168 442 Introduction to Image Processing

The First Semester of Class 2546

Dr. Nawapak Eua-Anant

Department of Computer Engineering
Khon Kaen University
Course Syllabus

Date and Time: MW 11.00-12.00 EN 4510, LAB1 TU17-20, LAB2 TH17-20

Assessments:
- Attendance & Homework 5%
- Lab and Homework 35%
- Midterm 30%
- Final 30%

Grading:
- 85-100% A, 75-85% B+, 70-75% B, 65-70% C+, 60-65% C,
- 55-60% D+, 50-55% D, 0-50% F

References:
Course Outline

1. Introduction
2. Digital Image Fundamentals
3. Image Transforms
4. Image Enhancement
5. Image Segmentation
6. Image Compression
7. Image Morphology
Chapter 1

Introduction to Image Processing
What is Digital Image Processing?

Processing of a multidimensional pictures by a digital computer.

Why we need Digital Image Processing?

1. เพื่อบันทึกและจัดเก็บภาพ
2. เพื่อปรับปรุงภาพให้ดีขึ้นโดยใช้กระบวนการทางคณิตศาสตร์
3. เพื่อช่วยในการวิเคราะห์รูปภาพ
4. เพื่อสังเคราะห์ภาพ
5. เพื่อสร้างระบบการมองเห็นให้กับคอมพิวเตอร์
**Digital Image**

Digital image = a multidimensional array of numbers (such as intensity image) or vectors (such as color image)

Each component in the image called pixel associates with the pixel value (a single number in the case of intensity images or a vector in the case of color images).
Visual Perception: Human Eye

(Picture from Microsoft Encarta 2000)
Chapter 2: Digital Image Fundamentals

FIGURE 2.1
Simplified diagram of a cross section of the human eye.
Visual Perception: Human Eye (cont.)

1. The **lens** contains 60-70% water, 6% of fat.
2. 
3. The **iris** diaphragm controls amount of light that enters the eye.
4. 
5. **Light receptors** in the **retina**
   - About 6-7 millions **cones** for bright light vision called **photopic**
   - Density of cones is about 150,000 elements/mm².
   - Cones involve in color vision.
   - Cones are concentrated in **fovea** about 1.5x1.5 mm².
   - About 75-150 millions **rods** for dim light vision called **scotopic**
   - Rods are sensitive to low level of light and are not involved color vision.

4. **Blind spot** is the region of emergence of the optic nerve from the eye.
FIGURE 2.2
Distribution of rods and cones in the retina.
Image Formation in Human Eye

Normal focus

Nearsighted focus

Farsighted focus

© Microsoft Corporation. All Rights Reserved.
FIGURE 2.3
Graphical representation of the eye looking at a palm tree. Point $C$ is the optical center of the lens.
Brightness Adaptation of Human Eye

Intensities of surrounding points effect perceived brightness at each point. In this image, edges between bars appear brighter on the right side and darker on the left side.
In area A, brightness perceived is darker while in area B is brighter. This phenomenon is called *Mach Band Effect*. 
Simultaneous contrast. All small squares have exactly the same intensity but they appear progressively darker as background becomes lighter.
**Imaging Geometry: Perspective Transformation**

\[(X, Y, Z) = \text{world coordinate}\]

\[(x, y, z) = \text{Camera coordinate system}\]

\[\lambda = \text{focal length}\]

\[
x = -\frac{X}{\lambda} - \frac{X}{Z - \lambda}\]

\[
y = -\frac{Y}{\lambda} - \frac{Y}{Z - \lambda}\]

\[
z = \frac{Z - \lambda}{\lambda}\]
**Imaging Geometry: Perspective Transformation (cont.)**

**Question:** How can we project the real world object at \((X,Y,Z)\) onto the image plane (such as photographic film)?

**Answer:** Relation between camera coordinate \((x,y,z)\) and world coordinate \((X,Y,Z)\) are given by

\[
\begin{bmatrix}
  x \\
  y \\
  z
\end{bmatrix} = \begin{bmatrix}
  \frac{\lambda X}{\lambda - Z} \\
  \frac{\lambda Y}{\lambda - Z} \\
  \frac{\lambda Z}{\lambda - Z}
\end{bmatrix}
\]

Eq. 1.2

Since on the image plane \(z\) is always zero, \(z=0\), we consider only \((x,y)\) while \(z\) is neglected.
Equation 1.2 is not linear because of $Z$ in the dividers so we introduce the *homogeneous coordinate* to solve this problem.

Cartesian coordinate: \[ w = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \]

Homogeneous coordinate: \[ w_h = \begin{bmatrix} kX \\ kY \\ kZ \\ k \end{bmatrix} \]

$k = \text{nonzero constant}$

To convert from the homogeneous coordinate $w_h$ to the Cartesian coordinate $w$, we divide the first 3 components of $w_h$ by the fourth component.
Imaging Geometry: Perspective Transformation (cont.)

The perspective transformation matrix for the homogeneous coordinate:

\[
P = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & -\frac{1}{\lambda} & 1
\end{bmatrix}
\]

Perspective transformation becomes:

\[
c_h = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & -\frac{1}{\lambda} & 1
\end{bmatrix}\begin{bmatrix}
kX \\
kY \\
kZ \\
k
\end{bmatrix} = \begin{bmatrix}
kX \\
kY \\
kZ \\
-\frac{k(Z-\lambda)}{\lambda}
\end{bmatrix}
\]

Eq. 1.3
Imaging Geometry: Perspective Transformation (cont.)

From homogeneous coordinate

\[
c_h = \begin{bmatrix} kX \\ kY \\ kZ \\ -k(Z - \lambda) \\ \lambda \end{bmatrix}
\]

We get camera coordinate in the image plane:

\[
c = \begin{bmatrix} kX \cdot \frac{\lambda}{-k(Z - \lambda)} \\ kY \cdot \frac{\lambda}{-k(Z - \lambda)} \\ kZ \cdot \frac{\lambda}{-k(Z - \lambda)} \end{bmatrix} = \begin{bmatrix} \frac{\lambda X}{\lambda - Z} \\ \frac{\lambda Y}{\lambda - Z} \\ \frac{\lambda Z}{\lambda - Z} \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}
\]
Imaging Geometry: Inverse Perspective Transformation

\[ w_h = P^{-1} c_h \quad \text{where} \quad P^{-1} = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & \frac{1}{\lambda} & 1
\end{bmatrix} \]

Eq. 1.4
Inverse Perspective Transformation (cont.)

For an image point \((x_0, y_0)\), since on the image plane \(z=0\), we have

\[
c_h = \begin{bmatrix}
kx_0 \\
ky_0 \\
0 \\
k
\end{bmatrix}
\]

We get the world coordinate:

\[
w_h = P^{-1}c_h = \begin{bmatrix}
kx_0 \\
ky_0 \\
0 \\
k
\end{bmatrix}
\]

\[
or \quad w = \begin{bmatrix}
x_0 \\
y_0 \\
0
\end{bmatrix} \rightarrow ???
\]

*Since the perspective transformation maps 3-D coordinates to 2-D Coordinates, we cannot get the inverse transform unless we have additional information.*
Inverse Perspective Transformation (cont.)

To find the solution, let

\[ \mathbf{c}_h = \begin{bmatrix} kx_0 \\ ky_0 \\ kz \\ k \end{bmatrix} \]

We get

\[ w_h = P^{-1} c_h = \begin{bmatrix} kx_0 \\ ky_0 \\ kz \\ k(z + \lambda) \end{bmatrix} \]

or

\[ w = \begin{bmatrix} X = \frac{\lambda x_0}{\lambda + z} \\ Y = \frac{\lambda y_0}{\lambda + z} \\ Z = \frac{\lambda z}{\lambda + z} \end{bmatrix} \]

Eq. 1.5
Inverse Perspective Transformation (cont.)

From Eq. 1.5,

We get

\[ z = \frac{\lambda Z}{\lambda - Z} \quad \text{Eq. 1.6} \]

Substituting Eq. 1.6 into Eq. 1.5, we get

\[ X = \frac{x_0}{\lambda} (\lambda - Z) \quad \text{Eq. 1.7} \]

\[ Y = \frac{y_0}{\lambda} (\lambda - Z) \]

Equations 1.7 show that inverse perspective transformation requires information of at least one component of the world coordinate of the point.
Inverse Perspective Transformation (cont.)

These equations: $X = \frac{x_0}{\lambda} (\lambda - Z)$ and $Y = \frac{y_0}{\lambda} (\lambda - Z)$ show that points on Line $L$ in the world coordinate map to a single point in the image plane.

Points $(X_1, Y_1, Z_1)$, $(X_2, Y_2, Z_2)$, and $(X_3, Y_3, Z_3)$ map to Point $(x, y)$ in the image plane.
Stereo Imaging: How we get depth information from 2 eyes

- **Image 1**
  - Left lens center
  - Optical axis
  - World point \((X,Y,Z)\)

- **Image 2**
  - Right lens center
  - \((x_2, y_2)\)

- **World point**
  - \((X,Y,Z)\)

- **Left lens center**
  - \((x_1,y_1)\)
Stereo Imaging: How we get depth information from 2 eyes (cont.)

**Problem:** we know camera coordinates of the object on left and right image planes \((x_1, y_1)\) and \((x_2, y_2)\) and want to how far from the camera the object is located.

**Note:** when y-axis is parallel to the ground, we have \(y_1 = y_2\)
Stereo Imaging: How we get depth information from 2 eyes (cont.)

1. From the inverse perspective transform, we compute $X_1$ and $X_2$:

$$X_1 = \frac{x_1}{\lambda} (\lambda - Z_1) \quad \text{and} \quad X_2 = \frac{x_2}{\lambda} (\lambda - Z_2)$$

2. $Z_1$ and $Z_2$ must be equal, we get $Z_1 = Z_2$

3. Since left and right lenses are separated by distance $B$, we have

$$X_2 = X_1 + B$$

4. From 1, 2 and 3, we get

$$X_1 = \frac{x_1}{\lambda} (\lambda - Z) \quad \text{and} \quad X_1 + B = \frac{x_2}{\lambda} (\lambda - Z)$$

Solving $Z$ yields

$$Z = \lambda - \frac{\lambda B}{x_2 - x_1}$$
**Stereo Imaging: How we get depth information from 2 eyes (cont.)**

We can locate the object if we know positions of the object in left and right image planes using Equation:

\[ Z = \lambda - \frac{\lambda B}{x_2 - x_1} \]

**Question:** While the equation is so simple but why it is very difficult to build an automatic stereo vision system that can reconstruct 3-D scene from images obtained from 2 cameras?

**Answers:** for a computer, locating the corresponding points on left and right images is the most difficult task.
Imaging Geometry : Affine Transformations

1. Translation

2. Scaling

3. Rotating
**Image Geometry: Translation of Object**

Displace the object by vector \((X_0, Y_0, Z_0)\) with respect to its old position.

\[
\begin{align*}
X^* &= X + X_0 \\
Y^* &= Y + Y_0 \\
Z^* &= Z + Z_0
\end{align*}
\]

\[
\begin{bmatrix}
X^* \\
Y^* \\
Z^*
\end{bmatrix} =
\begin{bmatrix}
1 & 0 & 0 & X_0 \\
0 & 1 & 0 & Y_0 \\
0 & 0 & 1 & Z_0 \\
1 & 0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
X \\
Y \\
Z \\
1
\end{bmatrix}
\]
**Image Geometry: Translation of Frame**

Translate the origin point of the frame by \((X_0, Y_0, Z_0)\) with respect to the old frame.

\[
\begin{align*}
X^* &= X - X_0 \\
Y^* &= Y - Y_0 \\
Z^* &= Z - Z_0
\end{align*}
\]

\[
\begin{bmatrix}
X^* \\
Y^* \\
Z^*
\end{bmatrix} =
\begin{bmatrix}
1 & 0 & 0 & -X_0 \\
0 & 1 & 0 & -Y_0 \\
0 & 0 & 1 & -Z_0
\end{bmatrix}
\begin{bmatrix}
X \\
Y \\
Z \\
1
\end{bmatrix}
\]

The object still stays at the same position. Only the frame is moved.
Image Geometry: Scaling

Scale by factors $S_x$, $S_y$, $S_z$ along $X$, $Y$, and $Z$ axes.

$$X^* = S_x X$$
$$Y^* = S_y Y$$
$$Z^* = S_z Z$$

\[
\begin{bmatrix}
X^* \\
Y^* \\
Z^* \\
1
\end{bmatrix} =
\begin{bmatrix}
S_x & 0 & 0 & 0 & X \\
0 & S_y & 0 & 0 & Y \\
0 & 0 & S_z & 0 & Z \\
1 & 0 & 0 & 0 & 1
\end{bmatrix}
\]

Note: Origin point is unchanged.
**Image Geometry: Rotating an object about X-axis**

Rotate an object about \( X \)-axis by \( \theta_x \) in a counterclockwise direction.

\[
\begin{bmatrix}
    X^* \\
    Y^* \\
    Z^*
\end{bmatrix}
= \begin{bmatrix}
    1 & 0 & 0 & 0 & 0 \\
    0 & \cos \theta_x & -\sin \theta_x & 0 & 0 \\
    0 & \sin \theta_x & \cos \theta_x & 0 & 0 \\
    1 & 0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
    X \\
    Y \\
    Z \\
    1
\end{bmatrix}
\]

Note: In this case the object is moved. Only \( y \) and \( z \) are changed while \( x \) stills the same.
Image Geometry: Rotating a frame about X-axis

Rotate the frame about X-axis by $\theta_x$ in a counterclockwise direction.

$$
\begin{bmatrix}
X^* \\
Y^* \\
Z^*
\end{bmatrix} =
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & \cos \theta_x & \sin \theta_x & 0 & 0 \\
0 & -\sin \theta_x & \cos \theta_x & 0 & 0 \\
1 & 0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
X \\
Y \\
Z
\end{bmatrix}
$$

Note: In this case the object is not moved. The frame is rotated instead.
**Image Geometry: Rotating an object about Y-axis**

Rotate an object about $Y$-axis by $\theta_y$ in a counterclockwise direction.

$$
\begin{bmatrix}
X^* \\
Y^* \\
Z^*
\end{bmatrix} = \begin{bmatrix}
\cos \theta_y & 0 & \sin \theta_y & 0 \\
0 & 1 & 0 & 0 \\
-\sin \theta_y & 0 & \cos \theta_y & 0 \\
1 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
X \\
Y \\
Z \\
1
\end{bmatrix}
$$

Note: In this case the object is moved. Only $x$ and $z$ are changed while $y$ stills the same.
**Image Geometry: Rotating a frame about Y-axis**

Rotate the frame about Y-axis by $\theta_y$ in a counterclockwise direction.

\[
\begin{bmatrix}
X^* \\
Y^* \\
Z^*
\end{bmatrix} =
\begin{bmatrix}
\cos \theta_y & 0 & -\sin \theta_y & 0 \\
0 & 1 & 0 & 0 \\
\sin \theta_y & 0 & \cos \theta_y & 0 \\
1 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
X \\
Y \\
Z \\
1
\end{bmatrix}
\]

Note: In this case the object is not moved. The frame is rotated instead.
Image Geometry: Rotating an object about Z-axis

Rotate an object about Z-axis by $\theta_z$ in a counterclockwise direction.

\[
\begin{bmatrix}
X^* \\
Y^* \\
Z^*
\end{bmatrix} = \begin{bmatrix}
\cos \theta_z & -\sin \theta_z & 0 & 0 \\
\sin \theta_z & \cos \theta_z & 0 & 0 \\
0 & 0 & 1 & 0 \\
1 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
X \\
Y \\
Z \\
1
\end{bmatrix}
\]

Note: In this case the object is moved. Only $x$ and $y$ are changed while $z$ stills the same.
**Image Geometry: Rotating a frame about Z-axis**

Rotate the frame about Z-axis by $\theta_z$ in a counterclockwise direction.

$$\begin{bmatrix} X^* \\ Y^* \\ Z^* \end{bmatrix} = \begin{bmatrix} \cos \theta_z & \sin \theta_z & 0 & 0 \\ -\sin \theta_z & \cos \theta_z & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

Note: In this case the object is not moved. The frame is rotated instead.
**Image Geometry: How to compute a point on an image plane from the world coordinate**

**Problem:** we know the location of the object and want to know where it will be projected on the film (image plane).

**Answer:**
1. Transform the world coordinate to the camera coordinate
2. Perform the perspective transformation
Before using the perspective transformation, the world axes $X$-$Y$-$Z$ must coincide with the camera axes $x$-$y$-$z$, (we need some transformations).

$(x,y,z) = \text{Camera coordinate}, \ (X,Y,Z) = \text{World coordinate}$
Image Geometry: Compute the camera coordinate from the world coordinate

Steps from Gonzalez’s book

1. Translate* by \( w_0 \)
2. Pan the camera* (rotate about Z-axis)
3. Tilt the camera* (rotate about X-axis)
4. Translate* by \( z = r \)
5. Compute the perspective Tr.

Note: *perform on the frame
Image Geometry: Compute the camera coordinate from the world coordinate (cont.)

Formula from Gonzalez’s book

Camera coordinate \( c_h = P \cdot C \cdot R \cdot G \cdot w_h \)

- Perspective tr.
- Translate to the image plane
- Center by \( z = r \)
- Translate to the gimbal center \( w_0 \)
- Rotate by Pan \( (\theta_z) \)
- and Tilt \( (\theta_x) \)
Image Geometry: Compute the camera coordinate from the world coordinate (cont.)

General case
1. Translate* by \( w_1 \)
2. Pan the camera*
3. Tilt the camera*
4. Twist the camera* (rotate about \( Y \)-axis)
5. Compute the perspective Tr.

Note: *perform on the frame
Image Geometry: Compute the camera coordinate from the world coordinate (cont.)

General case

Camera coordinate

\[ c_h = P \cdot R \cdot T \cdot w_h \]

Perspective tr.

Rotate by
- Pan \((\theta_z)\),
- Tilt \((\theta_x)\),
- Twist \((\theta_y)\)

Translate to the image plane center \(w_I\)