Lecture 2: Image Formation

( unless otherwise stated slides are taken or adopted from Bob Woodham, Jim Little and Fred Tung )
Menu for Today (January 11, 2022)

Topics:

— Image Formation
— Cameras and Lenses
— Projection

Readings:

— Today’s Lecture: Forsyth & Ponce (2nd ed.) 1.1.1 — 1.1.3
— Next Lecture: Forsyth & Ponce (2nd ed.) 4.1, 4.5

Reminders:

— Complete Assignment 0 (ungraded) by Monday, January 1
— Please sign up for Piazza (116 students signed up so far)
— CoLab and Jupyter Notebooks for assignments
Today’s “fun” Example
Today’s “fun” Example

Photo credit: reddit user Liammm
Today’s “fun” Example: **Eye Sink Illusion**

Photo credit: reddit user Liammm
Salvador Dali — Pareidolia
Lecture 1: Re-cap

Types of computer vision problems:
- Computing properties of the 3D world from visual data (measurement)
- Recognition of objects and scenes (perception and interpretation)
- Search and interact with visual data (search and organization)
- Manipulation or creation of image or video content (visual imagination)

Computer vision challenges:
- Fundamentally ill-posed
- Enormous computation and scale
- Lack of fundamental understanding of how human perception works
Computer vision technologies have moved **from research labs into commercial products and services**. Examples cited include:

- broadcast television sports
- electronic games (Microsoft Kinect)
- biometrics
- image search
- visual special effects
- medical imaging
- robotics

... many others
Lecture 2: Goal

To understand how images are formed

(and develop relevant mathematical concepts and abstractions)
What is **Computer Vision**?

Compute vision, broadly speaking, is a research field aimed to enable computers to process and interpret visual data, as sighted humans can.

![Diagram showing the process of sensing and interpreting visual data](https://www.flickr.com/photos/flipflop/8376271518)
What is **Computer Vision**?

Compute vision, broadly speaking, is a research field aimed to enable computers to **process and interpret visual data**, as sighted humans can.
Overview: Image Formation, Cameras and Lenses

The **image formation process** that produces a particular image depends on:

- **Lightening condition**
- **Scene geometry**
- **Surface properties**
- **Camera optics and viewpoint**

Sensor (or eye) **captures amount of light** reflected from the object.
Visible light is electromagnetic radiation in the 400nm-700nm band of wavelengths.
Visible light is electromagnetic radiation in the 400nm-700nm band of wavelengths

- Black is the absence of light
- Sunlight is a spectrum of light
Visible light is electromagnetic radiation in the 400nm-700nm band of wavelengths.
**Light and Color: A Short Preview**

Visible light is electromagnetic radiation in the 400nm-700nm band of wavelengths.

Light also behaves as particles with specific wavelengths — **photons**; that travel in straight lines within a medium.
Overview: Image Formation, Cameras and Lenses

The **image formation process** that produces a particular image depends on

- **Lightening** condition
- **Scene geometry**
- **Surface** properties
- **Camera optics** and **viewpoint**

Sensor (or eye) **captures amount of light** reflected from the object
Graphics Review

source

normal

surface element

sensor
Graphics Review

- Source
- Normal
- Surface element
- Sensor
Surface reflection depends on both the viewing $(\theta_v, \phi_v)$ and illumination $(\theta_i, \phi_i)$ direction, with Bidirectional Reflection Distribution Function: $\text{BRDF}(\theta_i, \phi_i, \theta_v, \phi_v)$ 

Slide adopted from: Ioannis (Yannis) Gkioulekas (CMU)
Surface reflection depends on both the **viewing** \((\theta_v, \phi_v)\) and **illumination** \((\theta_i, \phi_i)\) direction, with Bidirectional Reflection Distribution Function: \(\text{BRDF}(\theta_i, \phi_i, \theta_v, \phi_v)\)

**Lambertian surface:**

\[
\text{BRDF}(\theta_i, \phi_i, \theta_v, \phi_v) = \frac{\rho d}{\pi}
\]

constant, called **albedo**

*Slide adopted from: Ioannis (Yannis) Gkioulekas (CMU)*
Surface reflection depends on both the viewing \((\theta_v, \phi_v)\) and illumination \((\theta_i, \phi_i)\) direction, with Bidirectional Reflection Distribution Function: \(\text{BRDF}(\theta_i, \phi_i, \theta_v, \phi_v)\)

**Lambertian surface:**

\[
\text{BRDF}(\theta_i, \phi_i, \theta_v, \phi_v) = \frac{\rho d}{\pi}
\]

constant, called **albedo**

---

<table>
<thead>
<tr>
<th>Surface type</th>
<th>Typical value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fresh asphalt</td>
<td>0.03 – 0.04</td>
</tr>
<tr>
<td>Open ocean</td>
<td>0.06</td>
</tr>
<tr>
<td>Conifer forest (summer)</td>
<td>0.08 – 0.15</td>
</tr>
<tr>
<td>Worn asphalt</td>
<td>0.12</td>
</tr>
<tr>
<td>Deciduous trees</td>
<td>0.15 – 0.18</td>
</tr>
<tr>
<td>Sand</td>
<td>0.15 – 0.45</td>
</tr>
<tr>
<td>Tundra</td>
<td>0.18 – 0.25</td>
</tr>
<tr>
<td>Agricultural crops</td>
<td>0.18 – 0.25</td>
</tr>
<tr>
<td>Bare soil</td>
<td>0.17</td>
</tr>
<tr>
<td>Green grass</td>
<td>0.20 – 0.25</td>
</tr>
<tr>
<td>Dessert sand</td>
<td>0.30 – 0.40</td>
</tr>
<tr>
<td>Snow</td>
<td>0.40 – 0.90</td>
</tr>
<tr>
<td>Ocean ice</td>
<td>0.50 – 0.70</td>
</tr>
<tr>
<td>Fresh snow</td>
<td>0.80 – 0.90</td>
</tr>
</tbody>
</table>

---

*Slide adopted from: Ioannis (Yannis) Gkioulekas (CMU)*
Surface reflection depends on both the viewing \((\theta_v, \phi_v)\) and illumination \((\theta_i, \phi_i)\) direction, with Bidirectional Reflection Distribution Function: \(\text{BRDF}(\theta_i, \phi_i, \theta_v, \phi_v)\)

Lambertian surface:

\[
\text{BRDF}(\theta_i, \phi_i, \theta_v, \phi_v) = \frac{\rho d}{\pi}
\]

\[
L = \frac{\rho d}{\pi} I(\vec{i} \cdot \vec{n})
\]
**Question:** What are the simplifying assumptions we are making here?

**Lambertian surface:**

\[
\text{BRDF}(\theta_i, \phi_i, \theta_v, \phi_v) = \frac{\rho d}{\pi}
\]

\[
L = \frac{\rho d}{\pi} I(\vec{i} \cdot \vec{n})
\]

*Slide adopted from: Ioannis (Yannis) Gkioulkekas (CMU)*
**Question:** What are the simplifying assumptions we are making here?

1. BRDF is the same everywhere (i.e., surface has identical properties everywhere)
2. Light spectra is absorbed uniformly by the surface (no change in color)

**Lambertian surface:**

\[
\text{BRDF}(\theta_i, \phi_i, \theta_v, \phi_v) = \frac{\rho d}{\pi}
\]

\[
L = \frac{\rho d}{\pi} I(\vec{i} \cdot \vec{n})
\]

*Slide adopted from: Ioannis (Yannis) Gkioulekas (CMU)*
Surface reflection depends on both the viewing $(\theta_v, \phi_v)$ and illumination $(\theta_i, \phi_i)$ direction, with Bidirectional Reflection Distribution Function: $\text{BRDF}(\theta_i, \phi_i, \theta_v, \phi_v)$

**Lambertian** surface:

$$\text{BRDF}(\theta_i, \phi_i, \theta_v, \phi_v) = \frac{\rho d}{\pi}$$

**Mirror** surface: all incident light reflected in one directions $(\theta_v, \phi_v) = (\theta_r, \phi_r)$

*Slide adopted from: Ioannis (Yannis) Gkioulekas (CMU)*
Cameras

Old school **film** camera

Digital **CCD/CMOS** camera
Cameras

Old school *film* camera

Digital CCD/CMOS camera
Let’s say we have a sensor …

**Digital** CCD/CMOS camera

*Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)*
Let’s say we have a sensor …

**Digital** CCD/CMOS camera

*Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)*
Let’s say we have a sensor …

**Digital** CCD/CMOS camera

digital sensor (CCD or CMOS)

*Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)*
... and the **object** we would like to photograph

What would an image taken like this look like?

real-world object

digital sensor (CCD or CMOS)

*Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)*
Bare-sensor imaging

real-world object

digital sensor (CCD or CMOS)

Slide Credit: Ioannis (Yannis) Gkioulakes (CMU)
Bare-sensor imaging

real-world object

digital sensor (CCD or CMOS)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Bare-sensor imaging

real-world object

digital sensor (CCD or CMOS)

*Slide Credit:* Ioannis (Yannis) Gkioulakas (CMU)
Bare-sensor imaging

real-world object

digital sensor (CCD or CMOS)

All scene points contribute to all sensor pixels

Slide Credit: Ioannis (Yannis) Gkioullekas (CMU)
Bare-sensor imaging

All scene points contribute to all sensor pixels

*Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)*
What would an image taken like this look like?
**Pinhole Camera**

real-world object

most rays are blocked

one makes it through

digital sensor (CCD or CMOS)

---

*Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)*
Pinhole Camera

Each scene point contributes to only one sensor pixel

real-world object

digital sensor (CCD or CMOS)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Reinerus Gemma-Frisius observed an eclipse of the sun at Louvain on January 24, 1544. He used this illustration in his book, “De Radio Astronomica et Geometrica,” 1545. It is thought to be the first published illustration of a camera obscura.

*Credit: John H., Hammond, “Th Camera Obscure, A Chronicle”*
Camera Obscura (latin for “dark chamber”)

Reinerus Gemma-Frisius observed an eclipse of the sun at Louvain on January 24, 1544. He used this illustration in his book, “De Radio Astronomica et Geometrica,” 1545. It is thought to be the first published illustration of a camera obscura.

Credit: John H., Hammond, “Th Camera Obscure, A Chronicle”
First **Photograph** on Record

*La table servie*

Credit: Nicéphore Niepce, 1822
A pinhole camera is a box with a small hall (aperture) in it.
A pinhole camera is a box with a small hall (aperture) in it.
Image Formation

Forsyth & Ponce (2nd ed.) Figure 1.1

Credit: US Navy, Basic Optics and Optical Instruments. Dover, 1969
Accidental Pinhole Camera

Image Credit: Ioannis (Yannis) Gkioulekas (CMU)
Pinhole Camera (Simplified)

$f'$ is the **focal length** of the camera.
**Pinhole Camera (Simplified)**

\[ f' \text{ is the focal length of the camera} \]

![Diagram of a pinhole camera with labels: image plane, pinhole, object, and dimensions x', f', z, x.]

**Note:** In a pinhole camera we can adjust the focal length, all this will do is change the size of the resulting image.
It is convenient to think of the **image plane** which is in front of the pinhole.
Pinhole Camera (Simplified)

It is convenient to think of the image plane which is in front of the pinhole.

What happens if object moves towards the camera? Away from the camera?
Focal Length

For a fixed sensor size, focal length determines the **field of view** (FoV).
For a fixed sensor size, focal length determines the **field of view** (FoV).

**Exercise:** What is the field of view of a full frame (35mm) camera with a 50mm lens? 100mm lens?
Perspective Effects

Forsyth & Ponce (2nd ed.) Figure 1.3a
**Perspective Effects**

**Far objects** appear **smaller** than close ones.

Forsyth & Ponce (2nd ed.) Figure 1.3a
Perspective Effects

Far objects appear smaller than close ones

Forsyth & Ponce (2nd ed.) Figure 1.3a
Perspective Effects

Far objects appear smaller than close ones

Size is inversely proportional to distance

Forsyth & Ponce (2nd ed.) Figure 1.3a
Perspective Effects

Far objects appear smaller than close ones

Forsyth & Ponce (2nd ed.) Figure 1.3a
Perspective Effects

Forsyth & Ponce (1st ed.) Figure 1.3b
Parallel lines meet at a point (vanishing point)

Forsyth & Ponce (1st ed.) Figure 1.3b
Vanishing Points

Each set of parallel lines meet at a different point
— the point is called **vanishing point**
Vanishing Points

Each set of parallel lines meet at a different point
— the point is called vanishing point

Sets of parallel lines on the same plane lead to collinear vanishing points
— the line is called a horizon for that plane
Vanishing Points

Draw a horizon line.
Vanishing Points

1. Draw a horizon line.
2. Make a vanishing point.

*Slide Credit: David Jacobs*
Vanishing Points

1. Draw a horizon line.
2. Make a vanishing point.
3. Draw a square or rectangle.
Vanishing Points

1. Draw a horizon line.
2. Make a vanishing point.
3. Draw a square or rectangle.
4. Draw orthogonals from shape corners to vanishing point.

Slide Credit: David Jacobs
Vanishing Points

Draw a horizon line.

Make a vanishing point.

Draw a square or rectangle.

Draw orthogonals from shape corners to vanishing point.

Draw a horizontal line to end your form.

Slide Credit: David Jacobs
Vanishing Points

1. Draw a horizon line.
2. Make a vanishing point.
3. Draw a square or rectangle.
4. Draw orthogonal lines from shape corners to vanishing point.
5. Draw a horizontal line to end your form.
6. Draw a vertical line to make the form's side.

Slide Credit: David Jacobs
Vanishing Points

1. Draw a horizon line.
2. Make a vanishing point.
3. Draw a square or rectangle.
4. Draw orthogonal lines from shape corners to vanishing point.
5. Draw a horizontal line to end your form.
6. Draw a vertical line to make the form's side.
7. Erase the orthogonal lines.

Slide Credit: David Jacobs
Vanishing Points

1. Draw a horizon line.
2. Make a vanishing point.
3. Draw a square or rectangle.
4. Draw orthogonals from shape corners to vanishing point.
5. Draw a horizontal line to end your form.
6. Draw a vertical line to make the form’s side.
7. Erase the orthogonals.
8. Draw another form!

Slide Credit: David Jacobs
Vanishing Points

1. Draw a horizon line.
2. Make a vanishing point.
3. Draw a square or rectangle.
4. Draw orthogonals from shape corners to vanishing point.
5. Draw a horizontal line to end your form.
6. Draw a vertical line to make the form’s side.
7. Erase the orthogonals.
8. Draw another form!
9. Add windows and doors.

Slide Credit: David Jacobs
Vanishing Points

Each set of parallel lines meet at a different point
— the point is called **vanishing point**

Sets of parallel lines on the same plane lead to **collinear** vanishing points
— the line is called a **horizon** for that plane

Good way to **spot fake images**
— scale and perspective do not work
— vanishing points behave badly
Vanishing Points

Slide Credit: Efros (Berkeley), photo from Criminisi
Vanishing Points

Slide Credit: Efros (Berkeley), photo from Criminisi
Vanishing Points

Slide Credit: Efros (Berkeley), photo from Criminisi
Vanishing Points

Vertical vanishing point (at infinity)

Vanishing point

Vanishing point

Slide Credit: Efros (Berkeley), photo from Criminisi
Vanishing Points

Two point perspective

Vertical vanishing point (at infinity)

Vanishing point

Vanishing point

Slide Credit: Efros (Berkeley), photo from Criminisi
Vanishing Points

One point perspective

Add windows and doors.

Two point perspective

Vertical vanishing point (at infinity)

Vanishing point

Vanishing point

Slide Credit: Efros (Berkeley), photo from Criminisi
Perspective Aside

Image credit: http://www.martinacecilia.com/place-vanishing-points/
Perspective Aside

Image credit: http://www.martinacecilia.com/place-vanishing-points/
Properties of Projection

- **Points** project to **points**
- **Lines** project to **lines**
- **Planes** project to the **whole** or **half** image
- Angles are **not** preserved
Properties of Projection

- **Points** project to **points**
- **Lines** project to **lines**
- **Planes** project to the **whole** or **half** image
- Angles are **not** preserved

**Degenerate cases**
- Line through focal point projects to a point
- Plane through focal point projects to a line
Projection Illusion
Projection Illusion
Perspective Projection

3D object point

\[
P = \begin{bmatrix} x \\ y \\ z \end{bmatrix}
\]

projects to 2D image point

\[
P' = \begin{bmatrix} x' \\ y' \end{bmatrix}
\]

where

\[
x' = f' \frac{x}{z}
\]
\[
y' = f' \frac{y}{z}
\]

Note: this assumes world coordinate frame at the optical center (pinhole) and aligned with the image plane, image coordinate frame aligned with the camera coordinate frame.
Perspective Projection: Proof

3D object point

\[ P = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \]

projects to 2D image point

\[ P' = \begin{bmatrix} x' \\ y' \end{bmatrix} \]

where

\[
\begin{align*}
    x' &= f' \frac{x}{z} \\
    y' &= f' \frac{y}{z}
\end{align*}
\]

Note: this assumes world coordinate frame at the optical center (pinhole) and aligned with the image plane, image coordinate frame aligned with the camera coordinate frame.
Aside: Camera Matrix

Forsyth & Ponce (1st ed.) Figure 1.4

Camera Matrix

\[ C = \begin{bmatrix} f' & 0 & 0 & 0 \\ 0 & f' & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \]

3D object point

\[ P = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \]

projects to 2D image point

\[ P' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \]

where

\[ P' = CP \]

Note: this assumes world coordinate frame at the optical center (pinhole) and aligned with the image plane, image coordinate frame aligned with the camera coordinate frame.
 Aside: Camera Matrix

Camera Matrix

\[
C = \begin{bmatrix}
  f' & 0 & 0 & 0 \\
  0 & f' & 0 & 0 \\
  0 & 0 & 1 & 0 \\
\end{bmatrix}
\]

\[
P = \begin{bmatrix}
  x \\
  y \\
  z \\
  1
\end{bmatrix}
\]

projects to 2D image point

\[
P' = \begin{bmatrix}
  x' \\
  y' \\
  1
\end{bmatrix}
\]

where

\[
P' = CP
\]

\[
x' = f' \frac{x}{z}
\]

\[
y' = f' \frac{y}{z}
\]
Aside: Camera Matrix

Camera Matrix

\[
C = \begin{bmatrix}
  f' & 0 & 0 & 0 \\
  0 & f' & 0 & 0 \\
  0 & 0 & 1 & 0
\end{bmatrix}
\]

\[
\begin{bmatrix}
  f' & 0 & 0 & 0 \\
  0 & f' & 0 & 0 \\
  0 & 0 & 1 & 0
\end{bmatrix}\begin{bmatrix}
x \\
y \\
z \\
1
\end{bmatrix} = \begin{bmatrix}
f'x \\
f'y \\
\frac{f'}{z}
\end{bmatrix} = \begin{bmatrix}
\frac{f'}{z} \\
\frac{f'y}{z} \\
1
\end{bmatrix}
\]

\[
P = \begin{bmatrix}
x \\
y \\
z \\
1
\end{bmatrix}
\]
projects to 2D image point \( P' = \begin{bmatrix}
x' \\
y' \\
1
\end{bmatrix} \)
where \( P' = CP \)
Aside: Camera Matrix

Camera Matrix

$$C = \begin{bmatrix} f' & 0 & 0 & 0 \\ 0 & f' & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Pixels are squared / lens is perfectly symmetric
Sensor and pinhole perfectly aligned
Coordinate system centered at the pinhole

$$P = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$
projects to 2D image point

$$P' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$$

where

$$P' = CP$$
**Aside: Camera Matrix**

**Camera Matrix**

\[ C = \begin{bmatrix} f_x' & 0 & 0 & 0 \\ 0 & f_y' & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \]

- Pixels are squared / lens is perfectly symmetric
- Sensor and pinhole perfectly aligned
- Coordinate system centered at the pinhole

\[ P = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \] projects to 2D image point

\[ P' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \] where \[ P' = CP \]
Aside: Camera Matrix

**Camera Matrix**

\[
C = \begin{bmatrix}
  f_x' & 0 & 0 & c_x \\
  0 & f_y' & 0 & c_y \\
  0 & 0 & 1 & 0
\end{bmatrix}
\]

*Pixels are squared / lens is perfectly symmetric*

*Sensor and pinhole perfectly aligned*

*Coordinate system centered at the pinhole*

\[
P = \begin{bmatrix}
  x \\
  y \\
  z \\
  1
\end{bmatrix}
\]

projects to 2D image point \( P' = \begin{bmatrix}
  x' \\
  y' \\
  1
\end{bmatrix} \)

where \( P' = CP \)
Aside: Camera Matrix

Camera Matrix

\[ C = \begin{bmatrix} f_x' & 0 & 0 & c_x \\ 0 & f_y' & 0 & c_y \\ 0 & 0 & 1 & 0 \end{bmatrix} \mathbb{R}_{4 \times 4} \]

- Pixels are squared / lens is perfectly symmetric
- Sensor and pinhole perfectly aligned
- Coordinate system centered at the pinhole

\[ P = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \]

projects to 2D image point

\[ P' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \]

where

\[ P' = CP \]
Aside: Camera Matrix

**Camera Matrix**

\[
C = \begin{bmatrix}
    f'_x & 0 & 0 & c_x \\
    0 & f'_y & 0 & c_y \\
    0 & 0 & 1 & 0
\end{bmatrix} \in \mathbb{R}^{4 \times 4}
\]

**Camera calibration** is the process of estimating parameters of the camera matrix based on set of 3D-2D correspondences (usually requires a pattern whose structure and size is known).

\[
P = \begin{bmatrix}
    x \\
    y \\
    z \\
    1
\end{bmatrix}
\]

projects to 2D image point \( P' = \begin{bmatrix}
    x' \\
    y' \\
    1
\end{bmatrix} \) where \( P' = CP \).
**Perspective Projection**

3D object point

\[ P = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \]

projects to 2D image point

\[ P' = \begin{bmatrix} x' \\ y' \end{bmatrix} \]

where

\[ x' = f' \frac{x}{z} \]

\[ y' = f' \frac{y}{z} \]

**Note:** this assumes world coordinate frame at the optical center (pinhole) and aligned with the image plane, image coordinate frame aligned with the camera coordinate frame.
Weak Perspective

3D object point \( P = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \) in \( \Pi_0 \) projects to 2D image point \( P' = \begin{bmatrix} x' \\ y' \end{bmatrix} \)

where \( x' = mx \) and \( y' = my \)

and \( m = \frac{f'}{z_0} \)
Orthographic Projection

3D object point \( P = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \) projects to 2D image point \( P' = \begin{bmatrix} x' \\ y' \end{bmatrix} \)

where

\[
\begin{align*}
x' &= x \\
y' &= y
\end{align*}
\]
Summary of **Projection Equations**

3D object point  \( P = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \) projects to 2D image point  \( P' = \begin{bmatrix} x' \\ y' \end{bmatrix} \) where

- **Perspective**
  \[
  \begin{align*}
  x' &= f' \frac{x}{z} \\
  y' &= f' \frac{y}{z}
  \end{align*}
  \]
- **Weak Perspective**
  \[
  \begin{align*}
  x' &= mx \\
  y' &= my
  \end{align*}
  \]
  \[
  m = \frac{f'}{z_0}
  \]
- **Orthographic**
  \[
  \begin{align*}
  x' &= x \\
  y' &= y
  \end{align*}
  \]
Projection Models: Pros and Cons

Weak perspective (including orthographic) has simpler mathematics
  — accurate when object is small and/or distant
  — useful for recognition

Perspective is more accurate for real scenes

When maximum accuracy is required, it is necessary to model additional details of a particular camera
  — use perspective projection with additional parameters (e.g., lens distortion)
Why Not a Pinhole Camera?

— If pinhole is too big then many directions are averaged, blurring the image

— If pinhole is too small then diffraction becomes a factor, also blurring the image

— Generally, pinhole cameras are dark, because only a very small set of rays from a particular scene point hits the image plane

— Pinhole cameras are slow, because only a very small amount of light from a particular scene point hits the image plane per unit time

Image Credit: Credit: E. Hecht. “Optics,” Addison-Wesley, 1987
Reason for **Lenses**

A real camera must have a finite aperture to get enough light, but this causes blur in the image.

**Solution**: use a lens to focus light onto the image plane
Reason for **Lenses**

A real camera must have a finite aperture to get enough light, but this causes blur in the image.

The role of a lens is to **capture more light** while preserving, as much as possible, the abstraction of an ideal pinhole camera.

**Solution**: use a **lens** to focus light onto the image plane.
Snell’s Law

\[ n_1 \sin \alpha_1 = n_2 \sin \alpha_2 \]
Snell's Law

\[ n_1 \sin \alpha_1 = n_2 \sin \alpha_2 \]
Pinhole Model (Simplified) with Lens
General Lens

Ray of light

Normal

Light bent away from the normal
Thin Lens Equation

\[ \frac{1}{z'} - \frac{1}{z} = \frac{1}{f} \]

Forsyth & Ponce (1st ed.) Figure 1.9
Thin Lens Equation

\[ \frac{1}{z'} + \frac{1}{z} = \frac{1}{f} \]

Forsyth & Ponce (1st ed.) Figure 1.9
Thin Lens Equation

Forsyth & Ponce (1st ed.) Figure 1.9

\[
\frac{1}{z'} - \frac{1}{z} = \frac{1}{f}
\]
Thin Lens Equation: Derivation

\[ \frac{1}{z'} - \frac{1}{z} = \frac{1}{f} \]

Forsyth & Ponce (1st ed.) Figure 1.9
Thin Lens Equation: Derivation

\[
\frac{y}{-z} = \frac{-y'}{z'} \\
\frac{y}{y'} = \frac{z}{z'}
\]

\[
\frac{1}{z'} - \frac{1}{z} = \frac{1}{f}
\]

Forsyth & Ponce (1st ed.) Figure 1.9
Thin Lens Equation: Derivation

\[
\frac{y}{-z} = \frac{-y'}{z'} \quad \frac{y}{y'} = \frac{z}{z'}
\]

Forsyth & Ponce (1st ed.) Figure 1.9

\[
\frac{1}{z'} - \frac{1}{z} = \frac{1}{f}
\]
Thin Lens Equation: Derivation

\[ \frac{y}{-z} = \frac{-y'}{z'} \]
\[ \frac{y}{y'} = \frac{z}{z'} \]

\[ \frac{-y'}{f} = \frac{y - y'}{-z} \]
\[ \frac{1}{f} = \frac{y - y'}{z y'} \]
\[ = \frac{y}{z y'} - \frac{y'}{z y'} \]
\[ = \frac{y}{z y'} - \frac{1}{z} \]

\[ \frac{1}{z'} - \frac{1}{z} = \frac{1}{f} \]

Forsyth & Ponce (1st ed.) Figure 1.9
Thin Lens Equation: Derivation

Forsyth & Ponce (1st ed.) Figure 1.9

\[ \frac{y}{-z} = \frac{-y'}{z'} \]

\[ \frac{y}{y'} = \frac{z}{z'} \]

Substitute:

\[ \frac{1}{z'} - \frac{1}{z} = \frac{1}{f} \]

\[ \frac{-y'}{f} = \frac{y - y'}{-z} \]

\[ \frac{1}{f} = \frac{y - y'}{zy'} \]

\[ = \frac{y}{zy'} - \frac{y'}{zy'} \]

\[ = \frac{y}{zy'} - \frac{1}{z} \]

Substitute: \[ \frac{1}{f} = \frac{1}{\frac{z}{z'} - \frac{1}{z}} \]
Possible Uses of Thin Lens Abstraction

Forsyth & Ponce (1st ed.) Figure 1.9

\[ \frac{1}{z'} - \frac{1}{z} = \frac{1}{f} \]
A lens focuses parallel rays (from points at infinity) at focal length of the lens.

Rays passing through the center of the lens are not bent.

To focus closer, we have to move the image plane back.

\[ \frac{1}{z'} - \frac{1}{z} = \frac{1}{f} \]
Lenses focus all rays from a (parallel to lense) plane in the world.

\[ \frac{1}{z'} - \frac{1}{z} = \frac{1}{f} \]

Objects off the plane are blurred depending on the distance.
Effect of **Aperture Size**

Smaller aperture $\Rightarrow$ smaller blur, larger **depth of field**
Depth of Field

Aperture size = f/N, $\Rightarrow$ large N = small aperture