images around
    - Necessary for larger panoramas

Example – Warped to same coordinate system

Note that x is negative.

Compositing images

- Overlap
- Average
- Median
- Graph cut

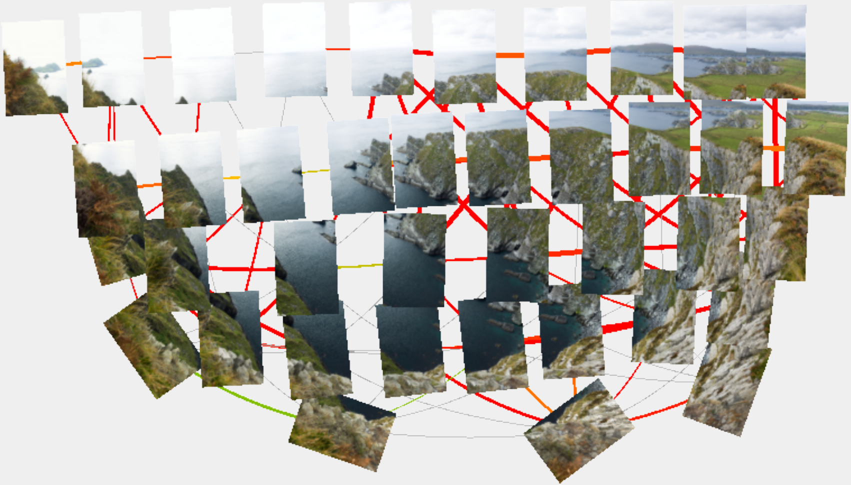
Example – Composited

Compositing – Average

Compositing – Graph Cut

Multiple images

Multiple images



Multiple images – Final result

How do we handle more than two images?

Example with three images:

Find the homographies between each pair $\boldsymbol{H}_{1\to 2}$ and $\boldsymbol{H}_{2\to 3}$.

Products of homographies are new homographies!

$$\boldsymbol{H}_{1\to 3} = \boldsymbol{H}_{1\to 2}\boldsymbol{H}_{2\to 3}$$

This principle extends to as many images as desired.

Multiple images

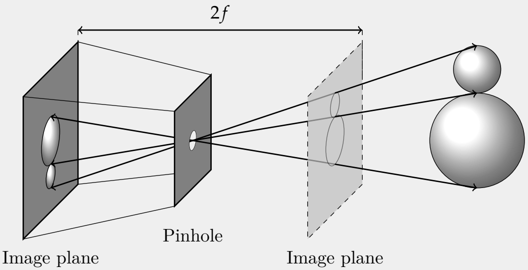
- Choose composting surface
  - I.e. which image is our baseline
  - Easy option is to choose one image to be the center
- Joint optimization

Nodal point

Capturing your own images

Which point should you actually rotate around?

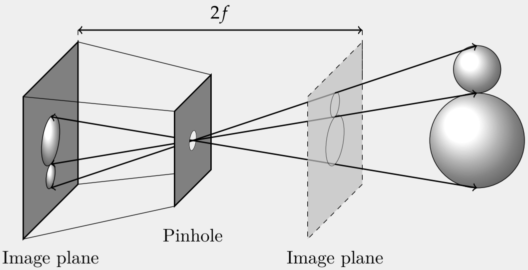
- Image sensor?
- Pinhole?



Capturing your own images

Which point should you actually rotate around?

- Image sensor?
- Pinhole?



Real world camera

Which point should you actually rotate around?

Real world camera

Which point should you actually rotate around?

It depends on the camera, usually the middle of the lens.

What happens if we rotate around the wrong point?

Real world demo[ish]

And more!

- Known focal length reduces degrees of freedom
- Other projection models
- Exposure/color correction

Learning objectives

After this lecture you should be able to:

- explain and implement RANSAC to fit homographies
- understand explain the challenges involved in stitching panoramas

Exercise time!