Reconstruction
Reconstruction

• The forward process:
  • Given
    • 3D shapes
    • Their locations+orientations
    • Their material+paint
    • Light directions+intensity
    • The Camera parameters
  • Produce the image

• Reconstruction: *Reverse* this process

• Next two/three classes: reconstructing geometry
Final perspective projection

Camera extrinsics: where your camera is relative to the world. Changes if you move the camera

\[ \mathbf{x}_{img} \equiv \begin{bmatrix} K & R & t \end{bmatrix} \mathbf{x}_w \]

Camera intrinsics: how your camera handles pixel. Changes if you change your camera

\[ \mathbf{x}_{img} \equiv P\mathbf{x}_w \]
Final perspective projection

\[ \tilde{x}_{img} \equiv K \begin{bmatrix} R & t \end{bmatrix} \tilde{x}_w \]

Camera parameters

\[ \tilde{x}_{img} \equiv P \tilde{x}_w \]
Camera calibration

• Goal: find the parameters of the camera

\[ \vec{x}_{img} \equiv K \begin{bmatrix} R & t \end{bmatrix} \vec{x}_w \]

• Why?
  • Tells you where the camera is relative to the world/particular objects
  • Equivalently, tells you where objects are relative to the camera
  • Can allow you to "render" new objects into the scene
Camera calibration

$$\vec{x}_{img} \equiv P\vec{x}_w$$

• Need to estimate $P$
• How many parameters does $P$ have?
  • Size of $P$ : $3 \times 4$
  • But: $\lambda P\vec{x}_w \equiv P\vec{x}_w$
  • $P$ can only be known \textit{upto a scale}
  • $3*4 - 1 = 11 \text{ parameters}$
Camera calibration

\[ \vec{x}_{img} \equiv P \vec{x}_w \]

- Suppose we know that \((X,Y,Z)\) in the world projects to \((x,y)\) in the image.
- How many equations does this provide?

\[
\begin{bmatrix}
  x \\
  y \\
  1
\end{bmatrix} \equiv P
\begin{bmatrix}
  X \\
  Y \\
  Z \\
  1
\end{bmatrix}
\]

Need to convert equivalence into equality.
Camera calibration

\[ \vec{x}_{img} \equiv P \vec{x}_w \]

• Suppose we know that \((X,Y,Z)\) in the world projects to \((x,y)\) in the image.

• How many equations does this provide?

Note: \(\lambda\) is unknown

\[
\begin{bmatrix}
\lambda x \\
\lambda y \\
\lambda \\
1
\end{bmatrix}
= P
\begin{bmatrix}
X \\
Y \\
Z \\
1
\end{bmatrix}
\]
Camera calibration

\[ \mathbf{x}_{img} \equiv P\mathbf{x}_w \]

• Suppose we know that \((X,Y,Z)\) in the world projects to \((x,y)\) in the image.

• How many equations does this provide?

\[
\begin{bmatrix}
\lambda x \\
\lambda y \\
\lambda 
\end{bmatrix} = 
\begin{bmatrix}
P_{11} & P_{12} & P_{13} & P_{14} \\
P_{21} & P_{22} & P_{23} & P_{24} \\
P_{31} & P_{32} & P_{33} & P_{34} \\
1 & 1 & 1 & 1 
\end{bmatrix}
\begin{bmatrix}
X \\
Y \\
Z \\
1 
\end{bmatrix}
\]
Camera calibration

\[ \vec{x}_{img} \equiv P\vec{x}_w \]

• Suppose we know that (X,Y,Z) in the world projects to (x,y) in the image.

• How many equations does this provide?

\[
\lambda x = P_{11}X + P_{12}Y + P_{13}Z + P_{14} \\
\lambda y = P_{21}X + P_{22}Y + P_{23}Z + P_{24} \\
\lambda = P_{31}X + P_{32}Y + P_{33}Z + P_{34}
\]
Camera calibration

\[ \vec{x}_{img} \equiv P \vec{x}_w \]

• Suppose we know that (X,Y,Z) in the world projects to (x,y) in the image.

• How many equations does this provide?

\[
(P_{31}X + P_{32}Y + P_{33}Z + P_{34})x = P_{11}X + P_{12}Y + P_{13}Z + P_{14}
\]

\[
(P_{31}X + P_{32}Y + P_{33}Z + P_{34})y = P_{21}X + P_{22}Y + P_{23}Z + P_{24}
\]

• 2 equations!

• Are the equations linear in the parameters?

• How many equations do we need?
Camera calibration

\[(P_{31}X + P_{32}Y + P_{33}Z + P_{34})x = P_{11}X + P_{12}Y + P_{13}Z + P_{14}\]

\[XxP_{31} + YxP_{32} + ZxP_{33} + xP_{34} - XP_{11} - YP_{12} - ZP_{13} - P_{14} = 0\]

• In matrix vector form: \(A\mathbf{p} = \mathbf{0}\)
• 6 points give 12 equations, 12 variables to solve for
• But can only solve up to scale
Camera calibration

- In matrix vector form: $A\mathbf{p} = 0$
- We want non-trivial solutions
- If $\mathbf{p}$ is a solution, $\alpha \mathbf{p}$ is a solution too
- Let’s just search for a solution with unit norm

\[
A\mathbf{p} = 0
\quad \text{s.t}
\quad \|\mathbf{p}\| = 1
\]
Camera calibration

• In matrix vector form: \( A p = 0 \)
• We want non-trivial solutions
• If \( p \) is a solution, \( \alpha p \) is a solution too
• Alternative form to deal with noisy inputs
  \[
  \min_p ||Ap||^2 \equiv \min_p p^T A^T A p
  \]
  \[
  \text{s.t. } ||p|| = 1
  \]
• How do you solve this?
  • *Eigenvector of \( A^T A \) with smallest eigenvalue!*
Camera calibration

• We need 6 world points for which we know image locations
• Would any 6 points work?
  • What if all 6 points are the same?
• Need at least 6 non-coplanar points!
Camera calibration

\[
\begin{align*}
\mathbf{\tilde{x}}_{img} & \equiv P\mathbf{\tilde{x}}_w \\
\mathbf{\tilde{x}}_{img} & \equiv K \begin{bmatrix} R & t \end{bmatrix} \mathbf{\tilde{x}}_w
\end{align*}
\]

• How do we get K, R and t from P?
• Need to make some assumptions about K
• What if K is upper triangular?

\[
K = \begin{bmatrix}
s_x & \alpha & t_u \\
0 & s_y & t_v \\
0 & 0 & 1
\end{bmatrix}
\]

Added skew if image x and y axes are not perpendicular
Camera calibration

- How do we get K, R and t from P?
- Need to make some assumptions about K
- What if K is upper triangular?

\[ K = \begin{bmatrix} s_x & \alpha & t_u \\ 0 & s_y & t_v \\ 0 & 0 & 1 \end{bmatrix} \]

- P = K [ R t]
- First 3 x 3 matrix of P is KR
- “RQ” decomposition: decomposes an n x n matrix into product of upper triangular and rotation matrix
Camera calibration

• How do we get K, R and t from P?
• Need to make some assumptions about K
• What if K is upper triangular?
• \[ P = K [ R \ t] \]
• First 3 x 3 matrix of P is KR
• “RQ” decomposition: decomposes an n x n matrix into product of upper triangular and rotation matrix
• \[ t = K^{-1}P[:,2] \leftarrow \text{last column of P} \]
Camera calibration and pose estimation

- Where camera is relative to car is equivalent to where car is relative to camera
Reconstructing world points given camera

\[ \vec{x}_{img} \equiv K \begin{bmatrix} R & t \end{bmatrix} \vec{x}_w \]

Can we recover this from just a single equation?

\[ \vec{x}_{img} \equiv P\vec{x}_w \]
Ambiguity

• A pixel corresponds to an entire ray
  • 2 linear equations in 3D space
• Need additional constraints!
Triangulation

- A pixel corresponds to an entire ray
  - 2 linear equations in 3D space
- If we have corresponding pixel from another view, can intersect rays!
  - 4 equations
Binocular stereo

• Single

• If we know where cameras are, we can shoot rays from corresponding pixels and intersect
Triangulation

• Suppose we have two cameras
  • Calibrated: parameters known
• And a pair of corresponding pixels
• Find 3D location of point!
Triangulation

• Suppose we have two cameras
  • Calibrated: parameters known
• And a pair of corresponding pixels
• Find 3D location of point!

$P^{(1)}$

$P^{(2)}$
Triangulation

\[
\begin{bmatrix}
x_1 \\
y_1 \\
1
\end{bmatrix} \leftarrow \vec{x}_{\text{img}}^{(1)} \equiv P^{(1)} \vec{x}_w
\]

\[
\begin{bmatrix}
x_2 \\
y_2 \\
1
\end{bmatrix} \leftarrow \vec{x}_{\text{img}}^{(2)} \equiv P^{(2)} \vec{x}_w
\]

\[
\begin{bmatrix}
X \\
Y \\
Z \\
1
\end{bmatrix}
\]
Triangulation

\[ \mathbf{x}_{img}^{(1)} \equiv P^{(1)} \mathbf{x}_w \]

\[ \lambda x_1 = P_{11}^{(1)} X + P_{12}^{(1)} Y + P_{13}^{(1)} Z + P_{14}^{(1)} \]

\[ \lambda y_1 = P_{21}^{(1)} X + P_{22}^{(1)} Y + P_{23}^{(1)} Z + P_{24}^{(1)} \]

\[ \lambda = P_{31}^{(1)} X + P_{32}^{(1)} Y + P_{33}^{(1)} Z + P_{34}^{(1)} \]

\[ (P_{31}^{(1)} X + P_{32}^{(1)} Y + P_{33}^{(1)} Z + P_{34}^{(1)}) x_1 = P_{11}^{(1)} X + P_{12}^{(1)} Y + P_{13}^{(1)} Z + P_{14}^{(1)} \]

\[ X (P_{31}^{(1)} x_1 - P_{11}^{(1)}) + Y (P_{32}^{(1)} x_1 - P_{12}^{(1)}) + Z (P_{33}^{(1)} x_1 - P_{13}^{(1)}) + (P_{34}^{(1)} x_1 - P_{14}^{(1)}) = 0 \]
Triangulation

\[ \mathbf{x}_{img}^{(1)} \equiv P^{(1)} \mathbf{x}_w \]

\[
X(P_{31}^{(1)} x_1 - P_{11}^{(1)}) + Y(P_{32}^{(1)} x_1 - P_{12}^{(1)}) + Z(P_{33}^{(1)} x_1 - P_{13}^{(1)}) + (P_{34}^{(1)} x_1 - P_{14}^{(1)}) = 0
\]

\[
X(P_{31}^{(1)} y_1 - P_{21}^{(1)}) + Y(P_{32}^{(1)} y_1 - P_{22}^{(1)}) + Z(P_{33}^{(1)} y_1 - P_{23}^{(1)}) + (P_{34}^{(1)} y_1 - P_{24}^{(1)}) = 0
\]

- 1 image gives 2 equations
- Need 2 images!
- Solve linear equations to get 3D point location
Linear vs non-linear optimization

\[ \lambda x_1 = P_{11}^{(1)} X + P_{12}^{(1)} Y + P_{13}^{(1)} Z + P_{14}^{(1)} \]

\[ \lambda y_1 = P_{21}^{(1)} X + P_{22}^{(1)} Y + P_{23}^{(1)} Z + P_{24}^{(1)} \]

\[ \lambda = P_{31}^{(1)} X + P_{32}^{(1)} Y + P_{33}^{(1)} Z + P_{34}^{(1)} \]

\[ x_1 = \frac{P_{11}^{(1)} X + P_{12}^{(1)} Y + P_{13}^{(1)} Z + P_{14}^{(1)}}{P_{31}^{(1)} X + P_{32}^{(1)} Y + P_{33}^{(1)} Z + P_{34}^{(1)}} \]

\[ y_1 = \frac{P_{21}^{(1)} X + P_{22}^{(1)} Y + P_{23}^{(1)} Z + P_{24}^{(1)}}{P_{31}^{(1)} X + P_{32}^{(1)} Y + P_{33}^{(1)} Z + P_{34}^{(1)}} \]
Linear vs non-linear optimization

\[ x_1 = \frac{P_{11}^{(1)} X + P_{12}^{(1)} Y + P_{13}^{(1)} Z + P_{14}^{(1)}}{P_{31}^{(1)} X + P_{32}^{(1)} Y + P_{33}^{(1)} Z + P_{34}^{(1)}} \]

\[ y_1 = \frac{P_{21}^{(1)} X + P_{22}^{(1)} Y + P_{23}^{(1)} Z + P_{24}^{(1)}}{P_{31}^{(1)} X + P_{32}^{(1)} Y + P_{33}^{(1)} Z + P_{34}^{(1)}} \]

\[ \left( x_1 - \frac{P_{11}^{(1)} X + P_{12}^{(1)} Y + P_{13}^{(1)} Z + P_{14}^{(1)}}{P_{31}^{(1)} X + P_{32}^{(1)} Y + P_{33}^{(1)} Z + P_{34}^{(1)}} \right)^2 + \left( y_1 - \frac{P_{21}^{(1)} X + P_{22}^{(1)} Y + P_{23}^{(1)} Z + P_{24}^{(1)}}{P_{31}^{(1)} X + P_{32}^{(1)} Y + P_{33}^{(1)} Z + P_{34}^{(1)}} \right)^2 \]

Reprojection error
Linear vs non-linear optimization

Reprojection error is the squared error between the true image coordinates of a point and the projected coordinates of hypothesized 3D point.

Actual error we care about.

Minimize total sum of reprojection error across all images.

Non-linear optimization.
Binocular stereo

• Given two calibrated cameras
  • Find pairs of corresponding pixels
  • Use corresponding image locations to set up equations on world coordinates
  • Solve!
Binocular stereo

- General case: cameras can be arbitrary locations and orientations
Binocular stereo

- Special case: cameras are parallel to each other and translated along X axis
Stereo with *rectified cameras*

- Special case: cameras are parallel to each other and translated along X axis
Stereo head

Kinect / depth cameras
Stereo with “rectified cameras”
Perspective projection in rectified cameras

• Without loss of generality, assume origin is at pinhole of 1st camera

\[
\vec{x}_{img}^{(1)} \equiv \begin{bmatrix} I & 0 \end{bmatrix} \vec{x}_w
\]

\[
\vec{x}_{img}^{(2)} \equiv \begin{bmatrix} I & t \end{bmatrix} \vec{x}_w
\]

\[
t = \begin{bmatrix} t_x \\ 0 \\ 0 \end{bmatrix}
\]
Perspective projection in rectified cameras

• Without loss of generality, assume origin is at pinhole of 1st camera

\[ \tilde{x}^{(1)}_{img} \equiv \begin{bmatrix} I & 0 \end{bmatrix} \tilde{x}_w \]

\[ \tilde{x}^{(2)}_{img} \equiv \begin{bmatrix} I & t \end{bmatrix} \tilde{x}_w \]

\[ t = \begin{bmatrix} t_x \\ 0 \\ 0 \end{bmatrix} \tilde{x}_w = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = [x_w] \]
Perspective projection in rectified cameras

• Without loss of generality, assume origin is at pinhole of 1st camera

\[
\mathbf{x}_{img}^{(1)} \equiv \begin{bmatrix} I & 0 \end{bmatrix} \begin{bmatrix} x_w \\ 1 \end{bmatrix} = x_w = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} x_w + t_x \\ Y \\ Z \end{bmatrix}
\]

\[
\mathbf{x}_{img}^{(2)} \equiv \begin{bmatrix} I & t \end{bmatrix} \begin{bmatrix} x_w \\ 1 \end{bmatrix} = x_w + t = \begin{bmatrix} X + t_x \\ Y \\ Z \end{bmatrix}
\]
Perspective projection in rectified cameras

• Without loss of generality, assume origin is at pinhole of 1st camera

\[
\vec{x}_{img}^{(1)} \equiv \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}
\]

\[
\vec{x}_{img}^{(2)} \equiv \begin{bmatrix} X + t_x \\ Y \\ Z \end{bmatrix}
\]
Perspective projection in rectified cameras

- Without loss of generality, assume origin is at pinhole of 1st camera

\[
\begin{bmatrix}
    x_1 \\
    y_1 \\
    1
\end{bmatrix}
\equiv
\begin{bmatrix}
    X \\
    Y \\
    Z
\end{bmatrix}
\]

\[
\begin{bmatrix}
    x_2 \\
    y_2 \\
    1
\end{bmatrix}
\equiv
\begin{bmatrix}
    X + t_x \\
    Y \\
    Z
\end{bmatrix}
\]
Perspective projection in rectified cameras

• Without loss of generality, assume origin is at pinhole of 1st camera

\[
\begin{pmatrix}
    x_1 \\
    y_1 \\
    1
\end{pmatrix}
\equiv
\begin{pmatrix}
    X \\
    Y \\
    Z
\end{pmatrix}
\]

\[
\begin{pmatrix}
    x_2 \\
    y_2 \\
    1
\end{pmatrix}
\equiv
\begin{pmatrix}
    X + t_x \\
    Y \\
    Z
\end{pmatrix}
\]
Perspective projection in rectified cameras

• Without loss of generality, assume origin is at pinhole of 1\textsuperscript{st} camera

\[
\begin{bmatrix}
\lambda x_1 \\
\lambda y_1 \\
\lambda \\
\lambda
\end{bmatrix}
= \begin{bmatrix}
X \\
Y \\
Z \\
X + t_x
\end{bmatrix}
\]

\[
\begin{bmatrix}
\lambda x_2 \\
\lambda y_2 \\
\lambda \\
\lambda
\end{bmatrix}
= \begin{bmatrix}
X \\
Y \\
Z
\end{bmatrix}
\]
Perspective projection in rectified cameras

• Without loss of generality, assume origin is at pinhole of 1\textsuperscript{st} camera

\[
\begin{align*}
x_1 &= \frac{X}{Z} & x_2 &= \frac{X + t_x}{Z} \\
y_1 &= \frac{Y}{Z} & y_2 &= \frac{Y}{Z}
\end{align*}
\]

Y coordinate is the same!
Perspective projection in rectified cameras

- For rectified cameras, correspondence problem is easier
- Only requires searching along a particular row.
Rectifying cameras

• Given two images from two cameras with known $P$, can we rectify them?
  • Can we create new images corresponding to a rectified setup?
Rectifying cameras

• Can we rotate / translate cameras?
  • Do we need to know the 3D structure of the world to do this?
Rotating cameras

\[ \vec{x}_{img} \equiv K \begin{bmatrix} R & t \end{bmatrix} \vec{x}_w \]

- Assume K is identity
- Assume coordinate system at camera pinhole

\[ \vec{x}_{img} \equiv \begin{bmatrix} I & 0 \end{bmatrix} \vec{x}_w \]

\[ \equiv \begin{bmatrix} I & 0 \end{bmatrix} \begin{bmatrix} x_w \\ 1 \end{bmatrix} \]

\[ \equiv x_w \]
Rotating cameras

\[ \hat{x}_{img} \equiv K \begin{bmatrix} R & t \end{bmatrix} \hat{x}_w \]

- Assume K is identity
- Assume coordinate system at camera pinhole

\[ \hat{x}_{img} \equiv \begin{bmatrix} I & 0 \end{bmatrix} \hat{x}_w \]
\[ \equiv \begin{bmatrix} I & 0 \end{bmatrix} \begin{bmatrix} x_w \\ 1 \end{bmatrix} \]
\[ \equiv x_w \]
Rotating cameras

\[ \vec{x}_{img} \equiv \begin{bmatrix} I & 0 \end{bmatrix} \begin{bmatrix} x_w \\ 1 \end{bmatrix} \]

\[ \vec{x}_{img} \equiv x_w \]

• What happens if the camera is rotated?

\[ \vec{x}'_{img} \equiv \begin{bmatrix} R & 0 \end{bmatrix} \begin{bmatrix} x_w \\ 1 \end{bmatrix} \]

\[ \equiv Rx_w \]

\[ \equiv Rx_{\vec{x}_{img}} \]
Rotating cameras

- What happens if the camera is rotated?

- No need to know the 3D structure
Rotating cameras
Rectifying cameras
Rectifying cameras
Rectifying cameras
Rectifying cameras