3D Sensing

- 3D Shape from X
- Perspective Geometry
- Camera Model
- Camera Calibration
- General Stereo Triangulation
- 3D Reconstruction
3D Shape from X

- shading
- silhouette
- texture

- stereo
- light striping
- motion

mainly research

used in practice
Perspective Imaging Model: 1D

This is the axis of the real image plane.

O is the center of projection.

This is the axis of the front image plane, which we use.

\[
\frac{x_i}{f} = \frac{x_c}{z_c}
\]
Perspective in 2D (Simplified)

3D object point
\[ P = (x_c, y_c, z_c) = (x_w, y_w, z_w) \]

Optical axis

Ray

Camera coordinates equal world coordinates.

\[ \frac{x_i}{f} = \frac{x_c}{z_c} \]
\[ x_i = (f/z_c)x_c \]
\[ \frac{y_i}{f} = \frac{y_c}{z_c} \]
\[ y_i = (f/z_c)y_c \]
3D from Stereo

3D point

left image

right image

disparity: the difference in image location of the same 3D point when projected under perspective to two different cameras.

\[ d = x_{\text{left}} - x_{\text{right}} \]
Depth Perception from Stereo
Simple Model: Parallel Optic Axes

\[ \frac{z}{f} = \frac{x}{x_l} \]
\[ \frac{z}{f} = \frac{x-b}{x_r} \]
\[ \frac{z}{f} = \frac{y}{y_l} = \frac{y}{y_r} \]

y-axis is perpendicular to the page.
Resultant Depth Calculation

For stereo cameras with parallel optical axes, focal length $f$, baseline $b$, corresponding image points $(x_l, y_l)$ and $(x_r, y_r)$ with disparity $d$:

$$ z = \frac{f \cdot b}{x_l - x_r} = \frac{f \cdot b}{d} $$

$$ x = \frac{x_l \cdot z}{f} \quad \text{or} \quad b + \frac{x_r \cdot z}{f} $$

$$ y = \frac{y_l \cdot z}{f} \quad \text{or} \quad \frac{y_r \cdot z}{f} $$

This method of determining depth from disparity is called **triangulation**.
Finding Correspondences

• If the correspondence is correct, triangulation works **VERY** well.

• But correspondence finding is not perfectly solved. *(What methods have we studied?)*

• For some very specific applications, it can be solved for those specific kind of images, e.g. windshield of a car.
3 Main Matching Methods

1. Cross correlation using small windows.

2. Symbolic feature matching, usually using segments/corners.

3. Use the newer interest operators, ie. SIFT.
Epipolar Geometry Constraint: 1. Normal Pair of Images

The epipolar plane cuts through the image plane(s) forming 2 epipolar lines.

The match for $P_1$ (or $P_2$) in the other image, must lie on the same epipolar line.
Epipolar Geometry: General Case
1. Epipolar Constraint: Matching points lie on corresponding epipolar lines.

2. Ordering Constraint: Usually in the same order across the lines.
Structured Light

3D data can also be derived using

• a single camera

• a light source that can produce stripe(s) on the 3D object
Structured Light
3D Computation

3D data can also be derived using

- a single camera

- a light source that can produce stripe(s) on the 3D object

\[
\begin{align*}
[b] &= \begin{bmatrix} x & y & z \end{bmatrix} = \begin{bmatrix} x' & y' & f \end{bmatrix} \\
3D &= f \cot \theta - x' \\
image &= \]
Depth from Multiple Light Stripes

What are these objects?
Our (former) System
4-camera light-striping stereo
Camera Model: Recall there are 5 Different Frames of Reference

- Object
- World
- Camera
- Real Image
- Pixel Image
The Camera Model

How do we get an image point IP from a world point P?

\[
\begin{pmatrix}
s & IP_r \\
s & IP_c \\
s & s
\end{pmatrix}
= \begin{pmatrix}
c_{11} & c_{12} & c_{13} & c_{14} \\
c_{21} & c_{22} & c_{23} & c_{24} \\
c_{31} & c_{32} & c_{33} & 1
\end{pmatrix}
\begin{pmatrix}
P_x \\
P_y \\
P_z \\
1
\end{pmatrix}
\]

What’s in C?

image point

camera matrix C

world point
The camera model handles the **rigid body** transformation from world coordinates to camera coordinates plus the **perspective** transformation to image coordinates.

1. \[ CP = T \cdot R \cdot WP \]
2. \[ FP = \pi(f) \cdot CP \]

Why is there not a scale factor here?

\[
\begin{pmatrix}
  s \cdot F_{P_x} \\
  s \cdot F_{P_y} \\
  s \cdot F_{P_z} \\
  s
\end{pmatrix} =
\begin{pmatrix}
  1 & 0 & 0 & 0 \\
  0 & 1 & 0 & 0 \\
  0 & 0 & 1 & 0 \\
  0 & 0 & 1/f & 0
\end{pmatrix}
\begin{pmatrix}
  C_{P_x} \\
  C_{P_y} \\
  C_{P_z} \\
  1
\end{pmatrix}
\]

3D point in camera coordinates
Camera Calibration

- In order to work in 3D, we need to know the parameters of the particular camera setup.

- Solving for the camera parameters is called calibration.

- **intrinsic** parameters are of the camera device

- **extrinsic** parameters are where the camera sits in the world
Intrinsic Parameters

- principal point \((u_0,v_0)\)
- scale factors \((d_x,d_y)\)
- aspect ratio distortion factor \(\gamma\)
- focal length \(f\)
- lens distortion factor \(\kappa\)
  (models radial lens distortion)
Extrinsic Parameters

• translation parameters
  \[ t = [t_x \ t_y \ t_z] \]

• rotation matrix

\[
R = \begin{bmatrix}
  r_{11} & r_{12} & r_{13} & 0 \\
  r_{21} & r_{22} & r_{23} & 0 \\
  r_{31} & r_{32} & r_{33} & 0 \\
  0 & 0 & 0 & 1
\end{bmatrix}
\]

Are there really nine parameters?
Calibration Object

The idea is to snap images at different depths and get a lot of 2D-3D point correspondences.
The Tsai Procedure

- The Tsai procedure was developed by Roger Tsai at IBM Research and is most widely used.

- Several images are taken of the calibration object yielding point correspondences at different distances.

- Tsai’s algorithm requires $n > 5$ correspondences

  $$\{(x_i, y_i, z_i), (u_i, v_i)\} \mid i = 1, \ldots, n$$

between (real) image points and 3D points.
In this* version of Tsai’s algorithm,

- The real-valued \((u,v)\) are computed from their pixel positions \((r,c)\):

\[
u = \gamma \, d_x \, (c-u_0) \quad v = -d_y \, (r - v_0)
\]

where

- \((u_0,v_0)\) is the center of the image
- \(d_x\) and \(d_y\) are the center-to-center (real) distances between pixels and come from the camera’s specs
- \(\gamma\) is a scale factor learned from previous trials

* This version is for single-plane calibration.
Tsai’s Procedure

1. Given the n point correspondences \(((x_i,y_i,z_i), (u_i,v_i))\)

   Compute matrix A with rows \(a_i\)

   \[ a_i = (v_i \cdot x_i, v_i \cdot y_i, -u_i \cdot x_i, -u_i \cdot v_i, v_i) \]

These are known quantities which will be used to solve for intermediate values, which will then be used to solve for the parameters sought.
2. The vector of unknowns is $\mathbf{\mu} = (\mu_1, \mu_2, \mu_3, \mu_4, \mu_5)$:

$$
\begin{align*}
\mu_1 &= r_{11}/t_y \\
\mu_2 &= r_{12}/t_y \\
\mu_3 &= r_{21}/t_y \\
\mu_4 &= r_{22}/t_y \\
\mu_5 &= t_x/t_y 
\end{align*}
$$

where the $r$’s and $t$’s are unknown rotation and translation parameters.

3. Let vector $\mathbf{b} = (u_1, u_2, \ldots, u_n)$ contain the $u$ image coordinates.

4. Solve the system of linear equations

$$
A \mathbf{\mu} = \mathbf{b}
$$

for unknown parameter vector $\mathbf{\mu}$. 
Use $\mu$ to solve for $t_y$, $t_x$, and 4 rotation parameters

5. Let $U = \mu_1^2 + \mu_2^2 + \mu_3^2 + \mu_4^2$. Use $U$ to calculate $t_y^2$.

$$t_y^2 = \begin{cases} 
\frac{U - [U^2 - 4(\mu_1\mu_4 - \mu_2\mu_3)^2]^{1/2}}{2(\mu_1\mu_4 - \mu_2\mu_3)^2} & \text{if } (\mu_1\mu_4 - \mu_2\mu_3) \neq 0 \\
\frac{1}{\mu_1^2 + \mu_2^2} & \text{if } (\mu_1^2 + \mu_2^2) \neq 0 \\
\frac{1}{\mu_3^2 + \mu_4^2} & \text{if } (\mu_3^2 + \mu_4^2) \neq 0 
\end{cases}$$
6. Try the positive square root $t_y = (t^2)^{1/2}$ and use it to compute translation and rotation parameters.

\[
\begin{align*}
    r_{11} &= \mu_1 t_y \\
    r_{12} &= \mu_2 t_y \\
    r_{21} &= \mu_3 t_y \\
    r_{22} &= \mu_4 t_y \\
    t_x &= \mu_5 t_y
\end{align*}
\]

Now we know 2 translation parameters and 4 rotation parameters. Except...
Determine true sign of $t_y$ and compute remaining rotation parameters.

7. Select an object point $P$ whose image coordinates $(u,v)$ are far from the image center.

8. Use $P$’s coordinates and the translation and rotation parameters so far to estimate the image point that corresponds to $P$.

If its coordinates have the same signs as $(u,v)$, then keep $t_y$, else negate it.

9. Use the first 4 rotation parameters to calculate the remaining 5.
Calculating the remaining 5 rotation parameters:

\[
\begin{align*}
  r_{13} &= (1 - r_{11}^2 - r_{12}^2)^{1/2} \\
  r_{23} &= (1 - r_{21}^2 - r_{22}^2)^{1/2} \\
  r_{31} &= \frac{1 - r_{11}^2 - r_{12}r_{21}}{r_{13}} \\
  r_{32} &= \frac{1 - r_{21}r_{12} - r_{22}^2}{r_{23}} \\
  r_{33} &= (1 - r_{31}r_{13} - r_{32}r_{23})^{1/2}
\end{align*}
\]
Solve another linear system.

10. We have $t_x$ and $t_y$ and the 9 rotation parameters. Next step is to find $t_z$ and $f$.

Form a matrix $A'$ whose rows are:

$$a_i' = (r_{21}x_i + r_{22}y_i + t_y, v_i)$$

and a vector $b'$ whose rows are:

$$b_i' = (r_{31}x_i + r_{32}y_i) * v_i$$

11. Solve $A'*v = b'$ for $v = (f, t_z)$. 
12. If $f$ is negative, change signs (see text).

13. Compute the lens distortion factor $\kappa$ and improve the estimates for $f$ and $t_z$ by solving a nonlinear system of equations by a nonlinear regression.

14. All parameters have been computed.

Use them in 3D data acquisition systems.
We use them for general stereo.
For a correspondence \((r_1, c_1)\) in image 1 to \((r_2, c_2)\) in image 2:

1. Both cameras were calibrated. Both camera matrices are then known. From the two camera equations B and C we get

4 linear equations in 3 unknowns.

\[
\begin{align*}
    r_1 &= (b_{11} - b_{31} r_1)x + (b_{12} - b_{32} r_1)y + (b_{13} - b_{33} r_1)z \\
    c_1 &= (b_{21} - b_{31} c_1)x + (b_{22} - b_{32} c_1)y + (b_{23} - b_{33} c_1)z \\
    r_2 &= (c_{11} - c_{31} r_2)x + (c_{12} - c_{32} r_2)y + (c_{13} - c_{33} r_2)z \\
    c_2 &= (c_{21} - c_{31} c_2)x + (c_{22} - c_{32} c_2)y + (c_{23} - c_{33} c_2)z
\end{align*}
\]

Direct solution uses 3 equations, won’t give reliable results.
Solve by computing the closest approach of the two skew rays.

\[ V = (P_1 + a_1 u_1) - (Q_1 + a_2 u_2) \]

\[(P_1 + a_1 u_1) - (Q_1 + a_2 u_2) \cdot u_1 = 0\]
\[(P_1 + a_1 u_1) - (Q_1 + a_2 u_2) \cdot u_2 = 0\]

If the rays intersected perfectly in 3D, the intersection would be P. Instead, we solve for the shortest line segment connecting the two rays and let P be its midpoint.
Surface Modeling and Display from Range and Color Data

Kari Pulli         UW
Michael Cohen      MSR
Tom Duchamp        UW
Hugues Hoppe       MSR
John McDonald      UW
Linda Shapiro      UW
Werner Stuetzle    UW

UW = University of Washington
    Seattle, WA USA
MSR = Microsoft Research
     Redmond, WA USA
Introduction

Goal

- develop robust algorithms for constructing 3D models from range & color data
- use those models to produce realistic renderings of the scanned objects
Surface Reconstruction

Step 1: Data acquisition
Obtain range data that covers the object. Filter, remove background.

Step 2: Registration
Register the range maps into a common coordinate system.

Step 3: Integration
Integrate the registered range data into a single surface representation.

Step 4: Optimization
Fit the surface more accurately to the data, simplify the representation.
Problem

Noisy registered data

Signed distance fn & marching cubes

Hierarchical & directional space carving
Carve space in cubes

Label cubes

- Project cube to image plane (hexagon)
- Test against data in the hexagon
Several views

Processing order:
FOR EACH cube
   FOR EACH view

Rules:
any view thinks cube’s out
   => it’s out
every view thinks cube’s in
   => it’s in
else
   => it’s at boundary
Hierarchical space carving

- Big cubes $\Rightarrow$ fast, poor results
- Small cubes $\Rightarrow$ slow, more accurate results
- Combination = octrees

RULES:
- cube's out $\Rightarrow$ done
- cube's in $\Rightarrow$ done
- else $\Rightarrow$ recurse
Hierarchical space carving

- Big cubes => fast, poor results
- Small cubes => slow, more accurate results
- Combination = octrees

RULES:
- cube's out => done
- cube's in => done
- else => recurse
The rest of the chair
Same for a husky pup
Optimizing the dog mesh

Registered points

Initial mesh

Optimized mesh
View dependent texturing
Our viewer