Binocular Stereo

- Take 2 images from different known viewpoints ⇒ 1st calibrate
- Identify corresponding points between 2 images
- Derive the 2 lines on which world point lies
- Intersect 2 lines
Stereo

- Basic Principle: Triangulation
  - Gives reconstruction as intersection of two rays
  - Requires
    • calibration
    • point correspondence

Depth from Disparity

\[ \text{disparity} = u - u' = \frac{\text{baselines}}{f} \]
Multi-View Geometry

- Different views of a scene are not unrelated
- Several relationships exist between two, three and more cameras

Question: Given an image point in one image, does this restrict the position of the corresponding image point in another image?

Epipolar Geometry: Formalism

- Depth can be reconstructed based on corresponding points (disparity)
- Finding corresponding points is hard & computationally expensive
- Epipolar geometry helps to significantly reduce search from 2-D to 1-D line

Epipolar Geometry: Demo

Java Applet

http://www-so.pierre.inria.fr/robotvis/personnel/sbougnou/Meta3DViewer/EpipolarGeo.html

Sylvain Bougnoux, INRIA Sophia Antipolis
• Scene point $P$ projects to image point $p_l = (x_l, y_l, f_l)$ in left image and point $p_r = (x_r, y_r, f_r)$ in right image
• Epipolar plane contains $P$, $O_l$, $O_r$, $p_l$ and $p_r$ – called **co-planarity constraint**
• Given point $p_l$ in left image, its corresponding point in right image is on line defined by intersection of epipolar plane defined by $p_l$, $O_l$, $O_r$ and image $I_r$ – called **epipolar line** of $p_l$
• In other words, $p_l$ and $O_l$ define a ray where $P$ may lie; projection of this ray into $I_r$ is the **epipolar line**

**Epipolar Line Geometry**

• **Epipolar Constraint**: The correct match for a point $p_l$ is constrained to a 1D search along the epipolar line in $I_r$
• All epipolar planes defined by all points in $I_l$ contain the line $O_l O_r$
  $\Rightarrow$ All epipolar lines in $I_r$ intersect at a point, $e_r$, called the **epipole**
• Left and right epipoles, $e_l$ and $e_r$, defined by the intersection of line $O_l O_r$ with the left and right images $I_l$ and $I_r$, respectively
Epipolar Geometry

- If $I_x$ and $I_y$ are parallel, the epipoles are at infinity and the epipolar lines are parallel.
- Given a pair of images, $I_x$ and $I_y$, the warping transformation that projects $I_x$ and $I_y$ onto a plane parallel to $O_B$ is called rectification. Usually done so that epipolar lines are parallel to new images’ horizontal axes.

$\Rightarrow$ Epipolar constraint is a search along image scanlines.

Epipolar Geometry: Rectification

- [Trucco 157-160]
- **Motivation**: Simplify search for corresponding points along scan lines (avoids interpolation and simplify sampling)
- **Technique**: Image planes parallel $\Rightarrow$ pairs of conjugate epipolar lines become collinear and parallel to image axis.
Stereo Image Rectification

- Image Reprojection
  - reproject image planes onto common
    plane parallel to line between optical centers
  - a homography (3x3 transform)
    applied to both input images
  - pixel motion is horizontal after this transformation

Rectification Example

before

after

Rectification Procedure
Given: Intrinsic and extrinsic parameters for 2 cameras

1. Rotate left camera so that the epipole goes to infinity along the horizontal axis
   ⇒ left image parallel to baseline
2. Rotate right camera using same transformation
3. Rotate right camera by R, the transformation of the right camera frame with respect to the left camera
4. Adjust scale in both cameras

Implement as backward transformations, and resample using bilinear interpolation
Definitions

- **Conjugate Epipolar Line**: A pair of epipolar lines in $I_l$ and $I_r$ defined by $P$, $O_l$ and $O_r$.

- **Conjugate (i.e., corresponding) Pair**: A pair of matching image points from $I_l$ and $I_r$ that are projections of a single scene point.
Notation and Definitions

- Given \( a = (a_1, a_2, a_3)^T \)
  
  then \[ [a]_x = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \]

  a 3x3 skew-symmetric matrix
  and Singular matrix

- Given 2 3-vectors, \( a \) and \( b \),
  
  \[ a \times b = a \wedge b = [a]_x b = (a^T b_n)^T \]

- Equivalently, epipolar equation can be written as:
  
  \[ P_r^T E P_r = 0 \]

  where \( E^T = -[t_s]_R^T \)

- Observation: \( E_p \) can be interpreted as the vector representing the epipolar line associated with \( p_k \) in the right image. So, the epipolar equation \( P_r^T E P_r = 0 \)
  
  expresses the fact that \( p_k \) lies on the epipolar line associated with the vector \( E_p \)

- \( E = [T]_R \) (is usual way of writing \( E \))
  
  is 3x3 matrix, \( 8 \) DOFs
  (3 for \( T \), 3 for \( R \),
  but an overall scale ambiguity)

- Essentially, epipolar equation can be written as:

  \[ P_r^T E P_r = 0 \]

  where \( E = RS \)

  \( E \) called Essential Matrix

  Using perspective projection equation, can also show:

  \[ P_r^T E P_r = 0 \]

  For change points \( p_r \) and \( p_k \) in
  \( I_r \) and \( I_k \), respectively, though
  \( p_r \) and \( p_k \) are in camera coordinates.

  But points known in image coordinates
  and need to know transformation
  from camera co-ords to image co-ords,
  i.e., intrinsic parameters of camera
  (pixel height, width of principal pt. etc. pixels)

- Essential matrix encodes information about the extrinsic parameters
  (rotation and translation) only,
  \( E \) is defined in terms of camera coordinates.

- If \( P_r \) and \( P_k \) are corresponding points
  in image (pixel) coordinates, then it
  can be shown:

  \[ P_r^T F P_k = 0 \]

  where \( F = H_r^T E H_k \) an Fundamental Matrix

  \[ H_r = \begin{pmatrix} 0 & -\overline{y}_1 & \overline{y}_2 \\ \overline{y}_1 & 0 & -\overline{x}_1 \\ -\overline{x}_1 & \overline{y}_2 & 0 \end{pmatrix} \]

  Intrinsic parameters of left camera:
  \( f_x \) : focal length
  \( (c_x, c_y) \) : center of principal pt.
  \( (x_1, y_1) \) : effective pixel size
Fundamental Matrix Properties
- Encoder data, about both.
- Intrinsic and Extrinsic Parameters.
- If you can estimate P, you can reconstruct the epipolar geometry with no info about cameras.
- HAS RANK 2.
- 3 x 3.
- HAS 7 DOF.
- Can be estimated from at least 8 correspondences using the RANSAC algorithm.
- Each point gives a homogeneous linear equation. \( \sum \mathbf{p}_i \mathbf{F} \mathbf{p}_i = 0 \).
- Homogeneous system has solution unique up to scale factor.
- Usually, use 8-10 points so system is overdetermined and solve using SVD.

Computing the Fundamental Matrix
Given a correspondence \( \mathbf{x} \equiv \mathbf{x}' \), we know \( \mathbf{x}'^T \mathbf{F} \mathbf{x} = 0 \) or \( \mathbf{x} \mathbf{x}'^T + \mathbf{x}' \mathbf{x}^T + \mathbf{x} \mathbf{F}^T + \cdots + \mathbf{F}^T_3 = 0 \) or \( (\mathbf{x}, \mathbf{x}', \mathbf{x}, \mathbf{x}', \mathbf{x}', \mathbf{x}', \mathbf{x}' \mathbf{x}', \mathbf{x}') \mathbf{F}_3 = 0 \) with \( \mathbf{F}_3 \) given by

\[
\begin{pmatrix}
F_{11} & F_{12} & F_{13} \\
F_{21} & F_{22} & F_{23} \\
F_{31} & F_{32} & F_{33}
\end{pmatrix}
\]

or

\[
\begin{pmatrix}
F_{11} & F_{12} & F_{13} \\
F_{21} & F_{22} & F_{23} \\
F_{31} & F_{32} & F_{33}
\end{pmatrix}
\]

The Epipolar Constraint
\[
\mathbf{F}_{x'}^T \mathbf{F} \mathbf{F}_{x} = 0
\]

\[
(\mathbf{u}, \mathbf{v}, 1) \begin{pmatrix}
\mathbf{F}_{x'} & \mathbf{F}_{x} \\
\mathbf{F}_{x'} & \mathbf{F}_{x}
\end{pmatrix} \begin{pmatrix}
\mathbf{u}' \\
\mathbf{v}'
\end{pmatrix} = 0
\]

\[
(\mathbf{u} \mathbf{v}, \mathbf{u} \mathbf{v}, \mathbf{u} \mathbf{v}, \mathbf{u} \mathbf{v}, \mathbf{v} \mathbf{u}, \mathbf{v} \mathbf{u}, \mathbf{v} \mathbf{u}, 1) \begin{pmatrix}
\mathbf{F}_{x'} & \mathbf{F}_{x} \\
\mathbf{F}_{x'} & \mathbf{F}_{x}
\end{pmatrix} \begin{pmatrix}
\mathbf{F}_{x'} & \mathbf{F}_{x}
\end{pmatrix} = 0
\]

Homogeneous equation in coefficients of \( \mathbf{F} \).
Given $m$ conjugate pairs

$$AF = \begin{bmatrix} x_{11} & x_{12} & \ldots & x_{1n} \\ x_{21} & x_{22} & \ldots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \ldots & x_{mn} \end{bmatrix} = 0$$

- Solution determined up to scale only
  - Need at least 5 point correspondences
- 8 points $\Rightarrow$ unique solution
- $F^T = D$ least-squares solution
  1. Form equations $AF = 0$
  2. Take SVD $A = UDV^T$
  3. Solution is last column of $V$ (argmin norm subject to smallest eigenvalue)

Fundamental matrix has rank 2: $\det(F) = 0$.

Left: Uncorrected $F$ — epipolar lines are not coincident.
Right: Epipolar lines from corrected $F$.

Least-squares solution does not enforce the singularity constraint — all epipolar lines intersect at epipole

$\Rightarrow \det(F) = 0$ ($F$ has rank 2)

8-point Algorithm (Hartley)

1. Compute linear solution
   - Solve $AF = 0$ to find $F$
2. Constraint enforcement
   - Replace $F$ by $F'$, the "closest" singular matrix to $F$

8-point algorithm performs badly with noise:
- Sensitive to origin position
- Translation and scaling of 8 data points
  - $\Rightarrow$ translate and scale points
    origin $\Rightarrow$ centroid of points
    scale $2 \times$ "average scale" (not $\lambda_{max}$)

Normalized 8-point Algorithm

1. Normalization
   $\tilde{x}_i^c = Tx_i^c$
   $\tilde{x}_i^{c'} = T^{c'}x_i^{c'}$
2. Linear Solution
   $\tilde{F}$ by solving $A\tilde{F} = 0$
3. Singularity Constraint
   Find closest singular $F'$ to $\tilde{F}$
4. Denormalization
   $F = T^{-1}F'T$
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

Stereo Correspondence

left image \( (x_1, y) \)  right image \( (x_2, y) \)  disparities

\[
\text{disparity} = x_1 - x_2 \quad \text{is inversely proportional to depth}
\]

3D scene structure recovery

\[ \text{triangular epipolar geometry} \]

\[
\begin{align*}
E_{12} E_{13} &= 0 \\
E_{21} E_{23} &= 0 \\
E_{31} E_{32} &= 0
\end{align*}
\]

(can compute \( f_3 \) from image "beats"
Stereo Matching

• Features vs. pixels?
  – Do we extract features prior to matching?

Julesz-style Random Dot Stereogram

Difficulties in Stereo Correspondence

Perfect case: never happens!

1) Image noise:

2) Low texture: ?
Local Approach

- Look at one image patch at a time
- Solve many small problems independently
- Faster, less accurate

Global Approach

- Look at the whole image
- Solve one large problem
- Slower, more accurate

How Difficult is Correspondence?

- **local** works for high texture
  - enough texture in a patch to disambiguate

- **global** works up to medium texture
  - propagates estimates from textured to untextured regions

- **salient regions** work up to low texture
  - propagation fails; some regions are inherently ambiguous, match only unambiguous regions

Local Approach [Levine’73]

- $d_p = i$ which gives best $C_i$

Common $C_{(SSD)} = \sum (C_p - C_i)^2$
Window Size

- Effect of window size
  - Smaller window: Better results with adaptive window
  - Larger window: Worse results

Sample Compact Windows [Veksler 2001]

Comparison to Fixed Window

- True disparities
- Veksler's compact windows: 16% errors
- Fixed small window: 33% errors
- Fixed large window: 30% errors

Results (% Errors)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Tsukuba</th>
<th>Venus</th>
<th>Sawtooth</th>
<th>Map</th>
</tr>
</thead>
<tbody>
<tr>
<td>Layered</td>
<td>1.58</td>
<td>1.52</td>
<td>0.34</td>
<td>0.37</td>
</tr>
<tr>
<td>Graph cuts</td>
<td>1.94</td>
<td>1.79</td>
<td>1.30</td>
<td>0.31</td>
</tr>
<tr>
<td>Belief prop</td>
<td>1.15</td>
<td>1.00</td>
<td>0.98</td>
<td>0.84</td>
</tr>
<tr>
<td>GC+occl.</td>
<td>1.27</td>
<td>2.79</td>
<td>0.36</td>
<td>1.79</td>
</tr>
<tr>
<td>Graph cuts</td>
<td>1.86</td>
<td>1.69</td>
<td>0.42</td>
<td>2.39</td>
</tr>
<tr>
<td>Multiw. Cut</td>
<td>8.08</td>
<td>0.53</td>
<td>0.61</td>
<td>0.26</td>
</tr>
<tr>
<td>Veksler's var. windows</td>
<td>3.36</td>
<td>1.67</td>
<td>1.61</td>
<td>0.33</td>
</tr>
</tbody>
</table>
Constraints

1) corresponding pixels should be close in color

2) most nearby pixels should have similar disparity

except a few places: disparity discontinuity

Additional geometric constraints for correspondence

- Ordering of points:
  Continuous surface: same order in both images.
- Is that always true?

Forbidden Zone

Practical applications:
- Object bulges out: ok
- In general: ordering across whole image is not reliable feature
- Use ordering constraints for neighbors of M within small neighborhood only
**Disparity Gradient** (Pollard, Mayhew and Frisby 1985) (Prattley 1985)
- pair of edges from same surface in scene appear with similar disparities
- allowed disparity difference increases with separation between matches
- sometimes discriminates only weakly

**Figural Continuity:** (Mayhew and Frisby 1981)
- edges on a contour in one image match edges along a similar contour in the other image
- non-contour edges do not meet requirements

**Continuity Constraint**
(Marr and Poggio; Grimson)
- nearby image points are projections of nearby 3D points
  ⇒ smoothness in disparity map
- doesn't apply at region boundaries and non-opaque objects

**Image Scene:**
- each of the four points in one view could match one of the four projections in the other view. If the possible matches exceed the noise and the scene contains the same areas or nearby points, disparities can be used to determine the 3D structure. (From Ayton, 1981, left: L. J. Mayhew and L. E. Frisby, 1981, 1985, 1987, and 1988.)
Types of Stereo Algorithms

1. Local Methods based on Correlation
   - Normalized cross-correlation
   - SSD match using max window centered on each point
   - Compute dense depth map

2. Global Optimization
   - Define an energy function
     \[ E(F) = E_{\text{smooth}}(F) + E_{\text{data}}(F) \]
   - where \( F \) is the disparity value at a given pixel, \( p \)
   - Example:
     \[ E_{\text{data}} = \sum_{p} |F(p) - I_p(p + \text{disparity}(p))|^2 \]
   - \( E_{\text{smooth}} = \sum_{p} (\text{adjacent pixels of different disparity near } p) \)

Harr-Poppie Stereo Algorithm

1. Convolve 2 rectified images with \( \nabla^2 G_p \) filters of size
   \( \sigma_x < \sigma_y < \sigma_z < \sigma_y \)

2. Detect zero-crossings in all images

3. At constant scale, \( \psi_p \) match zero-crossings with same disparity
   and roughly same orientation in a \( \{ -\psi_p, +\psi_p \} \) disparity range
   with \( \psi_p = 2\pi / \sigma_y \)

4. Use disparities found at coarser scales to cause unmatched regions at finer scales to come into correspondence.

Result is a sparse depth map

- \( E_{\text{smooth}} \) should be piecewise-smooth, not smooth everywhere, to allow for depth discontinuities
- Minimize energy function \( E \)
- Using optimization methods e.g. dynamic programming, simulated annealing
- May find local minimum
- Computes dense depth map

2 Constraints on Harr-Poppie

1. **Uniqueness**
   - Each point in left image can match only 1 point in right image, corresponding to fact that a single disparity value can be assigned

2. **Continuity**
   - Surface smoothness \( \Rightarrow \) disparity smoothness almost everywhere (except at depth discontinuities — occluding contours)

3. **Multi-resolution**
   - Coarse-to-fine tracking
Zitnick & Kanade's Algorithm

www-2.cs.cmu.edu/~clb/stereo.html
IEEE Trans. PAMI 22(9), 2000

3D Disparity Space Representation
(r,c,d)

and where

\[ s_n(r,s,d) = \sum_{\Phi} L_n(r+c,s+c,d+d') \]

\( \alpha > 1 \)
\( \Phi \) corresponds to smoothness assumption
\( \Psi \) corresponds to uniqueness assumption

1. Construct 3D array \( (r,s,d) \)
   for each pixel in reference image
   and disparity range.

2. Compute initial match values
   \[ L_0(r,s,d) = \sum_{\Phi} (L_{1r} L_{1s} r,s,d) \]
   \( \rightarrow \) computes match between
   \( I_{1r}(r,s) \) and \( I_{1s}(r,s+d) \)

3. Iteratively update match values
   until match values converge.
   \[ L_{n+1}(r,s,d) = L_n(r,s,d) + R_n(r,s,d) \]
   where
   \[ R_n(r,s,d) = \left( \frac{\sum_{\Phi} S_n(r,s,d) \neg \Psi(r,s,d) \neg \Phi \in \text{inhibition area}}{\Psi(r,s,d) \neg \Phi} \right) \]

4. Converges to 1 at correct matches
5. To prevent over-smoothing & loss of detail

\( L_0 \# R_n \) means only pairs with similar initial intensities will contribute to match value computation.
Global Approach [Horn’81, Poggio’84, …]
encode desirable properties of \( d \) in \( E(d) \):

\[
E(d) = \sum_{p \in \text{Neighbors}} \min_d \left( d_p, d_q \right) - \sum_{p \in \text{Neighbors}} \left( \sum_{(p,q) \in \text{Neighbors}} P(d_p, d_q) \right)
\]

- \( \min_d \left( d_p, d_q \right) \) matches pixels of similar color
- \( \sum_{(p,q) \in \text{Neighbors}} P(d_p, d_q) \) most nearby pixels have similar disparity

NP-hard problem \( \Rightarrow \) need approximations

Stereo as Energy Minimization

- Matching cost formulated as energy
  - “data” term penalizing bad matches
    
    \[
    D(x, y, d) = [I(x, y) - J(x + d, y)]
    \]
  - “neighborhood term” encouraging spatial smoothness (continuity; disparity gradient)
    
    \[
    V(d_1, d_2) = \text{cost of adjacent pixels with labels } d_1 \text{ and } d_2
    \]
    
    \[
    = |d_1 - d_2| \quad (\text{or something similar})
    \]

\[
E = \sum_{(x,y)} D(x, y, d_{x,y}) + \sum_{\text{neighbors } (x_1,y_1),(x_2,y_2)} V(d_{x_1,y_1}, d_{x_2,y_2})
\]

Minimization Methods

1. Continuous \( d \): Gradient Descent
   - Gets stuck in local minimum

2. Discrete \( d \): Simulated Annealing
   [Geman and Geman, PAMI 1984]
   - Takes forever or gets stuck in local minimum
Stereo as a Graph Problem [Boykov, 1999]

Graph Definition

- **Initial state**
  - Each pixel connected to its immediate neighbors
  - Each disparity label connected to all of the pixels

Graph Cuts

- **Graph** $G=(V,E)$
- **Edge weight** $w: E \rightarrow \mathbb{R}^+$
- **Cost** $(C) = \sum_{\text{edges in } C} w(\text{edge})$
- **Problem**: find min Cost cut

- Solved in polynomial time w/ min-cut/max-flow
- Boykov and Kolmogorov algorithm
  - runs in seconds

Stereo Matching by Graph Cuts

- **Graph Cut**
  - Delete enough edges so that
    - each pixel is (transitively) connected to exactly one label node
  - Cost of a cut: sum of deleted edge weights
  - Finding min cost cut equivalent to finding global minimum of the energy function
### Results of Boykov’s Graph Cut Algorithm

<table>
<thead>
<tr>
<th>Local: Compact Window</th>
<th>Global: Expansion</th>
</tr>
</thead>
<tbody>
<tr>
<td>high texture</td>
<td></td>
</tr>
<tr>
<td>18 sec, 16% error</td>
<td>10 sec, 0.33% error</td>
</tr>
<tr>
<td>medium texture</td>
<td></td>
</tr>
<tr>
<td>75 sec, $\lambda = 5$</td>
<td>33 sec, $\lambda = 100$</td>
</tr>
<tr>
<td>12 sec, 3.36% error</td>
<td>32 sec, 1.86% error, $\lambda = 20$</td>
</tr>
</tbody>
</table>

#### Difficulties
- Parameter selection
- Running time: from 34 to 86 seconds

#### Computing a Multi-way Cut
- With two labels: classical min-cut problem
  - Solvable by standard network flow algorithms
    - Polynomial time in theory, nearly linear in practice
- More than 2 labels: NP-hard [Dahlhaus et al., STOC ’92]
  - But efficient approximation algorithms exist
    - Within a factor of 2 of optimal
    - Computes local minimum in a strong sense
      - Even very large moves will not improve the energy
  - Basic idea
    - Reduce to a series of 2-way-cut sub-problems, using one of:
      - Swap move: pixels with label $L_1$ can change to $L_2$, and vice-versa
      - Expansion move: any pixel can change its label to $L_1$
State of the Art

Late 90’s state of the art

left image true disparities

Recent state of the art

Evaluation of Stereo Algorithms

http://bj.middlebury.edu/~schar/stereo/web/results.php

“A taxonomy and evaluation of dense two-frame stereo correspondence algorithms,”

Int. J. Computer Vision, 2002

Database by D. Scharstein and R. Szeliski

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Tsukuba</th>
<th>Sawtooth</th>
<th>Venus</th>
<th>Map</th>
</tr>
</thead>
<tbody>
<tr>
<td>Layered</td>
<td>1.58</td>
<td>0.34</td>
<td>1.52</td>
<td>0.37</td>
</tr>
<tr>
<td>Graph cuts</td>
<td>1.94</td>
<td>1.30</td>
<td>1.79</td>
<td>0.31</td>
</tr>
<tr>
<td>Belief prop.</td>
<td>1.15</td>
<td>0.98</td>
<td>1.00</td>
<td>0.84</td>
</tr>
<tr>
<td>GC+occ.</td>
<td>1.27</td>
<td>0.36</td>
<td>2.79</td>
<td>1.79</td>
</tr>
<tr>
<td>Graph cuts</td>
<td>1.66</td>
<td>0.42</td>
<td>1.69</td>
<td>2.39</td>
</tr>
<tr>
<td>Multiw. cut</td>
<td>8.08</td>
<td>0.61</td>
<td>0.53</td>
<td>0.26</td>
</tr>
<tr>
<td>Comp. win.</td>
<td>3.36</td>
<td>1.61</td>
<td>1.67</td>
<td>0.33</td>
</tr>
<tr>
<td>Realtime</td>
<td>4.25</td>
<td>1.32</td>
<td>1.53</td>
<td>0.81</td>
</tr>
<tr>
<td>Bay. diff.</td>
<td>6.49</td>
<td>1.45</td>
<td>4.00</td>
<td>0.20</td>
</tr>
<tr>
<td>Cooperative</td>
<td>3.49</td>
<td>2.03</td>
<td>2.57</td>
<td>0.22</td>
</tr>
<tr>
<td>SSD+MF</td>
<td>5.23</td>
<td>2.21</td>
<td>3.74</td>
<td>0.66</td>
</tr>
<tr>
<td>Stoch. diff.</td>
<td>3.95</td>
<td>2.45</td>
<td>2.45</td>
<td>1.31</td>
</tr>
<tr>
<td>Genetic</td>
<td>2.96</td>
<td>2.21</td>
<td>2.49</td>
<td>1.04</td>
</tr>
<tr>
<td>Pix-to-pix</td>
<td>5.12</td>
<td>2.31</td>
<td>6.30</td>
<td>0.50</td>
</tr>
<tr>
<td>Max flow</td>
<td>2.98</td>
<td>3.47</td>
<td>2.16</td>
<td>3.13</td>
</tr>
<tr>
<td>Scanl. opt.</td>
<td>5.08</td>
<td>4.06</td>
<td>9.44</td>
<td>1.84</td>
</tr>
<tr>
<td>Dyn. prog.</td>
<td>4.12</td>
<td>4.84</td>
<td>10.1</td>
<td>3.33</td>
</tr>
<tr>
<td>Shio</td>
<td>9.67</td>
<td>4.25</td>
<td>6.01</td>
<td>2.36</td>
</tr>
<tr>
<td>MMHM</td>
<td>9.76</td>
<td>4.78</td>
<td>6.48</td>
<td>8.42</td>
</tr>
<tr>
<td>Max. surf.</td>
<td>11.10</td>
<td>5.51</td>
<td>4.36</td>
<td>4.17</td>
</tr>
</tbody>
</table>
The Effect of Baseline on Depth Estimation

2. Match
   For each pixel \((i,j)\) in Base
   For each pixel \((k,l)\) in epipolar line of \((i,j)\) in \(X\)
   Compute SSD for \((i,j)\) and \((k,l)\) in \(X\) and Base
   \[ SSD_{ijkl}(i,j) = \sum ((x_{ij} + \frac{u}{2} - x_{kl})^2 + \frac{v}{2} - v_{kl})^2 \]
   \[ -\text{Base}(x_{kl})^2 \]
   where \( \mathbf{e} = (u, v) = \text{unit vector in epipolar line direction in } X \)
   \[ \mathbf{e}^T = \frac{\mathbf{Z}(x_{kl})}{\text{width of a pixel}} \]
   \[ \mathbf{Z}(x_{kl}) = \frac{x_{kl}}{2} \]

Algorithm

1. Edge Enhancement & Noise Suppression
   \[ \nabla^2 I \]
   Implemented in hardware
   \[ 3 \times 3 \text{ oriented Gaussian } \]
   \[ \approx \frac{25 \times 25}{I} \text{ filter} \]

2. Match and Combine
   Given \(n\) not cameras, where one is called \(Base\) and other called \(Imagery\), \(i\)
   Use \(n\) stereo pairs \(\{Base, \_\}\)

2.1 Rectify
   Rectify each imagery image
   with \(Base\) by warping and resampling

2.3 Match
   For each pixel \((i,j)\) in Base
   For each pixel \((k,l)\) in epipolar line of \((i,j)\) in \(X\)
   Compute SSD for \((i,j)\) and \((k,l)\) in \(X\) and Base
   \[ SSD_{ijkl}(i,j) = \sum ((x_{ij} + \frac{u}{2} - x_{kl})^2 + \frac{v}{2} - v_{kl})^2 \]
   \[ -\text{Base}(x_{kl})^2 \]
   where \( \mathbf{e} = (u, v) = \text{unit vector in epipolar line direction in } X \)
   \[ \mathbf{e}^T = \frac{\mathbf{Z}(x_{kl})}{\text{width of a pixel}} \]
   \[ \mathbf{Z}(x_{kl}) = \frac{x_{kl}}{2} \]
3. Estimate Depth Maps

Find value of \( z \) that minimizes \( \text{SSD} \): fit quadratic function to data points and interpolate to estimate \( z \).

Depth \( z = \frac{f}{Z} \) at each pixel.

- Camera Configurations Used
  - "Camera"
  - "Configurations Used"
- Base
- "Base"
- "256 x 240 images"
- "3D frames per second"
- "disparity range 60 pixels"

The CMU Video-Rate Stereo Machine

Video-Rate Stereo Machine

Stereo vision and multi-baseline method

Stereo ranging, which uses correspondence between sets of two or more images for depth measurement, has many advantages. It is passive and it does not emit any radio or light energy. With appropriate imaging geometry, optics, and high-resolution cameras, stereo can produce a dense, precise range image of even distant scenes. Our video-rate stereo machine is based on a new stereo technique which has been developed and tested at CMU over years. It uses multiple images obtained by multiple cameras to produce different baselines in stereo and in directions. The multi-baseline stereo method takes advantage of the redundancy contained in multi-baseline pairs, resulting in a straightforward algorithm which is appropriate for hardware implementation.
Real-Time Stereo

- Used for robot navigation (and other tasks)
  - Several software-based real-time stereo techniques have been developed (most based on simple discrete search)

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)

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

Li Zhang’s one-shot stereo

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

Digital Michelangelo Project
Portable 3D Laser Scanners

Minolta Vivid 910 can scan 300,000 points in 2.5 sec