Image Formation

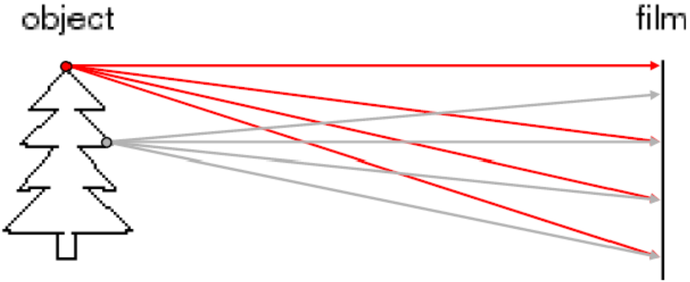
Dr. Randy C. Hoover

Department of Electrical and Computer Engineering
South Dakota School of Mines and Technology
Rapid City, SD 57701, USA

September 21, 2012

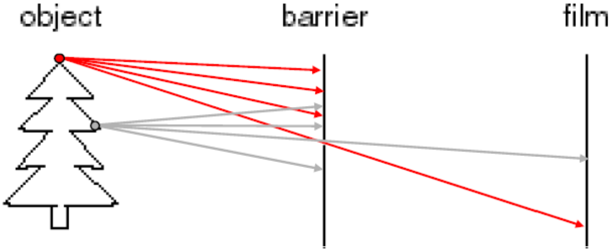
How to build a camera?

- We could always simply lay some film down and "collect" the light!
- What's wrong with this approach?



How to build a camera?

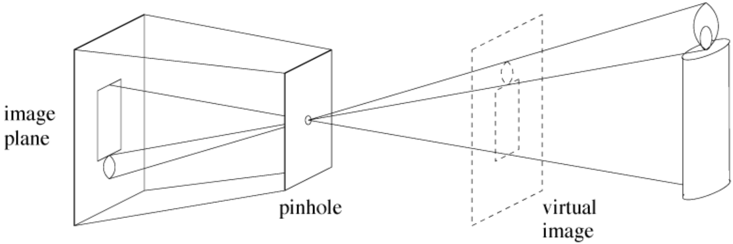
- Use a small hole (commonly referred to as an aperture) to focus the light
  - Barrier blocks excess rays
- Improves the quality by removing blurring effects



- Referred to as a pinhole camera

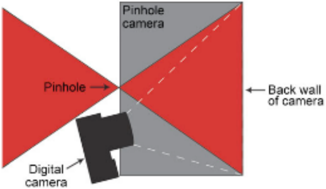
Creating a pinhole camera

- Object appears inverted on the camera plane
  - Common to consider a *virtual image* instead
    - plane lying in front of the pinhole at the same distance as the actual image plane



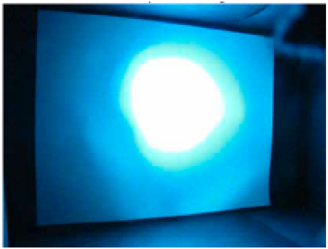
Creating a pinhole camera (Homework 4)

- Create your own pinhole camera!



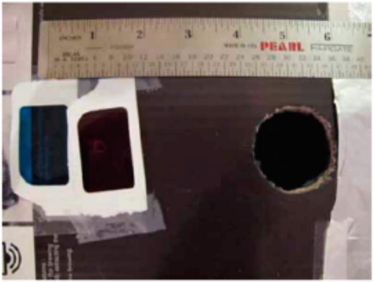
Creating a pinhole camera (Homework 4)

- Should get something like this! (notice the inverted image)



Creating a 3-D pinhole camera

- Can also make a 3-D pinhole!

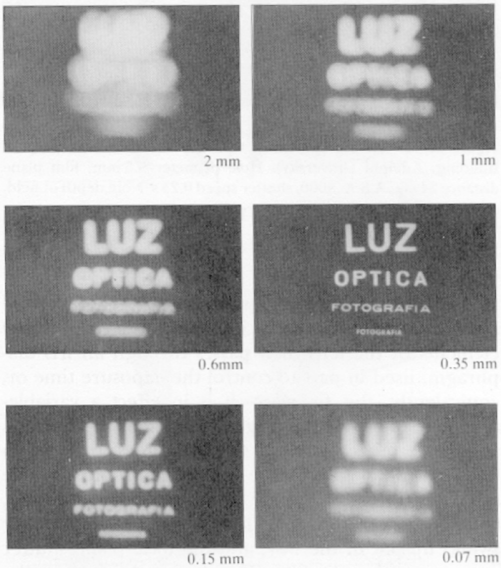


Creating a 3-D pinhole camera

- Image generated is referred to as an anaglyph)

What size do I make my hole?

- Too big $\implies$ blurring
- Too small $\implies$ diffraction

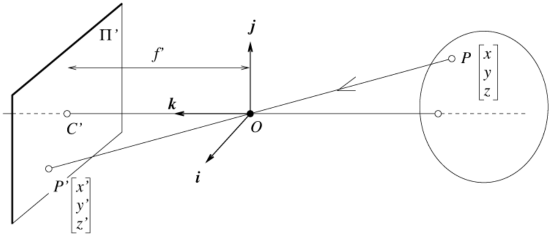


So how exactly do 3-D world points get "projected" onto the image plane?

- Many different models to represent this projection!
  - Orthographic, scaled orthographic, para-perspective, object centered, perspective

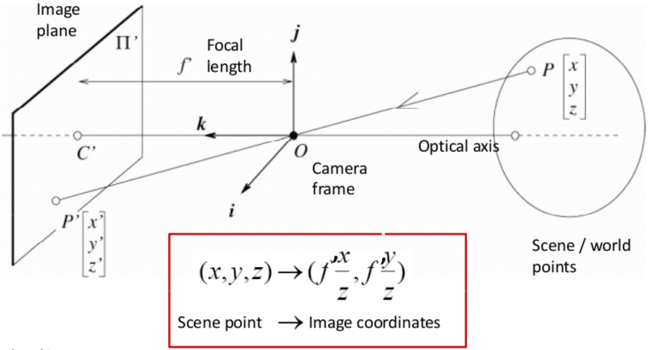
So how exactly do 3-D world points get "projected" onto the image plane?

- Many different models to represent this projection!
  - Orthographic, scaled orthographic, para-perspective, object centered, perspective
- The most common model (and the one we focus on) is the perspective projection model!



Perspective projection

- Given: Some 3-dimensional point in the world, this point is projected onto the image plane



Forsyth and Ponce

Perspective projection

- Where do these equations come from???

Perspective projection

- Where do these equations come from???
- Homogeneous coordinate transformations!

Perspective projection

- Where do these equations come from???
- Homogeneous coordinate transformations!
- Recall (using homogeneous coordinates):

$$(x, y) \Longrightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad \text{and} \quad (x, y, z) \Longrightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY Electrical & Computer Engineering

Perspective projection

- Where do these equations come from???
- Homogeneous coordinate transformations!
- Recall (using homogeneous coordinates):

$$(x, y) \Longrightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad \text{and} \quad (x, y, z) \Longrightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

- Homogeneous coordinates are only unique up to scale! Assign a "dummy" variable w

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Longrightarrow (x/w, y/w) \quad \text{and} \quad \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \Longrightarrow (x/w, y/w, z/w)$$

Perspective projection

- Where do these equations come from???
- Homogeneous coordinate transformations!
- Recall (using homogeneous coordinates):

$$(x, y) \implies \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad \text{and} \quad (x, y, z) \implies \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

- Homogeneous coordinates are only unique up to scale! Assign a "dummy" variable w

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \implies (x/w, y/w) \quad \text{and} \quad \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \implies (x/w, y/w, z/w)$$

- What happens when $w = 0$?

Perspective projection matrix

- Let f' be the camera focal length, then the projection is written as a matrix multiplication using homogeneous coordinates:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f' & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z/f' \end{bmatrix}$$

Perspective projection matrix

- Let f' be the camera focal length, then the projection is written as a matrix multiplication using homogeneous coordinates:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f' & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z/f' \end{bmatrix}$$

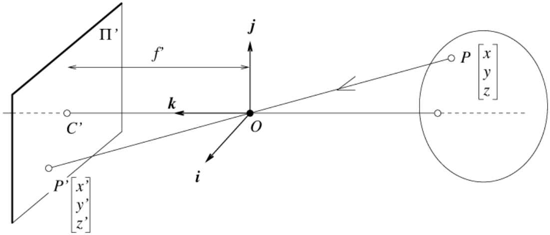
- Need to divide by the last element to convert back to non-homogeneous coordinates

$$\begin{bmatrix} x \\ y \\ z/f' \end{bmatrix} \Longrightarrow (f'\frac{x}{z}, f'\frac{y}{z})$$

Issues???

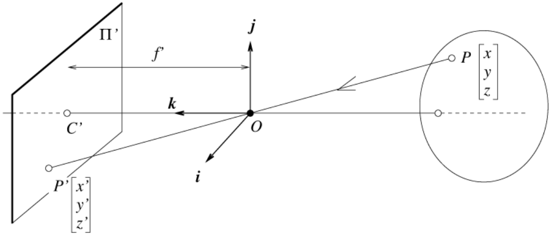
Issues???

- The perspective equations are in terms of the *camera's* local reference frame!



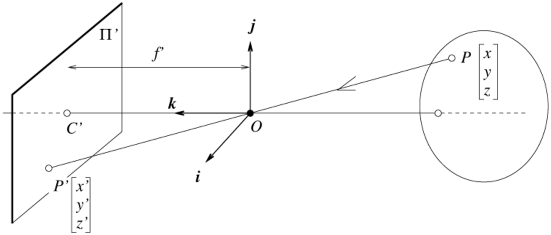
Issues???

- The perspective equations are in terms of the *camera's* local reference frame!
- We need to determine the camera's *intrinsic* and *extrinsic* parameters to calibrate the geometry



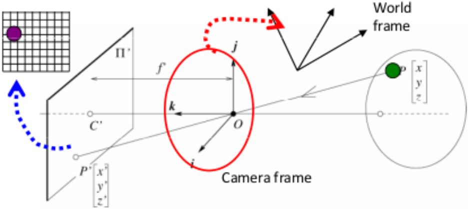
Issues???

- The perspective equations are in terms of the *camera's* local reference frame!
- We need to determine the camera's *intrinsic* and *extrinsic* parameters to calibrate the geometry
  - Need to determine the camera's calibration matrix



Perspective projection and calibration

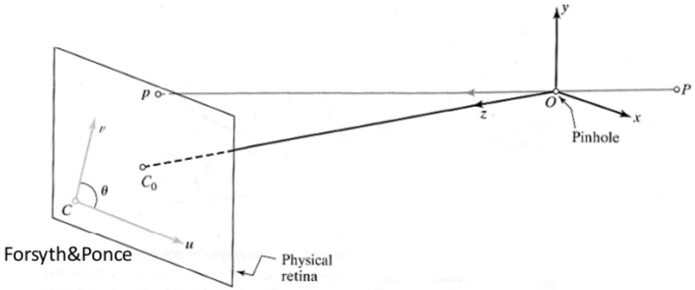
- Take some 3-D point in a world coordinate frame and transform to some pixel coordinates on the image sensor!



Extrinsic:
Camera frame ←→World frame

Intrinsic:
Image coordinates relative to camera ←→ Pixel coordinates

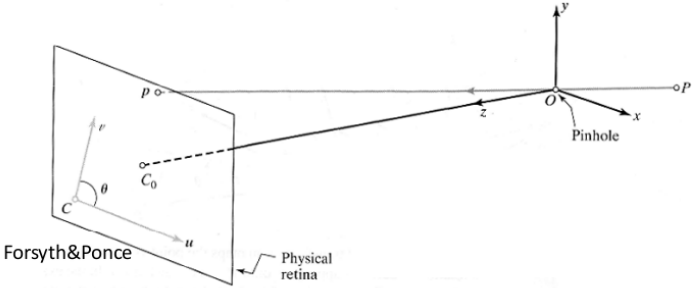
Intrinsic parameters: from idealized world coordinates to pixel values



- Using our standard perspective projection equations (and letting f be the focal length):

$$u = f\frac{x}{z} \quad v = f\frac{y}{z}$$

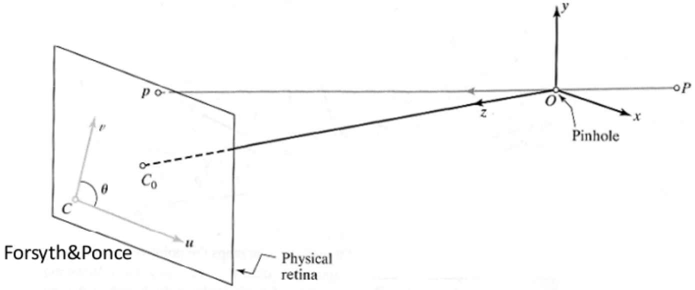
Intrinsic parameters: from idealized world coordinates to pixel values



- Unfortunately, "pixels" are in some arbitrary spatial units determined by the image sensor.

$$u = \alpha \frac{x}{z} \quad v = \alpha \frac{y}{z}$$

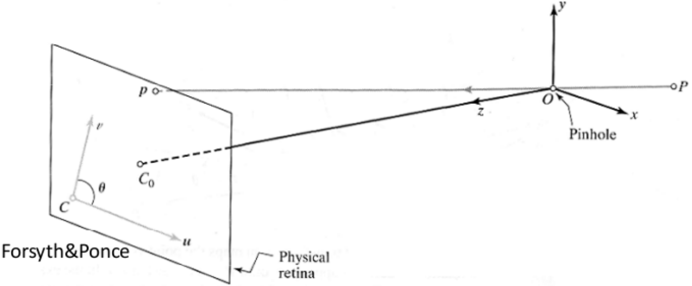
Intrinsic parameters: from idealized world coordinates to pixel values



- Pixels may not be square!

$$u = \alpha \frac{x}{z} \quad v = \beta \frac{y}{z}$$

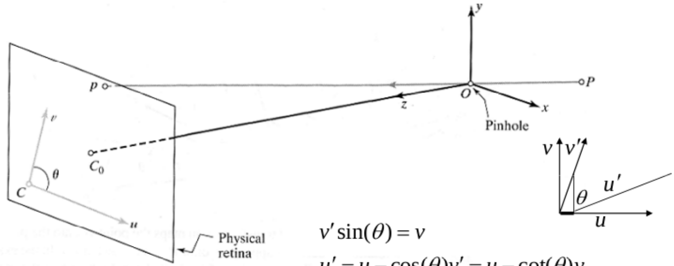
Intrinsic parameters: from idealized world coordinates to pixel values



- We need to account for the fact that the origin of our camera's pixel coordinates are unknown.

$$u = \alpha \frac{x}{z} + u_0 \quad v = \beta \frac{y}{z} + v_0$$

Intrinsic parameters: from idealized world coordinates to pixel values



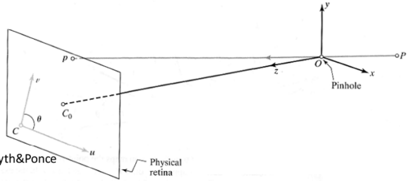
$$v' \sin(\theta) = v$$

$$u' = u - \cos(\theta)v' = u - \cot(\theta)v$$

- May be skew between the camera pixel axis.

$$u = \alpha\left(\frac{x}{z} - \cot(\theta)\frac{y}{z}\right) + u_0 \quad \text{and} \quad v = \frac{\beta}{\sin(\theta)}\frac{y}{z} + v_0$$

Intrinsic parameters: from idealized world coordinates to pixel values



Forsyth&Ponce

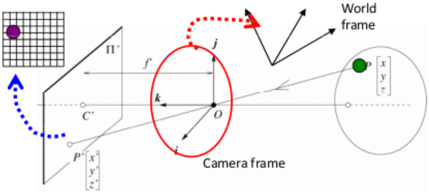
- Using homogeneous coordinates:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & \alpha \cot(\theta) & u_0 & 0 \\ 0 & \frac{\beta}{\sin(\theta)} & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

or $p = KP$ where p is in pixels, K is a calibration matrix, and P is a homogeneous coordinate vector in the camera's coordinate frame.

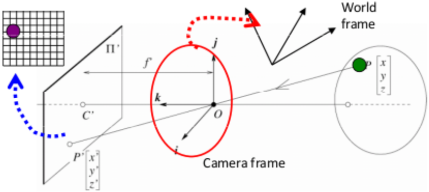
Extrinsic parameters: from world coordinates to pixel values

- What if the camera frame and the world frame are two distinct coordinate frames???



Extrinsic parameters: from world coordinates to pixel values

- What if the camera frame and the world frame are two distinct coordinate frames???
- Recall, we can always convert between frames via a rotation and translation!

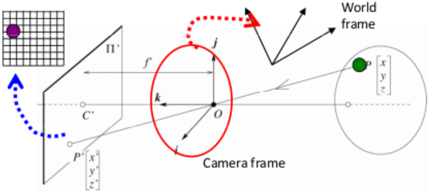


Extrinsic parameters: from world coordinates to pixel values

- What if the camera frame and the world frame are two distinct coordinate frames???
- Recall, we can always convert between frames via a rotation and translation!

  Let P denote the homogeneous coordinate vector in the camera's coordinate frame and ${}^{W}P$ the homogeneous vector in the world frame, then

$$P = \begin{bmatrix} R & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} {}^{W}P$$



Extrinsic parameters: from world coordinates to pixel values

- Using this notation, and substituting, we have

$$p = K\,{}^{W}P = K \begin{bmatrix} R & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} {}^{W}P = M\,{}^{W}P$$

where

$$M = \begin{bmatrix} \alpha \mathbf{r}_1^T - \alpha \cot(\theta) \mathbf{r}_2^T + u_0 \mathbf{r}_3^T & \alpha t_x - \alpha \cot(\theta) t_y + u_0 t_z \\ \frac{\beta}{\sin(\theta)} \mathbf{r}_2^T + v_0 \mathbf{r}_3^T & \frac{\beta}{\sin(\theta)} t_y + v_0 t_z \\ \mathbf{r}_3^T & t_z \end{bmatrix},$$

$\mathbf{r}_i$ is the i^{th} row of R, and t_i are the elements of $\mathbf{t}$.

The construction of M

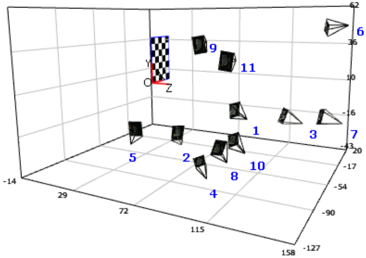
- Great: So M is a function of the five intrinsic camera parameters
  - $\alpha, \beta, u_0, v_0,$ and θ

The construction of M

- Great: So M is a function of the five intrinsic camera parameters
  - $\alpha, \beta, u_0, v_0,$ and θ
- If an arbitrary rotation matrix is constructed from three elementary rotations (common to use Euler angle parameterization), then we have the six extrinsic parameters
  - The three angles in R and the three coordinates of $\mathbf{t}$

The construction of M

- Great: So M is a function of the five intrinsic camera parameters
  - $\alpha, \beta, u_0, v_0,$ and θ
- If an arbitrary rotation matrix is constructed from three elementary rotations (common to use Euler angle parameterization), then we have the six extrinsic parameters
  - The three angles in R and the three coordinates of $\mathbf{t}$
- In practice, we commonly estimate M from a calibration image (checkerboard, calibration cube, etc.)



The construction of M (Camera calibration)

- Recall our transformation from world coords. to pixel coords.
  - $p = M\,{}^{W}P \implies \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} m_1^T \\ m_2^T \\ m_3^T \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix}$

South Dakota School of Mines & Technology Electrical & Computer Engineering

The construction of M (Camera calibration)

- Recall our transformation from world coords. to pixel coords.
  - $p = M\,{}^{W}P \implies \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} m_1^T \\ m_2^T \\ m_3^T \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix}$
- Converting from homogeneous coords. to inhomogeneous coords.

$$\begin{cases} u = \frac{m_1^T}{m_3^T} P \\ v = \frac{m_2^T}{m_3^T} P \end{cases}$$

The construction of M (Camera calibration)

- Recall our transformation from world coords. to pixel coords.
  - $p = M\,{}^{W}P \Longrightarrow \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} m_1^T \\ m_2^T \\ m_3^T \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix}$
- Converting from homogeneous coords. to inhomogeneous coords.

$$\begin{cases} u = \frac{m_1^T}{m_3^T} P \\ v = \frac{m_2^T}{m_3^T} P \end{cases}$$

- These equations relate image positions, u and v to points in the world (3-D positions) P (inhomogeneous coords.)

The construction of M (Camera calibration)

- Recall our transformation from world coords. to pixel coords.
  - $p = M\ {}^{W}P \implies \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} m_1^T \\ m_2^T \\ m_3^T \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix}$
- Converting from homogeneous coords. to inhomogeneous coords.

$$\begin{cases} u = \frac{m_1^T}{m_3^T} P \\ v = \frac{m_2^T}{m_3^T} P \end{cases}$$

- These equations relate image positions, u and v to points in the world (3-D positions) P (inhomogeneous coords.)

  $\therefore$ for each feature point i in our calibration image, we have:

  $(m_1 - u_i m_3)^T P = 0$

  $(m_2 - u_i m_3)^T P = 0$

The construction of M (Camera calibration)

- Note: for the measurement of $i = 1, 2, \ldots, n$ points,

$$(m_1 - u_i m_3)^T P = 0$$
$$(m_2 - u_i m_3)^T P = 0$$

we can re-write them in matrix form:

$$\begin{bmatrix} P_1^T & \mathbf{0}^T & -u_1 P_1^T \\ \mathbf{0}^T & P_1^T & -v_1 P_1^T \\ \cdots & \cdots & \cdots \\ P_n^T & \mathbf{0}^T & -u_n P_n^T \\ \mathbf{0}^T & P_n^T & -v_n P_n^T \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

The construction of M (Camera calibration)

- Expanding the equations yields:

$$\underbrace{\begin{bmatrix} P_{1x} & P_{1y} & P_{1z} & 1 & 0 & 0 & 0 & 0 & -u_1P_{1x} & -u_1P_{1y} & -u_1P_{1z} & -u_1 \\ 0 & 0 & 0 & 0 & P_{1x} & P_{1y} & P_{1z} & 1 & -v_1P_{1x} & -v_1P_{1y} & -v_1P_{1z} & -v_1 \\ \cdots & \cdots & \cdots & & & & & & & & & \\ P_{nx} & P_{ny} & P_{nz} & 1 & 0 & 0 & 0 & 0 & -u_nP_{nx} & -u_nP_{ny} & -u_nP_{nz} & -u_n \\ 0 & 0 & 0 & 0 & P_{nx} & P_{ny} & P_{nz} & 1 & -v_nP_{nx} & -v_nP_{ny} & -v_nP_{nz} & -v_n \end{bmatrix}}_{Q} \underbrace{\begin{bmatrix} m_{11} \\ m_{12} \\ m_{13} \\ m_{14} \\ m_{21} \\ m_{22} \\ m_{23} \\ m_{24} \\ m_{31} \\ m_{32} \\ m_{33} \\ m_{34} \end{bmatrix}}_{m} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

  or compactly we get $Qm = 0$.

- What does this tell you about Q in an ideal world?

The construction of M (Camera calibration)

- Goal: find the unit vector m that minimizes

$$\|Qm\|^2 = m^T Q^T Qm$$

The construction of M (Camera calibration)

- Goal: find the unit vector m that minimizes

$$\|Qm\|^2 = m^T Q^T Qm$$

- From linear optimization we know that the m that minimizes this is given by the eigenvector of $Q^T Q$ associated with the smallest eigenvalue! (this is the unit vector x that minimizes $x^T Q^T Qx$)

The construction of M (Camera calibration)

- Goal: find the unit vector m that minimizes

$$\|Qm\|^2 = m^T Q^T Qm$$

- From linear optimization we know that the m that minimizes this is given by the eigenvector of $Q^T Q$ associated with the smallest eigenvalue! (this is the unit vector x that minimizes $x^T Q^T Qx$)
- We note that the elements of m are the elements of the matrix M used to determine both the intrinsic and extrinsic camera parameters!

The construction of M (Camera calibration)

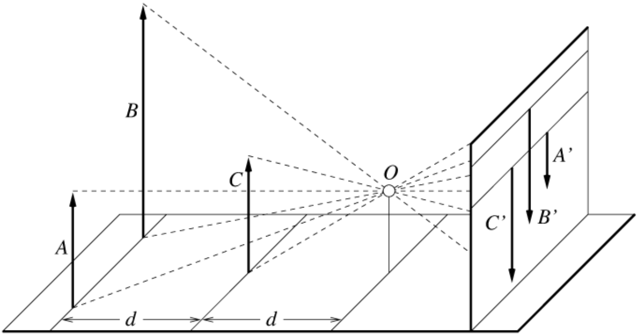
- Goal: find the unit vector m that minimizes

$$\|Qm\|^2 = m^T Q^T Qm$$

- From linear optimization we know that the m that minimizes this is given by the eigenvector of $Q^T Q$ associated with the smallest eigenvalue! (this is the unit vector x that minimizes $x^T Q^T Qx$)
- We note that the elements of m are the elements of the matrix M used to determine both the intrinsic and extrinsic camera parameters!
  - We can not relate arbitrary world points in the scene to pixel locations on the image sensor via the matrix M!

Perspective effects

- So now that we have our camera model, what effect does using a perspective projection model have?
  - Objects farther away from the camera appear smaller in the image!



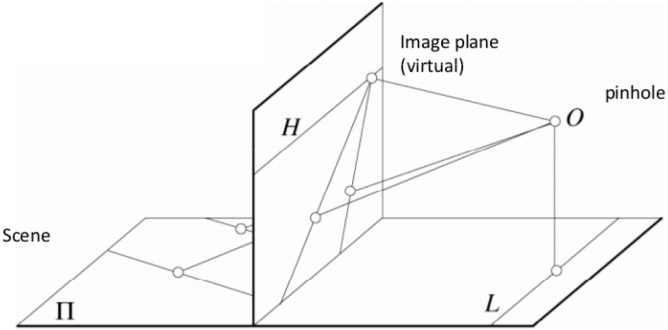
Perspective effects (far objects appear smaller)

Perspective effects (parallel lines converge to a point)

Perspective effects (parallel lines converge to a point)

Perspective effects

- Parallel lines in the scene intersect in the image
  - Converge in the image on a "horizon" line (a.k.a a vanishing point)



Properties of perspective projection

- Many-to-one: any points along the same ray map to the same point in the image
- Points map to?

Properties of perspective projection

- Many-to-one: any points along the same ray map to the same point in the image
- Points map to?
  - Points

Properties of perspective projection

- Many-to-one: any points along the same ray map to the same point in the image
- Points map to?
  - Points
- Lines map to?

Properties of perspective projection

- Many-to-one: any points along the same ray map to the same point in the image
- Points map to?
  - Points
- Lines map to?
  - Lines

Properties of perspective projection

- Many-to-one: any points along the same ray map to the same point in the image
- Points map to?
  - Points
- Lines map to?
  - Lines
- Distances and angles are/are not preserved?

Properties of perspective projection

- Many-to-one: any points along the same ray map to the same point in the image
- Points map to?
  - Points
- Lines map to?
  - Lines
- Distances and angles are/are not preserved?
  - Are NOT!

Properties of perspective projection

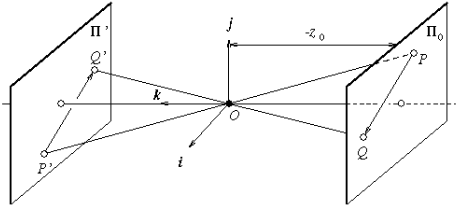
- Many-to-one: any points along the same ray map to the same point in the image
- Points map to?
  - Points
- Lines map to?
  - Lines
- Distances and angles are/are not preserved?
  - Are NOT!
- Degenerate cases:
  - Line through the focal point projects to a point.
  - Plane through focal point projects to a line.
  - Plane orthogonal to the image plane projects as part of the image.

Other projection models (briefly) - Weak perspective

- Approximation: treat magnification as a constant
  - Collect points into a single group at approx. the same depth, divide each point by the "depth" of it's group

Let $z = z_0$, then:

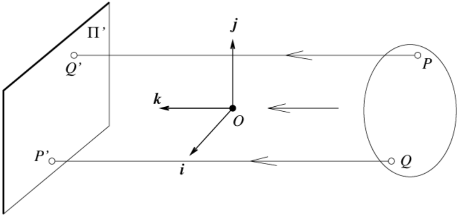
$$x' \approx f\frac{x}{z_0} \text{ and } y' \approx f\frac{y}{z_0}$$



Other projection models (briefly) - Orthographic

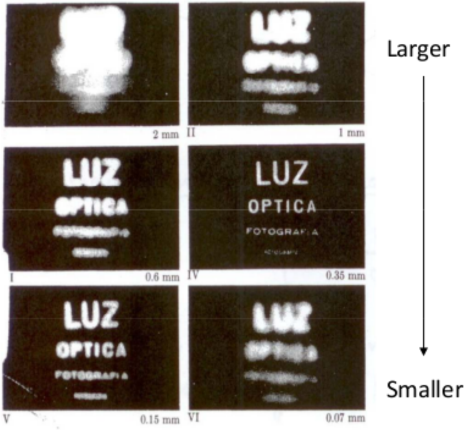
- Assume the camera is a **constant** distance from the scene!
  - World points projected along rays parallel to the optical axis $x' = x$ and $y' = y$ or

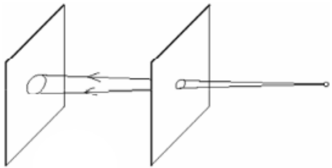
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \implies (x, y)$$



Pinhole approximation and lenses

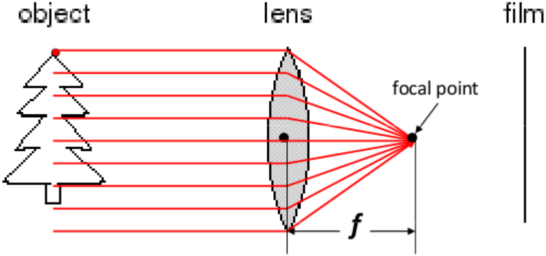
- Recall our pinhole model and how the aperture size effects the focus. (you should have noticed this in homework!)



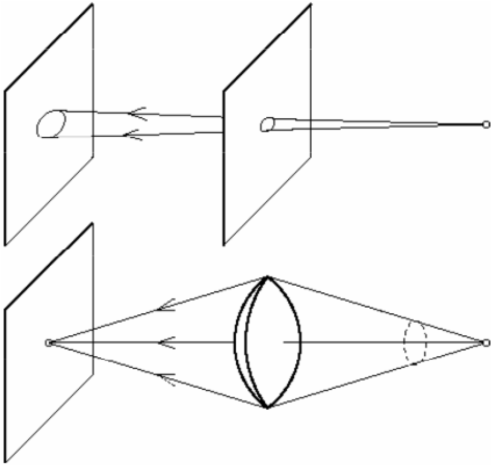


Adding a lens

- We can always add a lens!
  - A lens will focus light onto the film
    - Rays passing through the center are NOT deviated
    - All parallel rays converge to one point on a plane located at the *focal length f*

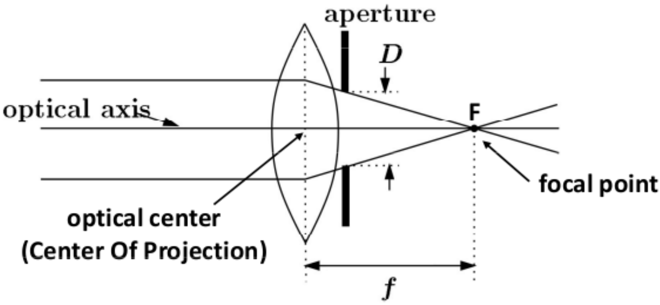


Pinhole vs. lens



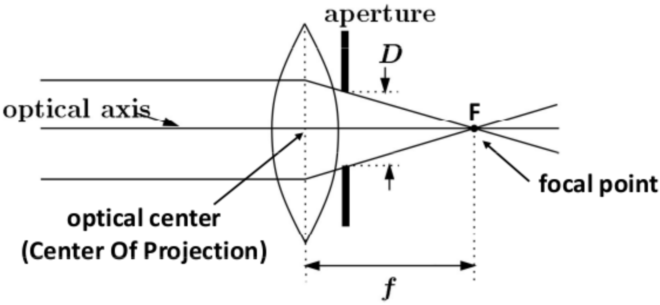
Adding a lens

- The lens focuses parallel rays onto a single focal point
  - This allows us to gather more light while maintaining focus



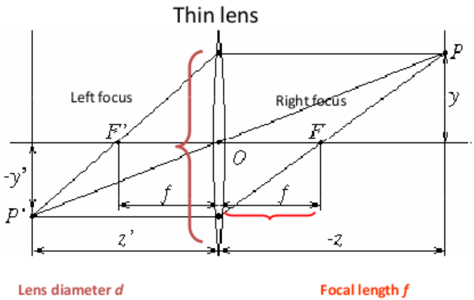
Adding a lens

- The lens focuses parallel rays onto a single focal point
  - This allows us to gather more light while maintaining focus
    - i.e., this makes the pinhole perspective projection model practical!



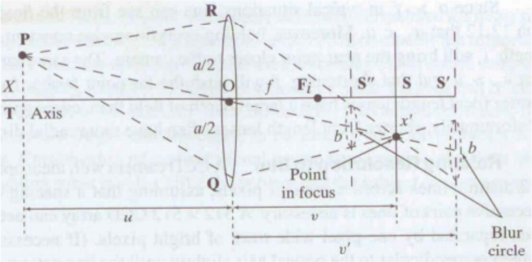
Adding a lens

- We assume a thin lens model:
  - Rays entering parallel on one side go through the lens and come to a common focus on the other side!
    - Ideally, all rays entering from P are images at P'!



What about depth of field???

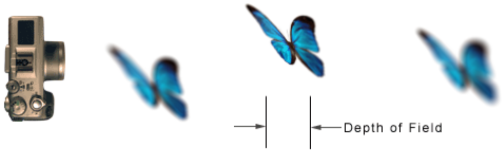
- Lens $\rightarrow$ points in the scene come in to focus at different image planes!



← "circles of confusion" →

What about depth of field???

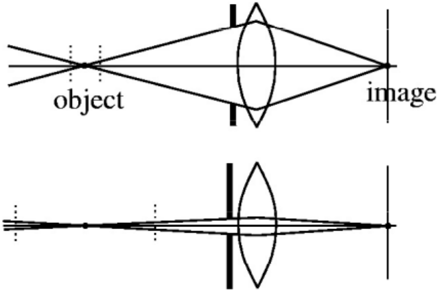
- Lens → points in the scene come in to focus at different image planes!
- What does this mean???



- Credit: http://en.wikipedia.org/wiki/Depth_of_field_diagram.png

Focus and depth of field

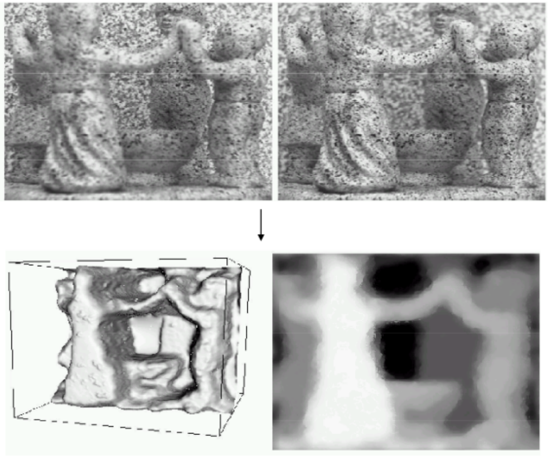
- How does the aperture effect the depth of field?
  - Smaller aperture $\implies$ larger depth of field!



- Flower image credits:
  http://en.wikipedia.org/wiki/Depth_of_field_diagram.png

How can we use this information?

- We can actually recover depth information from focus (sort of)



Images from same point of view, different camera parameters

3d shape / depth estimates

[figs from H. Jin and P. Favaro, 2002]

Field of view

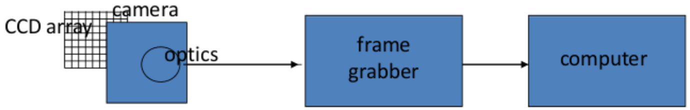
- Angular measure of the portion of the 3-D space seen by the camera

28 mm lens, 65.5° × 46.4°
50 mm lens, 39.6° × 27.0°
70 mm lens, 28.9° × 19.5°
210 mm lens, 9.8° × 6.5°

- Credits:
http://en.wikipedia.org/wiki/Angle_of_view.png

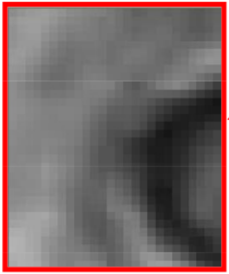
Digital cameras

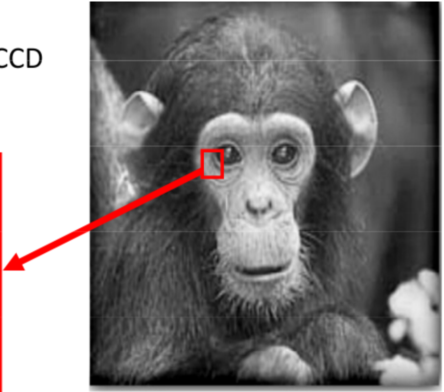
- Film → sensor array
- Can be an array of charge coupled devices (CCD) or CMOS
- Each CCD is a light sensitive diode that converts photons to electrons!



Digital images

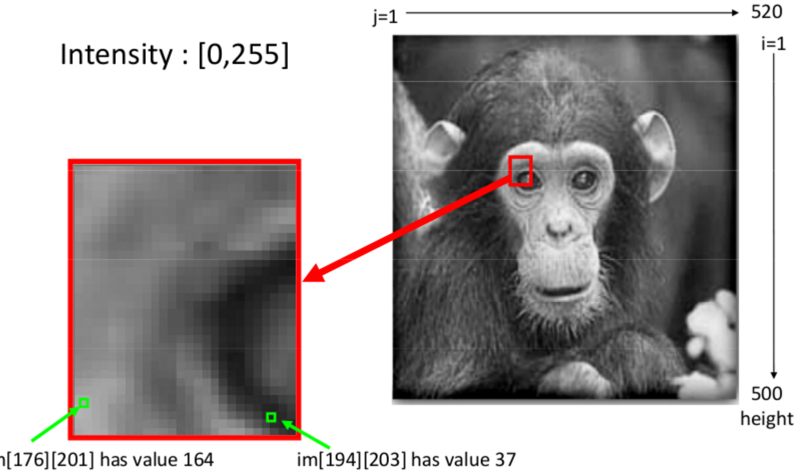
Think of images as matrices taken from CCD array.





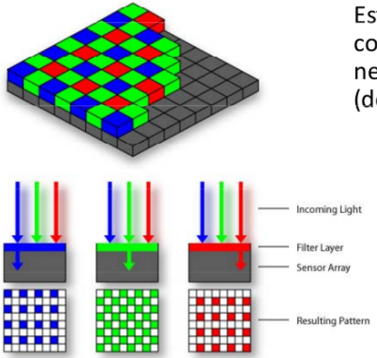
Digital images

Intensity : [0,255]



How do we detect color?

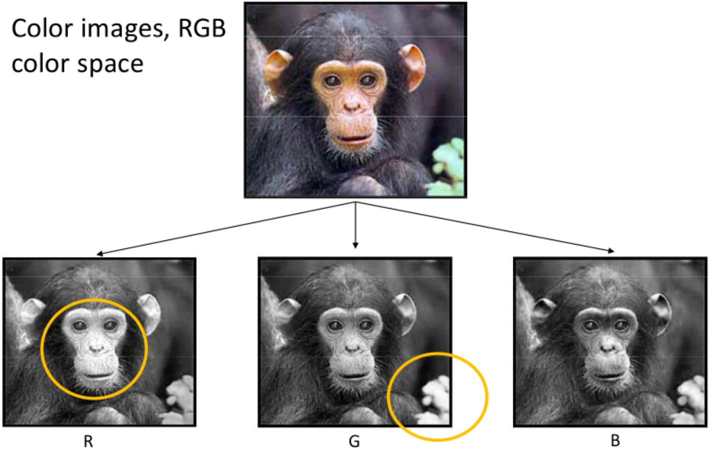
Bayer grid



Estimate missing components from neighboring values (demosaicing)

How do we detect color?

Color images, RGB color space



Summary

- Image formation is affected by geometry, photometry, and optics
- Projection equations express how 3-D world points are mapped to a 2-D image sensor
- Homogeneous coordinates allow us to use a linear system of equations for the projection matrix
- Lenses make the pinhole projection model practical
- Digital imagers, Bayer demosaicing

Parameters (focal length, aperture size, lens diameter, sensor sampling, etc.) all effect the image obtained!

Slide credits

- Dr. Jeff McGough
- Prof. Trever Darrel (who references his slide credits within)
- Forsyth and Ponce
- Rich Szeliski