Lecture 15
Robust Estimation : RANSAC
RECALL: Parameter Estimation:

Let’s say we have found point matches between two images, and we think they are related by some parametric transformation (e.g. translation; scaled Euclidean; affine). How do we estimate the parameters of that transformation?

General Strategy

• Least-Squares estimation from point correspondences

But there are problems with that approach....
Problem : Outliers

Loosely speaking, outliers are points that don’t “fit” the model.
Bad Data $\Rightarrow$ Outliers

Loosely speaking, outliers are points that don’t “fit” the model. Points that do fit are called “inliers”
Problem with Outliers

Least squares estimation is sensitive to outliers, so that a few outliers can greatly skew the result.

Solution: Estimation methods that are robust to outliers.
Outliers aren’t the Only Problem

Multiple structures can also skew the results. (the fit procedure implicitly assumes there is only one instance of the model in the data).
Robust Estimation

• View estimation as a two-stage process:
  – Classify data points as outliers or inliers
  – Fit model to inliers while ignoring outliers

• Example technique: RANSAC
  (RANdom SAmple Consensus)

Ransac Procedure
Ransac Procedure
Ransac Procedure

Count = 4
Ransac Procedure
Ransac Procedure

Count = 6
Ransac Procedure

Count = 19
Ransac Procedure
Ransac Procedure

Count = 13
Ransac Procedure

Count = 4
Count = 6
Count = 19
Count = 13
Algorithm 15.4: RANSAC: fitting lines using random sample consensus

Determine:
  - \( S \) — the smallest number of points required
  - \( N \) — the number of iterations required
  - \( d \) — the threshold used to identify a point that fits well
  - \( T \) — the number of nearby points required to assert a model fits well

Until \( N \) iterations have occurred
  - Draw a sample of \( S \) points from the data uniformly and at random
  - Fit to that set of \( S \) points
  - For each data point outside the sample
    - Test the distance from the point to the line against \( d \) if the distance from the point to the line is less than \( d \) the point is close
  - If there are \( T \) or more points close to the line then there is a good fit. Refit the line using all these points.

Use the best fit from this collection, using the fitting error as a criterion

(Forsyth & Ponce)
How Many Samples to Choose?

\[ 1 - (1 - (1 - e)^s)^N = p \]

- \( e \) = probability that a point is an outlier
- \( s \) = number of points in a sample
- \( N \) = number of samples (we want to compute this)
- \( p \) = desired probability that we get a good sample

Solve the following for \( N \):

Where in the world did that come from? ....
How Many Samples to Choose?

\[ 1 - (1 - (1 - e)^s)^N = p \]

\[ \text{Probability that choosing one point yields an inlier} \]

\( e \) = probability that a point is an outlier
\( s \) = number of points in a sample
\( N \) = number of samples (we want to compute this)
\( p \) = desired probability that we get a good sample
How Many Samples to Choose?

\[ 1 - (1 - (1 - e)^s)^N = p \]

- \( e \) = probability that a point is an outlier
- \( s \) = number of points in a sample
- \( N \) = number of samples (we want to compute this)
- \( p \) = desired probability that we get a good sample

Probability of choosing \( s \) inliers in a row (sample only contains inliers)
How Many Samples to Choose?

\[
e = \text{probability that a point is an outlier}
\]

\[
s = \text{number of points in a sample}
\]

\[
N = \text{number of samples (we want to compute this)}
\]

\[
p = \text{desired probability that we get a good sample}
\]

\[
1 - (1 - (1 - e)^s)^N = p
\]

Probability that one or more points in the sample were outliers (sample is contaminated).
How Many Samples to Choose?

\[ 1 - (1 - (1 - e)^s)^N = p \]

- \( e \) = probability that a point is an outlier
- \( s \) = number of points in a sample
- \( N \) = number of samples (we want to compute this)
- \( p \) = desired probability that we get a good sample

Probability that \( N \) samples were contaminated.
How Many Samples to Choose?

\[
e = \text{probability that a point is an outlier}
\]
\[
s = \text{number of points in a sample}
\]
\[
N = \text{number of samples (we want to compute this)}
\]
\[
p = \text{desired probability that we get a good sample}
\]

\[
1 - (1 - (1 - e)^s)^N = p
\]

Probability that at least one sample was not contaminated
(at least one sample of s points is composed of only inliers).
How many samples?

Choose $N$ so that, with probability $p$, at least one random sample is free from outliers. e.g. $p=0.99$

$$(1 - (1 - e)^s)^N = 1 - p$$

$$N = \frac{\log(1 - p)}{\log (1 - (1 - e)^s)}$$

<table>
<thead>
<tr>
<th>$s$</th>
<th>5%</th>
<th>10%</th>
<th>20%</th>
<th>25%</th>
<th>30%</th>
<th>40%</th>
<th>50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>11</td>
<td>17</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>4</td>
<td>7</td>
<td>9</td>
<td>11</td>
<td>19</td>
<td>35</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>13</td>
<td>17</td>
<td>34</td>
<td>72</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>6</td>
<td>12</td>
<td>17</td>
<td>26</td>
<td>57</td>
<td>146</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>7</td>
<td>16</td>
<td>24</td>
<td>37</td>
<td>97</td>
<td>293</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>8</td>
<td>20</td>
<td>33</td>
<td>54</td>
<td>163</td>
<td>588</td>
</tr>
<tr>
<td>8</td>
<td>5</td>
<td>9</td>
<td>26</td>
<td>44</td>
<td>78</td>
<td>272</td>
<td>1177</td>
</tr>
</tbody>
</table>
Example: \( N \) for the line-fitting problem

- \( n = 12 \) points
- Minimal sample size \( s = 2 \)
- 2 outliers: \( e = 1/6 \Rightarrow 20\% \)
- So \( N = 5 \) gives us a 99\% chance of getting a pure-inlier sample
  - Compared to \( N = 66 \) by trying every pair of points

from Hartley & Zisserman
Acceptable consensus set?

- We have seen that we don’t have to exhaustively sample subsets of points, we just need to randomly sample N subsets.

- However, typically, we don’t even have to sample N sets!

- **Early termination**: terminate when inlier ratio reaches expected ratio of inliers

\[
T = (1 - e) \times (\text{total number of data points})
\]
RANSAC: Picking Distance Threshold $d$

- Usually chosen empirically
- But... when measurement error is known to be Gaussian with mean 0 and variance $s^2$:
  - Sum of squared errors follows a $\chi^2$ distribution with $m$ DOF, where $m$ is the DOF of the error measure (the codimension)
  - $(\text{dimension} + \text{codimension}) = \text{dimension of parameter space}$
    - E.g., $m = 1$ for line fitting because error is perpendicular distance
    - E.g., $m = 2$ for point distance
- Examples for probability $p = 0.95$ that point is inlier

<table>
<thead>
<tr>
<th>$m$</th>
<th>Model</th>
<th>$d^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Line, fundamental matrix</td>
<td>3.84 $s^2$</td>
</tr>
<tr>
<td>2</td>
<td>Homography, camera matrix</td>
<td>5.99 $s^2$</td>
</tr>
</tbody>
</table>
After RANSAC

- RANSAC divides data into inliers and outliers and yields estimate computed from minimal set of inliers with greatest support.
- Improve this initial estimate with Least Squares estimation over all inliers (i.e., standard minimization).
- Find inliers wrt that L.S. line, and compute L.S. one more time.

From Hartley & Zisserman
Practical Example

• Stabilizing aerial imagery using RANSAC
  - find corners in two images
  - hypothesize matches using NCC
  - do RANSAC to find matches consistent with an affine transformation
  - take the inlier set found and estimate a full projective transformation (homography)
Stabilization Application

Input: two images from an aerial video sequence.

Note that the motion of the camera is “disturbing”
Step 1: extract Harris corners from both frames. We use a small threshold for R because we want LOTS of corners (fodder for our next step, which is matching).
Step 2: hypothesize matches. For each corner in image 1, look for matching intensity patch in image 2 using NCC. Make sure matching pairs have highest NCC match scores in BOTH directions.
Stabilization Application

Step 2: hypothesize matches.

As you can see, a lot of false matches get hypothesized. The job of RANSAC will be to clean this mess up.
Step 3: Use RANSAC to robustly fit best affine transformation to the set of point matches.

\[ x_i' = a x_i + c y_i + c \]
\[ y_i' = d x_i + e y_i + f \]

How many unknowns?
How many point matches are needed?
Stabilization Application

Step 3: Use RANSAC to robustly fit best affine transformation to the set of point matches.

Affine transformation has 6 degrees of freedom. We therefore need 3 point matches [each gives 2 equations]

Randomly sample sets of 3 point matches. For each, compute the unique affine transformation they define. How?
Stabilization Application

How to compute affine transformation from 3 point matches? Use Least Squares! (renewed life for a nonrobust approach)

\[
\begin{bmatrix}
3x_i^2 & 3x_iy_i & 3x_i & 0 & 0 & 0 \\
3x_iy_i & 3y_i^2 & 3y_i & 0 & 0 & 0 \\
3x_i & 3y_i & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 3x_i^2 & 3x_iy_i & 3x_i \\
0 & 0 & 0 & 3x_iy_i & 3y_i^2 & 3y_i \\
0 & 0 & 0 & 3x_i & 3y_i & 1
\end{bmatrix}
\middle/\begin{bmatrix}
x_i' \\
y_i' \\
1
\end{bmatrix} =\begin{bmatrix}
x_i \\
y_i \\
1
\end{bmatrix}
\]

Then transform all points from image1 to image2 using that computed transformation, and see how many other matches confirm the hypothesis.

Repeat N times.
Stabilization Application

original point matches

labels from RANSAC
  green: inliers
  red: outliers
Stabilization Example

Step 4: Take inlier set labeled by RANSAC, and now use least squares to estimate a projective transformation that aligns the images. (we will discuss this ad nauseum in a later lecture).
Stabilization Example

Step 4: estimate projective transformation that aligns the images.

Now it is easier for people (and computers) to see the moving objects.
Stabilization Examples
Stabilization Examples

original point matches

labels from RANSAC
green: inliers
red: outliers
Stabilization Examples
Stabilization Examples
Stabilization Examples

original point matches

labels from RANSAC
  green: inliers
  red: outliers
Stabilization Examples