Stereo

Readings
- Trucco & Verri, Chapter 7
  - Read through 7.1, 7.2.1, 7.2.2, 7.3.1, 7.3.2, 7.3.7 and 7.4, 7.4.1.
  - The rest is optional.
Stereo

- Scene point
- Image plane
- Optical center

Basic Principle: Triangulation
- Gives reconstruction as intersection of two rays
  - Requires
    - Camera pose (calibration)
    - Point correspondence
Stereo correspondence

Determine Pixel Correspondence

- Pairs of points that correspond to same scene point

Epipolar Constraint

- Reduces correspondence problem to 1D search along conjugate epipolar lines

Fundamental matrix

- This epipolar geometry of two views is described by a Very Special 3x3 matrix \( F \), called the fundamental matrix
- \( F \) maps (homogeneous) points in image 1 to lines in image 2!
- The epipolar line (in image 2) of point \( p \) is: \( Fp \)
- Epipolar constraint on corresponding points: \( q^TFp = 0 \)

Fundamental matrix – uncalibrated case

- \( K_1 \) : intrinsics of camera 1
- \( K_2 \) : intrinsics of camera 2
- \( R \) : rotation of image 2 w.r.t. camera 1

\[
q^T K_2^{-T} R [t]_\times K_1^{-1} p = 0
\]

Cross-product as linear operator

**Useful fact:** Cross product with a vector \( t \) can be represented as multiplication with a (skew-symmetric) 3x3 matrix

\[
[t]_\times = \begin{bmatrix}
0 & -t_z & t_y \\
t_z & 0 & -t_x \\
-t_y & t_x & 0
\end{bmatrix}
\]

\[
t \times \tilde{p} = [t]_\times \tilde{p}
\]
Fundamental matrix – calibrated case

\[ \hat{p} = K_1^{-1} p \]
\[ \hat{q} = K_2^{-1} q \]

Ray through \( p \) in camera 1’s (and world) coordinate system
Ray through \( q \) in camera 2’s coordinate system

\[ \hat{q}^T R [t] \hat{p} = 0 \]
\[ \hat{q}^T E \hat{p} = 0 \]

\( E \) is the Essential matrix

Properties of the Fundamental Matrix

- \( F \hat{p} \) is the epipolar line associated with \( p \)
- \( F^T \hat{q} \) is the epipolar line associated with \( q \)
- \( F e_1 = 0 \) and \( F^T e_2 = 0 \)
- \( F \) is rank 2
- How many parameters does \( F \) have?

Recalculated case

\[ R = I_{3 \times 3} \]
\[ t = [1 \ 0 \ 0]^T \]

\[ E = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \]

Stereo image rectification
Stereo image rectification

- Reproject image planes onto a common plane parallel to the line between optical centers
- Pixel motion is horizontal after this transformation
- Two homographies (3x3 transform), one for each input image reprojection


Estimating F

- If we don’t know $K_1$, $K_2$, $R$, or $t$, can we estimate $F$ for two images?
- Yes, given enough correspondences. We’ll see soon...

Stereo Matching

Given a pixel in the left image, how to find its match?

- Assume the photos have been rectified

Your basic stereo algorithm

For each epipolar line
- For each pixel in the left image
  - Compare with every pixel on same epipolar line in right image
  - Pick pixel with minimum match cost

Improvement: match windows
- This should look familiar...
Window size

Effect of window size
- Smaller window
- Larger window

Stereo results
- Data from University of Tsukuba
- Similar results on other images without ground truth

Results with window search
- Window-based matching
- (best window size)
- Ground truth

Better methods exist...
- State of the art method
- Ground truth

Boykov et al., *Fast Approximate Energy Minimization Via Graph Cuts*, International Conference on Computer Vision, September 1999.

For the latest and greatest: [http://www.middlebury.edu/stereo/](http://www.middlebury.edu/stereo/)
What defines a good stereo correspondence?

1. **Match quality**
   - Want each pixel to find a good match in the other image

2. **Smoothness**
   - If two pixels are adjacent, they should (usually) move about the same amount

---

**Simple pixel / window matching**

- Choose the minimum of each column in the DSI independently:

$$d(x, y) = \arg\min_{d'} C(x, y, d')$$
Stereo as energy minimization

Better objective function

\[ E(d) = E_d(d) + \lambda E_s(d) \]

- **Match cost**: \( E_d(d) = \sum_{(x,y) \in I} C(x, y, d(x, y)) \)
- **Smoothness cost**: \( E_s(d) = \sum_{(p,q) \in \mathcal{E}} V(d_p, d_q) \)

\( \mathcal{E} \): set of neighboring pixels

**Match cost**
- 4-connected neighborhood
- 8-connected neighborhood

**Smoothness cost**

\[ E_s(d) = \sum_{(p,q) \in \mathcal{E}} V(d_p, d_q) \]

How do we choose \( V \)?

\[ V(d_p, d_q) = |d_p - d_q| \]

"Potts model"

**Dynamic programming**

\[ E(d) = E_d(d) + \lambda E_s(d) \]

Can minimize this independently per scanline using dynamic programming (DP)

\[ D(x, y, d) : \text{minimum cost of solution such that } d(x, y) = d \]

\[ D(x, y, d) = C(x, y, d) + \min \{ D(x-1, y, d') + \lambda |d - d'| \} \]
Dynamic programming

Finds "smooth" path through DPI from left to right

Dynamic Programming

Can we apply this trick in 2D as well?

No: \( d_{x+1,y} \) and \( d_{x,y+1} \) may depend on different values of \( d_{x+1,y+1} \)

Stereo as a minimization problem

\[
E(d) = E_d(d) + \lambda E_s(d)
\]

The 2D problem has many local minima
- Gradient descent doesn’t work well

And a large search space
- \( n \times m \) image w/ \( k \) disparities has \( k^{nm} \) possible solutions
- Finding the global minimum is NP-hard in general
Stereo as global optimization

Expressing this mathematically

1. Match quality
   - Want each pixel to find a good match in the other image
   \[ \text{matchCost} = \sum_{x,y} |I(x,y) - I(x + d_{xy}, y)| \]

2. Smoothness
   - If two pixels are adjacent, they should (usually) move about the same amount
   \[ \text{smoothnessCost} = \sum_{\text{neighbor} \ x,y} |d_y - d_y'| \]

We want to minimize sum of these two cost terms

• This is a special type of cost function known as an MRF (Markov Random Field)
  - Effective and fast algorithms have been recently developed:
    » Graph cuts, belief propagation....
    » For more details (and code): [http://vision.middlebury.edu/MRF/](http://vision.middlebury.edu/MRF/)

Middlebury Stereo Evaluation

[http://vision.middlebury.edu/stereo/](http://vision.middlebury.edu/stereo/)

Depth from disparity

\[ \text{disparity} = x - x' = \frac{\text{baseline} \cdot f}{z} \]

Real-time stereo

Nomad robot searches for meteorites in Antarctica
[http://www.frc.ri.cmu.edu/projects/meteorobot/index.html](http://www.frc.ri.cmu.edu/projects/meteorobot/index.html)

Used for robot navigation (and other tasks)

• Several software-based real-time stereo techniques have been developed (most based on simple discrete search)
Stereo reconstruction pipeline

Steps
- Calibrate cameras
- Rectify images
- Compute disparity
- Estimate depth

What will cause errors?
- Camera calibration errors
- Poor image resolution
- Occlusions
- Violations of brightness constancy (specular reflections)
- Large motions
- Low-contrast image regions

Active stereo with structured light

Project "structured" light patterns onto the object
- simplifies the correspondence problem
- can remove one of the cameras (replace with projector)

Active stereo with structured light

Laser scanning

Optical triangulation
- Project a single stripe of laser light
- Scan it across the surface of the object
- This is a very precise version of structured light scanning
Laser scanned models

The Digital Michelangelo Project, Levoy et al.

Laser scanned models

The Digital Michelangelo Project, Levoy et al.

Laser scanned models

The Digital Michelangelo Project, Levoy et al.

Laser scanned models

The Digital Michelangelo Project, Levoy et al.
Laser scanned models

---

The Digital Michelangelo Project, Levoy et al.

---

Estimating $F$

- If we don’t know $K_1$, $K_2$, $R$, or $t$, can we estimate $F$ for two images?
- Yes, given enough correspondences

---

Estimating $F$ – 8-point algorithm

- The fundamental matrix $F$ is defined by

$$\mathbf{x}^T \mathbf{F} \mathbf{x} = 0$$

for any pair of matches $x$ and $x'$ in two images.
- Let $\mathbf{x} = (u, v, 1)^T$ and $\mathbf{x}' = (u', v', 1)^T$, $\mathbf{F} = \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix}$

  each match gives a linear equation

$$uu'f_{11} + vu'f_{12} + u'f_{13} + vv'f_{21} + uv'f_{22} + v'f_{23} + vuf_{31} + vif_{32} + f_{33} = 0$$

---

8-point algorithm

$$\begin{bmatrix} u_i u'_i & v_i u'_i & u'_i & u_i v'_i & v_i v'_i & v'_i & u_i & v_i & 1 \\ u_2 u'_2 & v_2 u'_2 & u'_2 & u_2 v'_2 & v_2 v'_2 & v'_2 & u_2 & v_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_n u'_n & v_n u'_n & u'_n & u_n v'_n & v_n v'_n & v'_n & u_n & v_n & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = 0$$

- In reality, instead of solving $\mathbf{A}\mathbf{f} = 0$, we seek $\mathbf{f}$

  to minimize $|\mathbf{A}\mathbf{f}|$ least eigenvector of $\mathbf{A}^T\mathbf{A}$.
8-point algorithm – Problem?

- \( F \) should have rank 2
- To enforce that \( F \) is of rank 2, \( F \) is replaced by \( F' \) that minimizes \( \| F - F' \| \) subject to the rank constraint.
- This is achieved by SVD. Let \( F = U \Sigma V^T \), where
  \[
  \Sigma = \begin{bmatrix}
  \sigma_1 & 0 & 0 \\
  0 & \sigma_2 & 0 \\
  0 & 0 & \sigma_3 \\
  \end{bmatrix}, \quad \text{let} \quad \Sigma' = \begin{bmatrix}
  \sigma_1 & 0 & 0 \\
  0 & \sigma_2 & 0 \\
  0 & 0 & 0 \\
  \end{bmatrix}
  \]
  then \( F' = U \Sigma' V^T \) is the solution.

8-point algorithm

• Pros: it is linear, easy to implement and fast
• Cons: susceptible to noise

8-point algorithm

% Build the constraint matrix
\[
A = [x2(1,:)*x1(1,:)', x2(1,:)*x1(2,:)', x2(1,:)';
    x2(2,:)*x1(1,:)', x2(2,:)*x1(2,:)', x2(2,:)';
    x1(1,:)', x1(2,:)';
    ones(npts,1)];
\]

[U,D,V] = svd(A);
% Extract fundamental matrix from the column of V corresponding to the smallest singular value.
\[
F = reshape(V(:,9),3,3)';
\]
% Enforce rank2 constraint
[U,D,V] = svd(F);
\[
F = U*diag([D(1,1) D(2,2) 0])*V';
\]