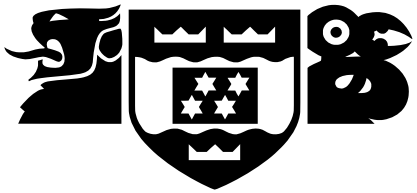
VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui



School of Engineering and Computer Science

COMP 422 — Weeks 2-3

Computer Vision and Image Processing — Introduction

Mengjie Zhang

mengjie@ecs.vuw.ac.nz

Outline

- Overview – Computer Imaging
- Computer Vision and its applications
- Image Processing and its applications
- Image acquisition and display
- Image representation
- Image file formats
- Tools
- Image implementation

Announcements

- **netpbm/pbmplus** package:
  `/vol/courses/comp422/src/netpbm/`
- **jpeg/jpg** package:
  `/vol/courses/comp422/src/jpeg-6b/`
- An implementation of *pgm* image representation source code:
  `/vol/courses/comp422/src/mengjie/pgm-rep/`
- Three programs making a PGM image, circles and squares in an image: `/vol/courses/comp422/bin/`

Questions

- In what kinds of situation computers are used to process/analyse images/pictures? Give some examples.
- What are the differences between computer vision and image processing? Are the images examined/acted on by people or automatically by computers?
- How do you acquire/obtain an image? Give some examples of image acquisition device.
- How can we display an image? Give some examples of image display device.
- Have you used digital cameras and scanners before?
- What types can images be represented?
- What image formats have you seen before?
- What image packages have you used before?
- How do we use current packages to process images?

Goals

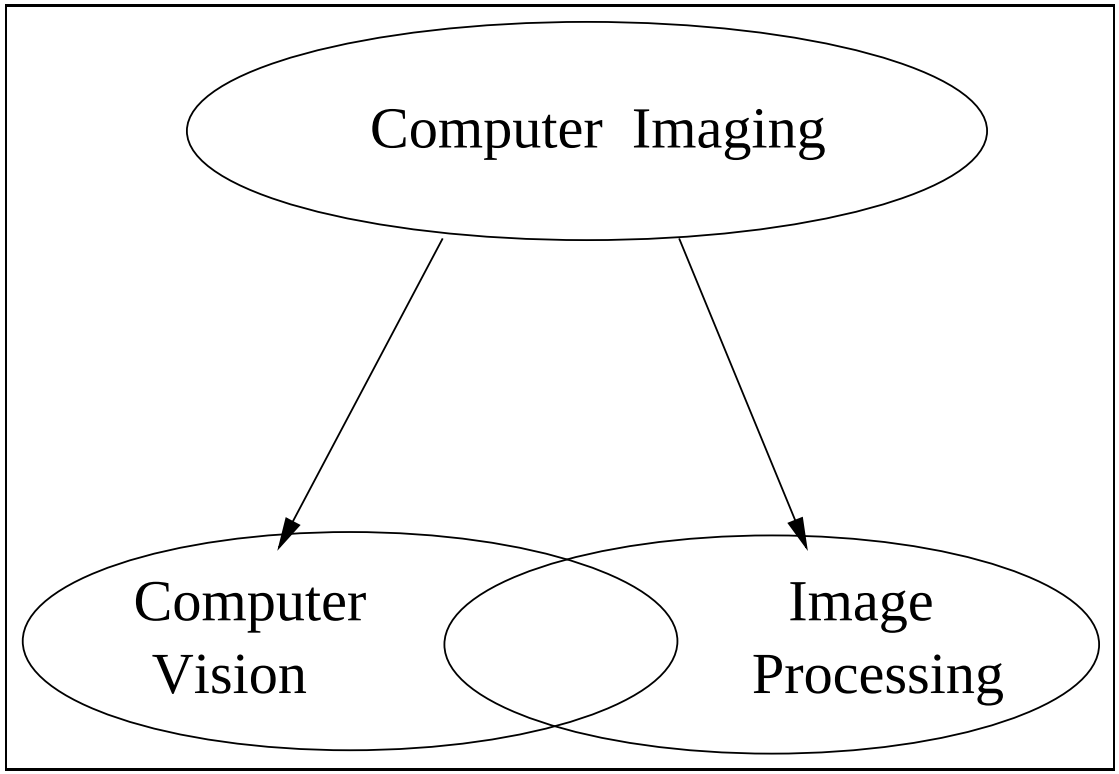
The main objectives of this topic – Introduction to computer vision and image processing are:

1. Give an overview of concepts and applications of computer vision and image processing.
2. Discuss basic algorithms and techniques for image analysis.
3. Understand image representations and image formats, and implementation of images.
4. Understand and learn commonly used image feature types and feature extraction methods.
5. Perform pattern classification on vision problems – data mining in image data – object classification.

Overview: Computer Imaging

- Computer imaging is needed: Send/receive complex data; A picture is worth a thousand words (in some cases); WWW, Visual information, Multimedia.
- Computer imaging can be defined as the acquisition and processing of visual information by computer. [Scott Umbaugh]
- Two categories:
  1. Processed images are for use by a computer – Vision applications
  2. Output images are for use by people – image processing applications.

Computer Imaging (Continued)



Computer Imaging (Continued)

Note: The boundaries separating these two are not very clear, or fuzzy.

- *Historically*, IP grew from electrical engineering as an extension of signal processing, while computer vision was largely derived from computer science discipline.
- *Recently*, the two groups have come together to create modern computer imaging. They share some basic processing algorithm and methods.
- They are often called together: "Image processing and computer vision". For example, the international conference on image processing and computer vision is annually held in different places of the world.

Computer Vision

- Computer vision deals with the processing of image data for use by a computer.
- In other words, the images are examined and acted upon by a computer and the application does not involve a human being in the visual loop.
- Note: People can participate the development of a vision system, however, a computer can directly apply the patterns/rules (knowledge) extracted by the system (to the unseen data).

Computer Vision (continued)

A major topic in computer vision is *Image Analysis*, which involves the examination of the image data for solving a vision problem. Typically, this includes:

- Edge Detection
- Segmentation
- Transformation
- Feature extraction
- Pattern (object) classification

These topics will be discussed in somewhat detail later.

Vision Applications

- Mechanical and manufacturing Engineering
  - Quality control (Robot)
  - Machinery Fault diagnosing system
- Medical community
  - Skin tumor diagnosis
  - Brain surgery aided system
  - Automatic clinic test system (ES)
  - Tissue and cell analysis (identification)

Vision Applications (Continued)

- Law and enforcement and security
  - Fingerprint identification
  - DNA analysis
  - Highway speed monitoring
- Object Recognition/Detection
  - Autonomous vehicle recognition
  - Vehicle tracking and identification
  - Cyclone finding in satellite images
  - Mine/Mine-like target detection
- Weather prediction

Image Processing

- Image processing involve the manipulation of image data for viewing by people.
- In other words, the images are examined and acted on by people in IP applications.
- Major topics:
  - Image restoration
  - Image enhancement
  - Image compression

Image Restoration

- Image restoration is the process of taking an image with some known or estimated degradation, and restoring it to its original appearance.
- It is often used in the situation that an image was somehow degraded but needs to be improved before the image can be reused. Publishing is such an example.
- To restore an image to its original appearance, the process of the degradation (such as a model) is often necessary.

Image Restoration Example

A typical example for image restoration in space exploration to eliminate artifacts generated by mechanical jitter in a spacecraft is:

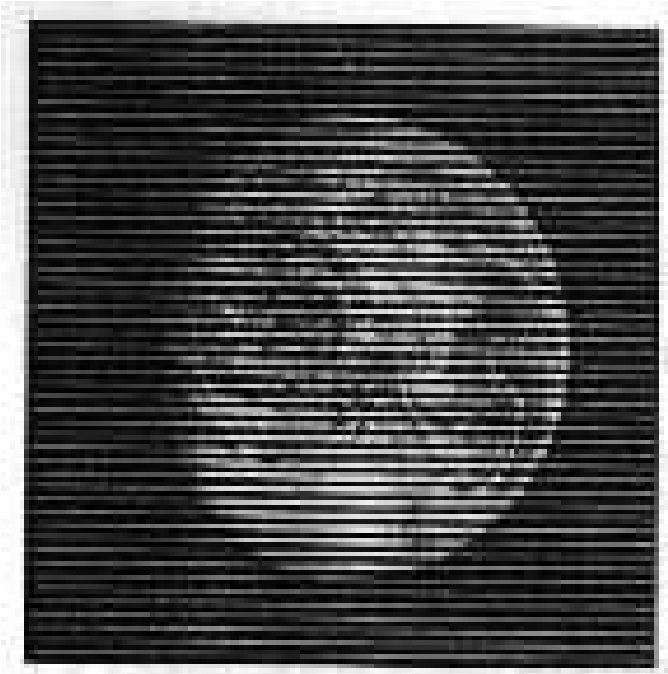


Image with distortion

Restored image

Image Enhancement

- Image enhancement involves taking an image and improving it visually, typically by taking advantage of human visual system's response.
- One typical image enhancement technique is *contrast stretching*, where the contrast of an image is simply stretched.
- Image enhancement methods are often domain dependent.
- Image enhancement vs image restoration

Image Enhancement Example

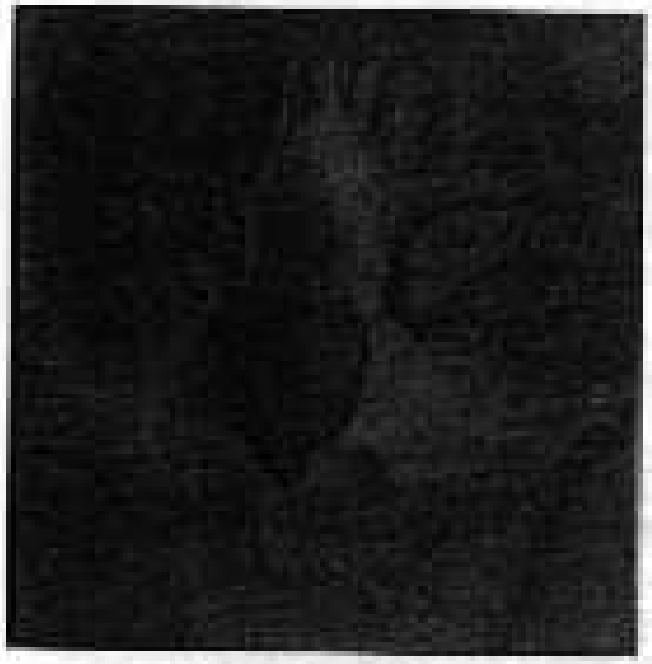


Image with poor contrast

Enhanced image

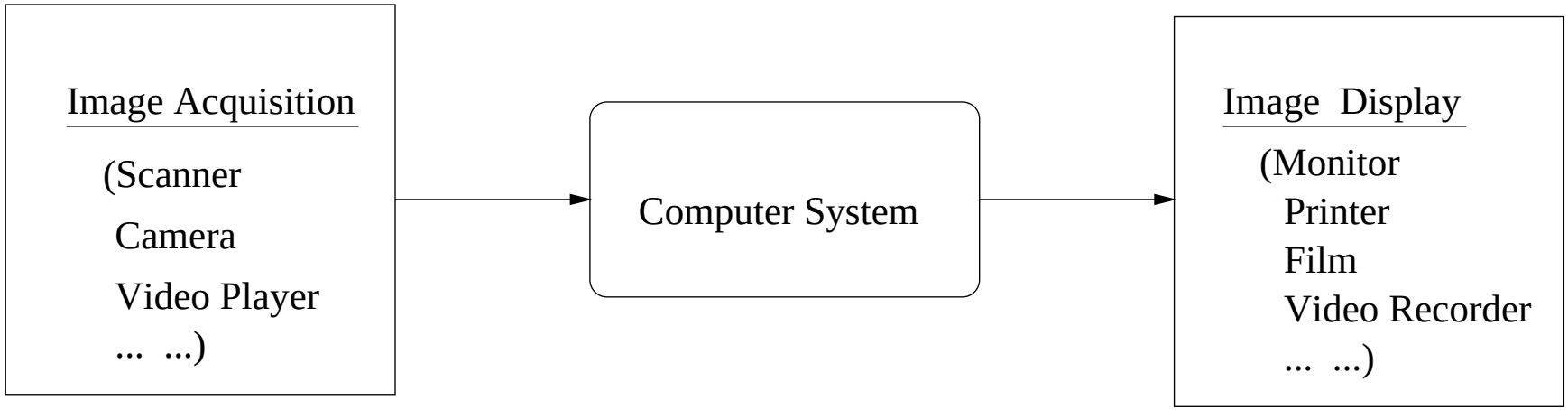
Image Compression

- Image compression involves reducing the typically massive amount of data needed to represent an image.
- Compression procedure only (often) eliminates data that are visually unnecessary and takes advantage of the redundancy that is inherent in most images.
- Some images can be reduced 50 times and some even more.

IP Applications

- Medical community in Diagnostic imaging: Computerized Tomograph (CT), Magnetic Resonance Imaging (MRI) scanning, Positron Emission Tomography (PET).
- Biological research: Enhance microscopic images to get more features.
- Entertainment industry: Editing, creating artificial scenes and beings; Processing new haircut styles, eyeglasses.
- Computer-aided design

Image Acquisition and Display



- Image acquisition device: Scanners, Cameras, Vedio Players, ...
- Image storage: Films, files, ...
- Image display device: Monitors, printers, films, video recorders, ...
- Computer imaging system: Process saved images.

Scanners

- Different types: flat-bed or drum
- 600×300 to 2000×2000 per inch
- Used when source images are already in hard copy form.
  - photos or radiographs
  - drawing

Cameras

- TV cameras: CCTV, Broadcast
- Digital cameras: from 200×200 to 4096×4096
  - Digital still cameras: Limited resolution but avoid the use of film for simple applications; downloading to computers is slow
  - High quality digital cameras: up to 4096×4096, can have 8, 10, 12 bits; require special interfacing
- Line-scan cameras: randomly addressable CCD
- Various spectral sensitivities: infrared, visible, ultra violet — X rays.

Issues in Image Output

- Resolution: Spatial (say 1280×1024 pixels on a monitor), color (16, 256,...), time (speed, 25, 500, 10^7 frames per second on TV)
- Dynamic range: Black (better on paper than a monitor screen), white (better on transparencies), Possible range of color, ...
- Accuracy

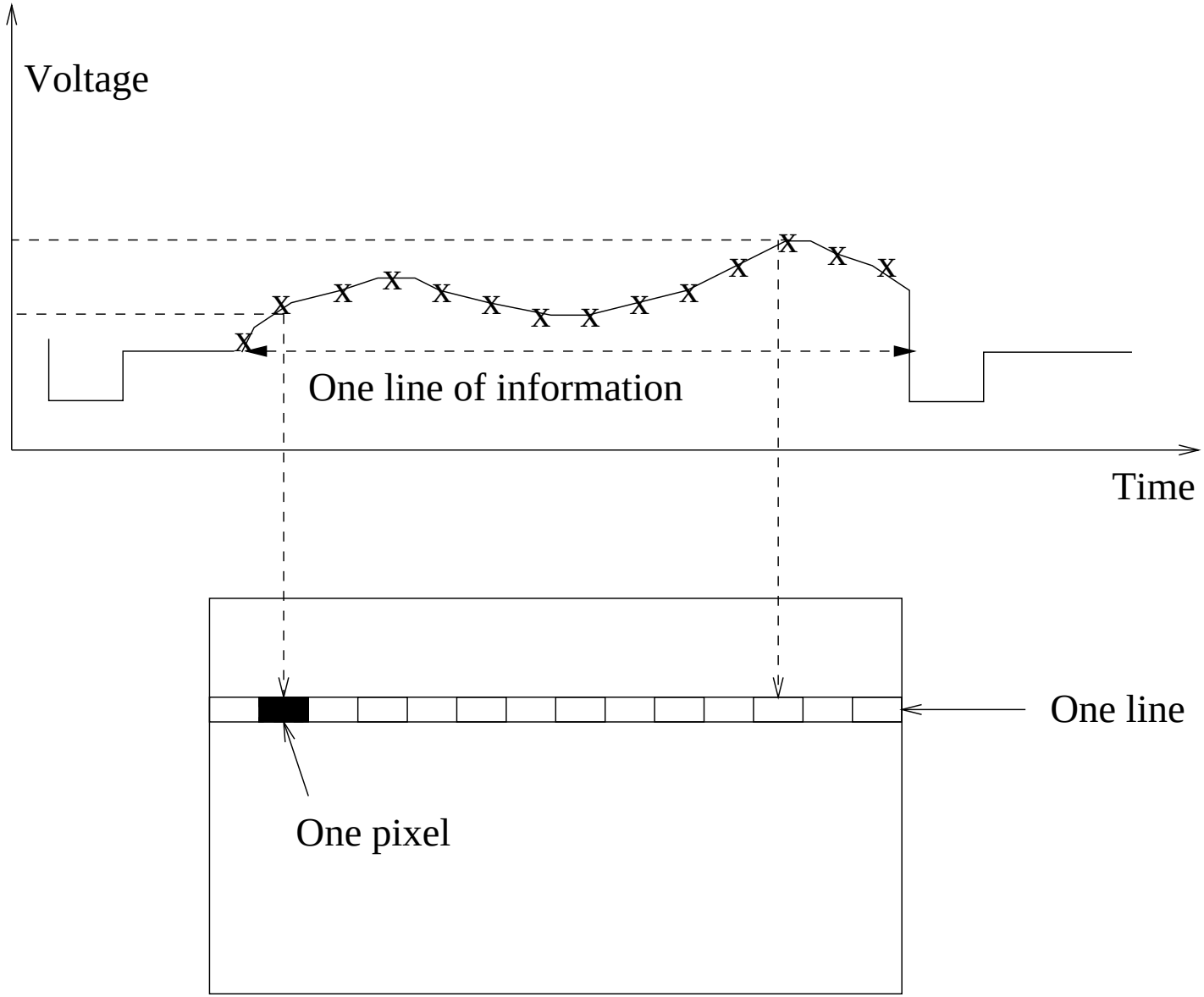
From Analog Video Signal to Digital Image

- Frame Grabber: A special-purpose hardware which accepts a standard video signal and produces an image in the form that a computer can understand.
- Digital image: The form of an image which a computer can understand is called a digital image.
- Video signal is continuous, while a digital image is discrete.
- Image Digitization: the process of transforming a standard video signal into a digital image is called image digitization or simply digitization.

Digitization (Continued)

- Digitization is done by sampling the continuous signal at a fixed rate.
- One line of a video signal is often digitized by instantaneously measuring the *voltage* of the signal at fixed intervals in time.
- The value of the voltage at each instant is converted into a number corresponding the brightness of the image at that point.
- The brightness of the image at one point is often called the "value" of that *pixel*.
- A pixel is a "point" in an image.
- An image is often accessed as a two dimensional array, in a form of $column \times row$. For example, 1024×768 pixels.

Digitization (Continued)



Hierarchical Image Pyramid

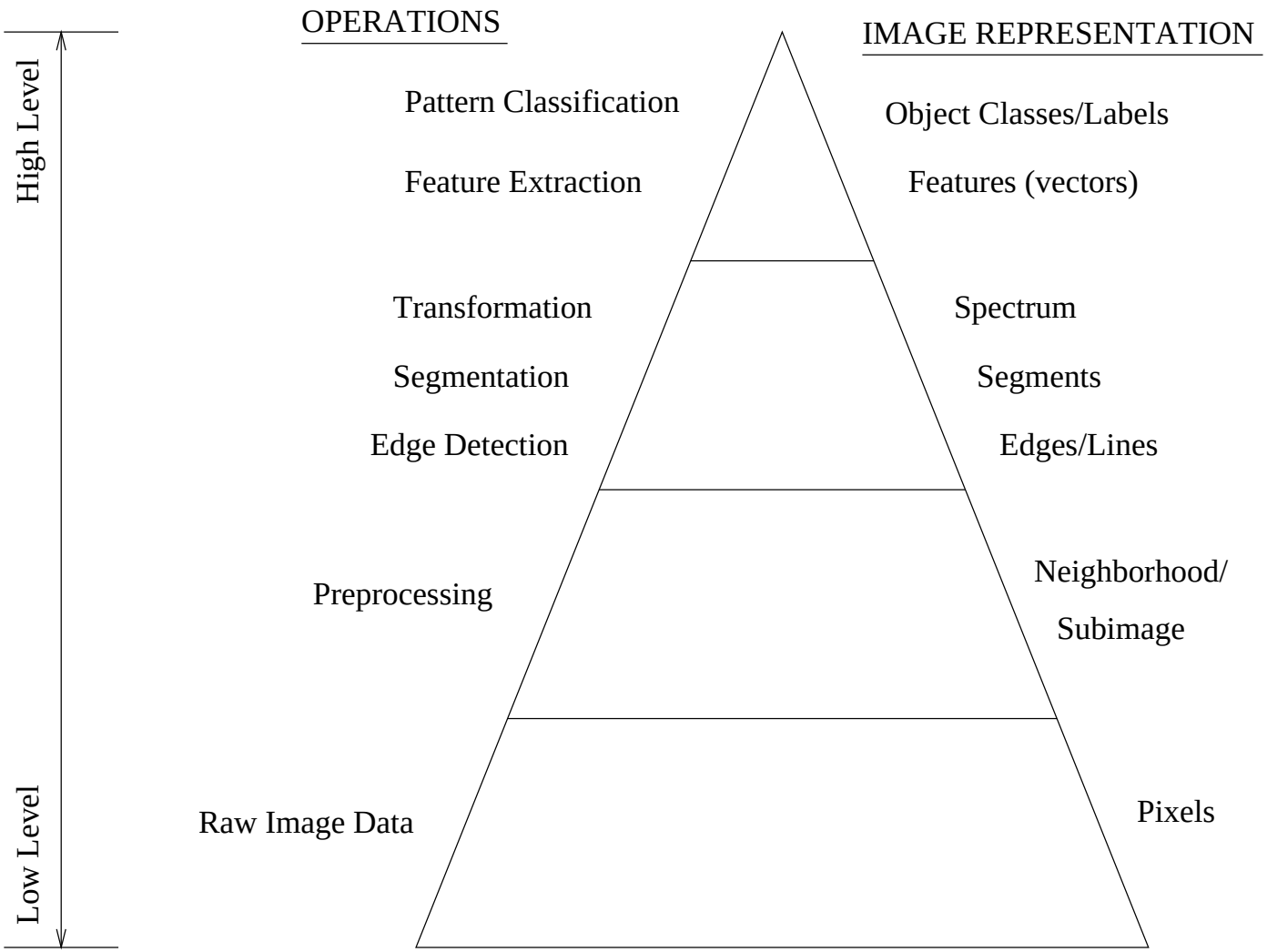


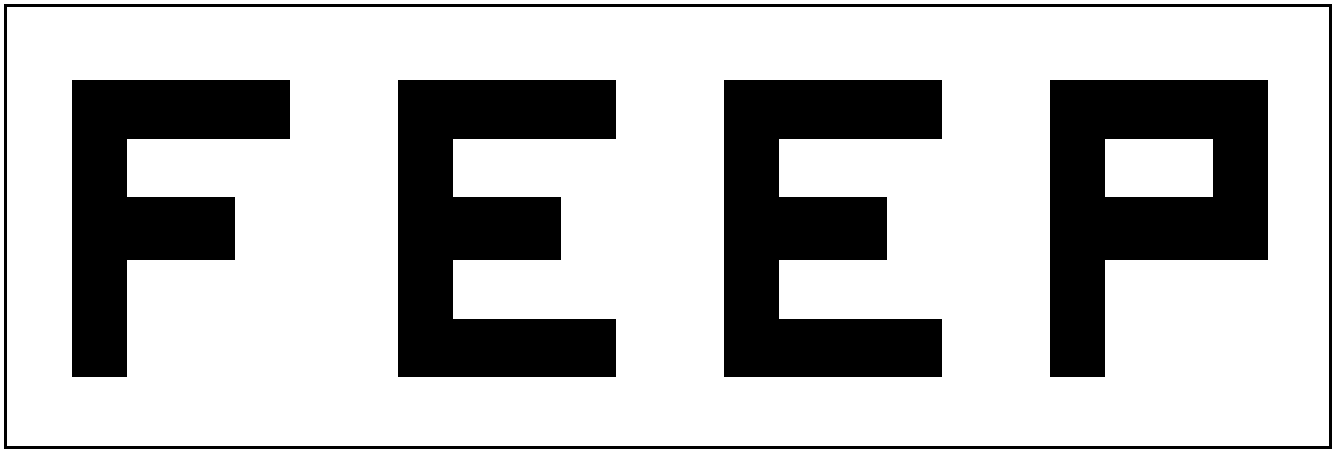
Image Representation

- From now on, without specific notice, images refer to digital images.
- Images can be represented as an array, or a matrix. Each row corresponds to a *vector*.
- A *pixel* at position (x, y) corresponds to the "brightness" of the image at the position.
- There are four basic representations for images:
  - Binary images
  - Gray scale images
  - color images
  - multispectral

Binary Images

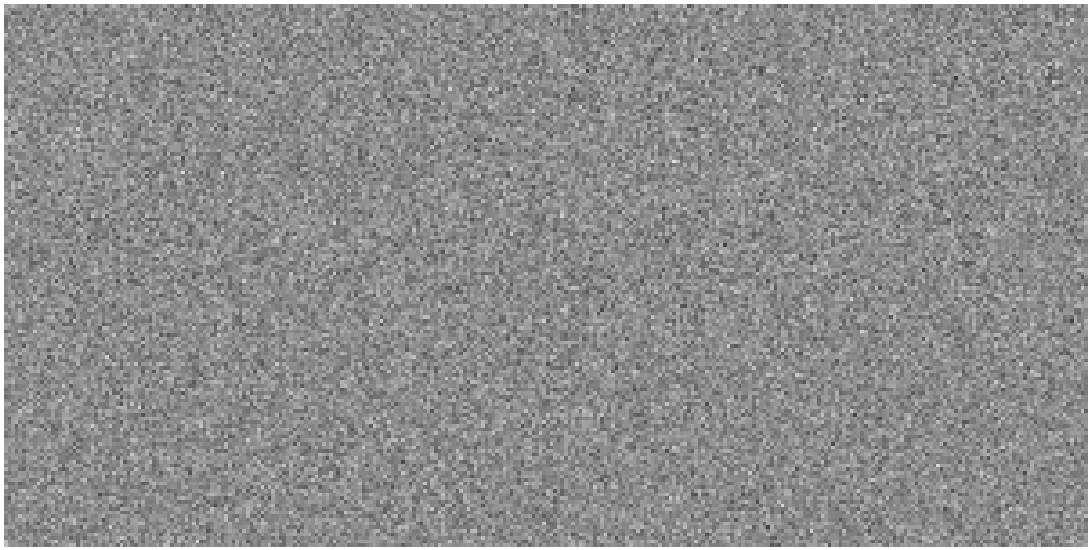
- Binary is the simplest type. It refers to *black* and *white* image.
- In other words, there are only two values: 0 for black; 1 for white, or *one bit* data, for each pixel in the image.
- These images are often used in the situation that only the general shape, outline or position information are needed. For example, optical character recognition, robotic gripper to grasp an object.
- It can be obtained from other formats, particularly from grey scale format by applying a single threshold.

Sample Images



Grey Scale Images

- Grey scale images only contain brightness information. Compared with binary images, they contain richer information.
- Typically, grey scale images contain 8 bits data. The range of pixel values is from 0 to 255. In other words, they have 256 grey levels.
- These images can provide some sorts of noise.

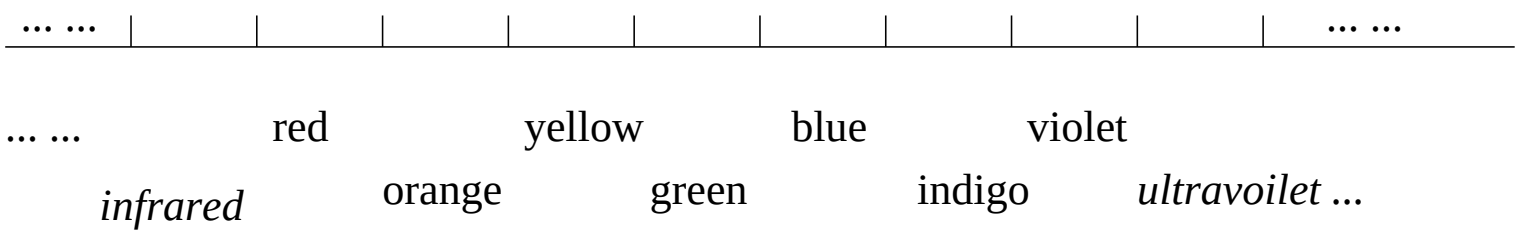


Color Images

- Color images have three-band monochrome image data, each of which corresponds to a different color.
- Typically, the three colors in color images are *red*, *green* and *Blue*, or RGB.
- Each of the three colors contain 8 bit data. In total, RBG images have 24 bit pixel data.
- At each position in a RGB image, a pixel corresponds to a color pixel vector – R, G, B.

Multispectral Images

- Miltispectral images typically contain information outside the human perceptual range. They contain more (30) bands.
- These images are not directly visible to human, however, they can be reduced into three bands RGB images, which human being can see.
- These images might include infrared, ultraviolet data.
- Satellite images, underwater sonar images, radar images, infrared images, medical X-ray images are examples.



"Colors" of the spectrum

Image File Formats

- Many image formats exists: At least 200 image formats in use
- Most image files contain header information and the raw pixel data.
- Image header information usually contain some of the following information:
  - Number of rows (*height*)
  - Number of columns (*width*)
  - Number of bands
  - Number of bits per pixel
  - file type
- We only review those commonly used

Image File Formats (Continued)

- BMP format (bitmap)
  - often have the extensions .bmp, .dib, .vga, .bga, .rle, .rl4, .rl8, ...
  - 1, 4 and 8 bit indexed color, 24 bit RBG color
- PNM format (Portable anymap file format)
  - pbm format (Portable bitmap file format, .pbm)
  - pgm format (Portable graymap file format, .pgm)
  - ppm format (portable pixmap file format, .ppm)
- TIFF format (Tagged Image File Format): 1, 4, 8, 32 bit indexed color; 24 bit RGB color; 32 bit RGB + alpha. (Often have the extension .tif)

Image File Formats (Continued)

- GIF (Graphics Interchange Format): 1,8 bits per pixel indexed color; always compressed using LZW (Lempel-Ziv-Welch, lossless); have the extension .gif.
- JPEG (Joint Photographic Experts Groups): Compressed; commonly used in HTML and World wide web (WWW); have the extension of .jpg or .jpeg.
- Others
  - SGI (Silicon Graphics, Inc.)
  - EPS (Encapsulated PostScript)
  - VIP (Visualization in Image Processing)

A Typical PGM Format Image

```
P2
# feep.pgm
24 7
15
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  3  3  3  3  0  0  7  7  7  7  0  0 11 11 11 11  0  0 15 15 15 15  0
0  3  0  0  0  0  0  7  0  0  0  0  0 11  0  0  0  0  0 15  0  0 15  0
0  3  3  3  0  0  0  7  7  7  0  0  0 11 11 11  0  0  0 15 15 15 15  0
0  3  0  0  0  0  0  7  0  0  0  0  0 11  0  0  0  0  0 15  0  0  0  0
0  3  0  0  0  0  0  7  7  7  7  0  0 11 11 11 11  0  0 15  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

A Typical PGM Format Image (Continued)

- A *magic number* for identifying the file type. A pgm file's magic number is the two characters "P2".
- A *width*, formatted as ASCII characters in decimal. (e.g. 24)
- A *height*, in ASCII decimal (e.g. 7)
- The maximum gray value, again in ASCII decimal (e.g. 15).
- *Width*$\times$*height* gray values, each in ASCII decimal, between 0 and the specified maximum value, separated by whitespace. A value of 0 means black, and the maximum value means white.
- Characters from a "#" to the next end-of-line are ignored (comments).

P5 Format PGM images

P5 is a variation of *P2* format. This format saves PGM images in a *RAWBITS* or *binary* way. Ii is different from P2 format in the following aspects:

- The magic number is *P5* instead of *P2*.
- The gray values are stored as plain bytes, instead of ASCII decimal.
- In the grays section, a newline character (typically) is set after the maxval (each row).
- The files are smaller and many times faster to read and write.
- This format can only be used for maxvals less than or equal to 255. If maxval is larger, it will automatically fall back on the plain format.

Programming with Images

- Loading a image
- Creating an image
- Processing an image
- Outputting/saving an image
- Destroying an image

Programming with Images (Continued)

```
#include <stdio.h>
#include <stdlib.h>

/* Macros and definitions */
#define TRUE    1
#define FALSE   0
#define OK      1

typedef unsigned char UCHAR;

typedef struct _PGMIMAGE /* Structure for containing a PGM format image */
{
    int greylevel;
    int xsize;
    int ysize;
    UCHAR *image;
} PGMIMAGE;
```

Programming with Images (Continued)

```
/*
 *  CreatePGM() -- Creates a PGM image structure in memory
 */

PGMIMAGE *CreatePGM(int xsize, int ysize, int greylevel)
{
    PGMIMAGE *pgm;

    if ((pgm = (PGMIMAGE *) malloc(sizeof(PGMIMAGE))) == NULL) return NULL;

    pgm->greylevel = greylevel;
    pgm->xsize = xsize;
    pgm->ysize = ysize;

    if ((pgm->image = (UCHAR *) malloc(sizeof(UCHAR) * xsize * ysize)) == NULL)
    {
        free(pgm);
        return NULL;
    }

    memset(pgm->image, 0, xsize * ysize);
    return pgm;
}
```

Programming with Images (Continued)

```
/* LoadPGM() -- Loads a PGM format image file into memory */

PGMIMAGE *LoadPGM(char *filename)
{
    FILE *fp;
    PGMIMAGE *pgm;
    int i;

    if ((fp = fopen(filename, "rb")) == NULL) return NULL;

    if ((pgm = (PGMIMAGE *) malloc(sizeof(PGMIMAGE))) == NULL)
    {
        fclose(fp);
        return NULL;
    }
    fscanf(fp, "P5 %d %d %d\n", &pgm->xsize,
           &pgm->ysize, &pgm->greylevel);

    if ((pgm->image = (UCHAR *) malloc(sizeof(UCHAR) * pgm->xsiz
           pgm->ysize)) == NULL)
    {
        free(pgm);
        fclose(fp);
        return NULL;
    }

    for (i = 0; i < pgm->xsize * pgm->ysize; i++)
    {
        pgm->image[i] = (UCHAR) fgetc(fp);
    }
    return pgm;
}
```

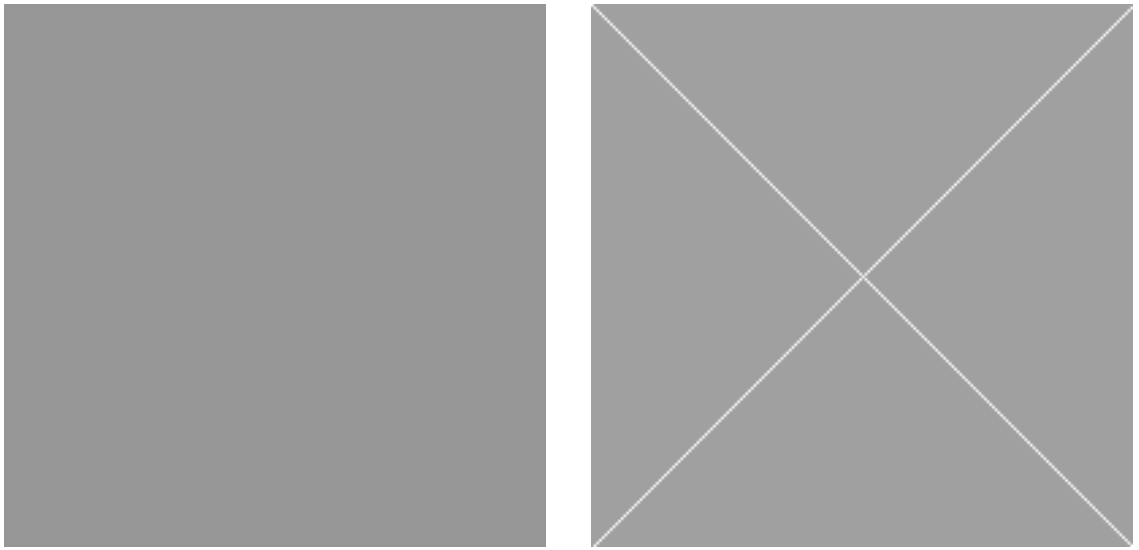
Programming with Images (Continued)

```
/* SavePGM() -- Save a PGM format image to a file */
int SavePGM(char *filename, PGMIMAGE *pgm)
{
    FILE *fp;
    int i;

    if ((fp = fopen(filename, "wb")) == NULL) return (int) NULL;

    fprintf(fp, "P5\n %d %d %d\n", pgm->xsize, pgm->ysize, pgm->greylevel);

    for (i = 0; i < pgm->xsize * pgm->ysize; i++)
    {
        fputc(pgm->image[i], fp);
    }

    fclose(fp);

    return OK;
}

/* DestroyPGM() -- Destroys a PGM format image in memory*/
void DestroyPGM(PGMIMAGE *pgm)
{
    free(pgm->image);
    free(pgm);
}
```

Programming with Images(Continued)

An example: process.c

- This program shows how to process a PGM (P5) format image.
- It loads an original image (source), draws white lines on diagonals (image processing), then save the results to a new image (target).



Programming with Images (Continued)

```
#include "pgm_mengjie.h"
typedef UCHAR byte;

int main(int argc, char *argv[])
{
   PGMIMAGE *source, *target;
   int i, row, col, xsize, ysize, greylevel;
   byte **temp;  /* int **temp; is also ok. For temporary storage */

   if (argc != 3)
   {
     fprintf(stderr, "\nUsage: %s [source-image] [target-image]\n", argv[0]);
     exit(0);
   }
   if ((source = LoadPGM(argv[1])) == NULL)
   {
      fprintf(stderr, "Error on reading %s\n", argv[1]);
      exit(0);
   }

   xsize = source->xsize;
   ysize = source->ysize;
   greylevel = source->greylevel;
```

Programming with Images(Continued)

```
/* Create an image for output/save */
if ((target = CreatePGM(xsize, ysize, greylevel)) == NULL)
{
   fprintf(stderr, "Not enough memory to create an image!\n");
   exit(0);
};

/* allocate memory for the temporary array for processing image(s) */
temp = calloc(xsize, sizeof(byte *));
if (temp == NULL)
{
   fprintf(stderr, "Not enough memory to create temp[]\n");
   exit(0);
}
for (i = 0; i < ysize; i++)
{
   temp[i] = calloc(xsize, sizeof(byte));
   if (temp[i] == NULL)
   {
      fprintf(stderr, "Not enough memory to create temp[][]\n");
      exit(0);
   }
}
```

Programming with Images (Continued)

```
/* Put the source image data to the temporary array */
for (row = 0; row < ysize; row++)
   for (col = 0; col < xsize; col++)
       temp[row][col] = (byte) source->image[xsize * row + col];

/* Image processing: draw white lines on the diagonals,
   add 10 to other pixels */
for (row = 0; row < ysize; row++)
{
   for (col = 0; col < xsize; col++)
   {
      if (col == row || col == xsize - row)
         temp[row][col] = greylevel;
      else
         temp[row][col] += 10;
   }
}

/* Assign the processing results to the target image */
for (row = 0; row < ysize; row++)
   for (col = 0; col < xsize; col++)
      target->image[xsize * row + col] = (byte) temp[row][col];
```

Programming with Images (Continued)

```
   /* save the target image */
   SavePGM(argv[2], target);

   fprintf(stderr, "The process is completed... \n");

   DestroyPGM(source);
   DestroyPGM(target);

   return 0;
}
```

Image Programs/Packages/Tools

- xv
- ImgStar
- pbmplus/netpbm
- jpeg-6b(jpegtran)
- **man** pnm, pgm, ppm, ...

Image Programs/Packages/Tools (Continued)

For example, `% man pnm`:

```
pnm(5)                                                    pnm(5)

NAME
       pnm - portable anymap file format

DESCRIPTION
       The  pnm  programs  operate on portable bitmaps, graymaps,
       and pixmaps, produced by the pbm, pgm, and  ppm  segments.
       There is no file format associated with pnm itself.

SEE ALSO
       anytopnm(1), rasttopnm(1), tifftopnm(1), xwdtopnm(1), pnm-
       tops(1),    pnmtorast(1),    pnmtotiff(1),    pnmtoxwd(1),
       pnmarith(1),  pnmcat(1),  pnmconvol(1),  pnmcrop(1),  pnm-
       cut(1),  pnmdepth(1),  pnmenlarge(1),   pnmfile(1),   pnm-
       flip(1),  pnmgamma(1),  pnmindex(1), pnminvert(1), pnmmar-
       gin(1),  pnmnoraw(1),  pnmpaste(1),   pnmrotate(1),   pnm-
       scale(1),  pnmshear(1),  pnmsmooth(1), pnmtile(1), ppm(5),
       pgm(5), pbm(5)
```

Summary

- Computer imaging = CV + IP
- CV vs IP
- CV main tasks and applications
- IP main topics and applications
- Commonly used image acquisition and display device/systems
- Image digitization
- Image representation/types: binary, gray scale/level, color, multispectral ...
- Image formats: .pnm, .jpg/jpeg, .gif, .bmp, ...
- PNM: P2/P5—gray level
- Processing implementation: Defining, loading, creating, processing, saving, destroying, ...
- Tools: xv, pbmplus/netpbm, ImgStar, jpeg-6b, CVIP, ...

Exercises

- Use *cir_make*, *squ_make*, *pgmmake* programs to generate some pictures.
- Write a program that reverses a PGM image.
  ($I_{new}(r, c)$ = 255 - $I_{old}(r, c)$)
- Write a program that adds two PGM images and produces a third image as output. (using netpbm/pbmplus or the representation in .../public/src/mengjie/pgm-rep/)
- Convert an image from PGM format to GIF format and JPG format using programs in netpbm/pbmplus package and/or jpeg-6b package.

Questions for Next Discussion

- What tasks/stages are commonly included/used in image analysis?
- What is preprocessing for? Give some examples for preprocessing.
- What algebraic operations can be applied to images? Give some examples.
- Give examples for edge detection and segmentation.
- Image features, feature types, feature extraction
- Object classification methods — Simple DM algorithms