Targil 11:
Robust Techniques
RANSAC
Hough Transform

What are robust methods

Goal of many algorithms: extract parametric information from an image (parametric fitting)
- Extracting edges
- Extracting motion model from correspondence points

In real images: inaccurate data
- Occlusions
- Noise
- Wrong feature extraction
- Data is likely to include multiple models/structures

Goal of robust method: tolerate outliers points

Example: Robust Computation of 2D Homography

from Hartley&Zisserman
Interest points (500/image)
(640x480)

assumed correspondences (268)
(Best match, SSD<20, ±320)
Outliers (117)
(40 iterations)
Inliers (151)

Final inliers (262)

Simple Model Fit Problem
Estimate a straight line fit to a set of 2D points

Simple Model Fit Problem
Least Square (LS) error is not robust: one outlier point may cause a very big mistake

A good LS fit to a set of points.
One outlier point leads to disaster.
Robust Norm

Another outlier, another disaster.

Close-up of the poor fit to the "true" data.

Robust Norm

A quadratic $\rho$ function gives too much weight to outliers instead, use robust norm:

$$\rho(r, \sigma) = \frac{r^2}{\sigma^2 + r^2}$$

Parameters:
- $\sigma$: unknown scale

The impact of tuning $\sigma$

Just right

Too small

Too large

Simple Model Fit Problem

Estimate a straight line fit to a set of 2D points

How can we do better?

Assumption: Given the correct model, we can validate this model

RANSAC (RANdom SAmple Consensus)

The idea: Randomly choose points and check the support of model resulted from those points — choose model with max support

Support of the line: the number of points that lie within a distance threshold
RANSAC – The Algorithm

Iterate N times:
- Randomly Choose S points from the data
- Solve the problem using the S points
- Check the solution using the rest of the points (the support)

At the end:
Use all the points which "agree" to compute the exact model!

RANSAC – The Algorithm

How Many Iterations are needed?

We want to ensure that in probability p at least one of the samples of S points is free from outliers (for example p=99%)

\[ w = \text{inliers probability} \]

What is N?

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

\[
N = \frac{\log(1 - p)}{\log(1 - w^e)}
\]

Sufficient Number of Iterations (N)

$p = 0.99$

\[ w : \text{inliers probability}, \quad e = 1 - w : \text{outlier probability} \]

<table>
<thead>
<tr>
<th>Sample Size</th>
<th>2%</th>
<th>5%</th>
<th>10%</th>
<th>25%</th>
<th>50%</th>
<th>75%</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>9</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>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>13</td>
<td>17</td>
<td>24</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>6</td>
<td>12</td>
<td>17</td>
<td>26</td>
<td>57</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>7</td>
<td>16</td>
<td>24</td>
<td>37</td>
<td>97</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>8</td>
<td>20</td>
<td>33</td>
<td>54</td>
<td>163</td>
</tr>
<tr>
<td>8</td>
<td>5</td>
<td>9</td>
<td>26</td>
<td>44</td>
<td>78</td>
<td>272</td>
</tr>
</tbody>
</table>

Adaptively determining the number of Iterations

\[ e \] is often unknown a priori, so pick worst case, e.g. 50%, and adapt if more inliers are found, e.g. 80% would yield \( e = 0.2 \)

- \( N=\infty, \text{sample\_count} = 0 \)
- While \( N > \text{sample\_count} \) repeat
  - Choose a sample and count the number of inliers
  - Set \( e = 1 - (\text{number of inliers})/(\text{total number of points}) \)
  - Recompute \( N \) from \( e \)
  - Increment the \text{sample\_count} by 1
- Terminate

RANSAC discussion

Advantages:
- General method suited for a wide range of model fitting problems
- Easy to implement and easy to calculate its failure rate

- Disadvantages:
  - Only handles a moderate percentage of outliers without cost blowing up
  - Many real problems have high rate of outliers (but sometimes selective choice of random subsets can help)

Hough Transform

(proposed by Hough in 1959)

Finding Shapes
• Straight Lines
• Rectangles
• Circles
• Others

More generally:
A parametric voting
Applications

- Find signs in the images (rectangles, circles).
- 3D reconstruction of buildings (lines).
- Find the eye in a face image.
- Tracking
- More...

The challenge:

- The image contains a lot of irrelevant information.
- There are partial occlusions.
- The image might be noisy

Edge Detection Review

1) Noise Smoothing
2) Edge Enhancement
3) Threshold

INPUT IMAGE

EDGE IMAGE

“GRADIENT” IMAGE

How Do we detect lines from Edge Images

- Given an edge image, we would like to group sets of edge pixels to identify dominant lines in an image.
- One possibility is template matching in the edge image
  - Generate hypothetical line “kernel” and convolve with the image
  - Requires a large set of masks to accurately localize lines, which is computationally expensive
  - We will instead perform an image transformation where edge pixels vote for lines
  - Eventually reduces the problem to vote thresholding

The Parameters Space

The line equation: \( \frac{y}{a} = x + b \) (slope-intercept representation)

Each line can be represented with 2 parameters: (a,b)
If we define the a parapets space, each line is a point in this space.

For each point on this line (x,y), there are infinite number of lines that goes through it in the parameters space:

\( b = x + a \cdot y \)
The Duality Property

A point is mapped to a line, and a line is mapped to a point.

Hough Transform: use a voting scheme

- Apply an edge detector on the image (gradient + threshold, or using Canny’s method)
- Prepare a table \( h(a,b) \) for \( a,b \) in a certain range.
- Loop over the image: for each pixel \((x,y)\) on an edge, vote for the cells \((a,b)\) which satisfy the equation \( y=ax+b \). (increase the cell accumulator \( h(a,b)++ \))
- Choose the cell with the maximal value (or the cells which pass a given threshold)

Example: Slope-Intercept Representation

Original image

Edge Map

Hough (before threshold)

Take two strongest lines

Issues to think of...

- **Representation**: Vertical lines are not covered by the representation \( y=ax+b \).
- **Range**: Selecting the range of values for the table (\( a \) and \( b \) are not bounded!)
- **Quantization**: choosing a grid.
- **Thresholding**: Similar lines will get similar scores.

A better representation

Use a polar representation instead:

\[
d = x \cdot \cos(\theta) + y \cdot \sin(\theta)
\]

\((\cos(\theta), \sin(\theta))\) - The line normal

\(d\) - The distance from \((0,0)\)

In the polar representation...

Points transform to an alternative \(r,\theta\) space

Intersection points estimate the line equation
Another Example

Original image + edges

Hough (before threshold)

Image + detected lines

Range and Quantization

In the polar representation it is easier to determine the range:

- The values of d and of θ are now bounded!
- The resolution can be application dependent. We have natural units (pixels for d, and degrees for θ)

Quantization:

- Option1: Vote for the “nearest” cell (i.e., for each θ take the nearest d).
- Option2: Perform a weighted vote.

Smoothing

Without smoothing

Hough

Image

With smoothing

Hough

Image

Making the Decision

Q. How can we find the two most dominant lines in the image?

The problem: if (r, θ) is the cell with the maximal value, then (r+ε, θ+ε) will get a high score too.

Solution 1: Search only for local maxima.

Solution 2: “Back Mapping” - Perform a second vote, where each point votes only for the best line passing through it.

Using more image features.

Example – using the local gradient

not using gradient

Hough

Image

using gradient

Hough

Image
Using more image features.

Example – using the local gradient

Detecting Complex Shapes – Circles

**Naïve solution:**
- Construct a table $h(a,b,r)$.
- For each pixel $(x,y)$ which is edge in the image, vote for all the cells satisfying:

$$ (x - a)^2 + (y - b)^2 = r^2 $$

**Problem:**
- The size of the table is $O(N^3)$.
- The voting for each pixel takes: $O(N^2)$.

**Solution:** Use the gradient information -
(Solve for the center $(a,b)$ and then for $r$)

Circle Example

With no orientation, each token (point) votes for all possible circles. With orientation, each token can vote for a smaller number of circles.

Finding Coins

Original

Edges (note noise)

Finding Coins (Continued)

Penny

Quarters

Finding Coins (Continued)

Note that because the quarters and penny are different sizes, a different Hough transform (with separate accumulators) was used for each circle size.

Coin finding sample images from: Vivik Kwatra
Another Example

How can we find the “dominant” direction of the lines?