Practical applications of Image Processing
Contents

• Colourspace transformation
• Face Detection & Extraction
• Image Enhancement
• Steganography
• Image Standardization
Colourspace Transformation

- Code:
  - Read jellyfish.jpg to variable ‘a’
  - \texttt{b=rgb2ntsc(a)};
  - \texttt{Imshow(b)}
- \texttt{La^*b^*} space is not supported by all MATLAB versions thus NTSC is used
- Different image handling equipment understand different image color formats
Colourspace Transformation (contd.)
Colourspace Transformation (contd.)
Face Detection and Extraction

• Most common approach is to first assume that background is darker or lighter than skin tone
• Then find image threshold and convert to black and white
• For standard software i.e. face identification in banks, license offices etc., it is assumed that the face lies in the centre region of the image thus corners aren’t search i.e. background is reduced
• Try the code for image with multiple faces
Face Detection and Extraction (contd.)

• In given program, it is assumed that skin region is lighter and thus when converted to B&W, its marked by only white pixels with black holes for eyes.

• The face region can be marked by any standard figure like rect, circle etc.

• The holes are not considered.
Face Detection and Extraction (contd.)

Commands used->

- `Rgb2gray()` : converts rgb colour type image to grayscale
- `Im2bw()` : converts grayscale image to black and white choosing gray threshold by default
- `Floor()` : to round off to nearest integer value
- `Find()` : locate pixels or array values based on criteria given
  - Try thr following code:
    
    ```matlab
    a=magic(6)
    find(a>15)
    ```
  - Discuss the values obtained
Face Detection and Extraction (contd.)

• **Hold**: holds the current graph so that all computational additions are made to the same graph instead of plotting the figure again and again.

• **Bwlabel()**: gives label i.e. index of all connected components, we can take 4 or 8 components at a time, this command is used to separate the all-white skin region.

• **Regionprops()**: Converts the above labels to proper region, with struct type of format.
  - Since image can only be displayed in matrix type of format the above struct doesn’t work.
  - Consequently, struct should be converted to cell.
  - Cell should be converted to matrix to obtain image format.
Image Enhancement

- Histogram is a plot of total graylevels vs. no. of pixels having those gray levels
Image Enhancement (using histogram equalization)

- Algorithm
  - Import image ‘pout’ and display histogram to identify the kind of enhancement required from the histogram
  - Find the no. of pixels having each graylevel i.e. n(r)
  - Find the cumulative probability distribution of the above
  - Multiply cdf obtained by (total no. of graylevels-1) i.e. 255
  - Round off to the nearest integer, this gives new gray levels
  - Map the newer gray levels as per code
- Try implementing the same program for a 3D image i.e. color image and see if it works
Steganography

• The program uses Least Significant Bit steganography

• An image is nothing more than strings and strings of bytes, each byte representing a different color.

• The last few bits in a color byte, do not hold as much significance as the first few.

• This is to say that two bytes that only differ in the last few bits can represent two colors that are virtually indistinguishable to the human eye.
Steganography (contd.)

• For example, 00100110 and 00100111 can be two different shades of red, but since it is only the last bit that differs between the two, it is impossible to see the color difference.

• LSB steganography, then, alters these last bits by hiding a message within them. One way of doing this is as follows.

• Say I wished to hide the an alphabet in the image

• The last bit of every byte is replaced with the corresponding bit from the letter I (ASCII code)

• The new image now contains the desired text of the ‘I’ without degrading the quality of the image since only the least significant bits were altered
Steganography (contd.)

Exercise

• Try to separate the LSB bit plane from any grayscale image
• Take another grayscale image and equalize the size of both of them by padding zeros
• Convert the second image to binary type of image
• Hide second image in the LSB bit plane of first image
• Concatenate the two to achieve image LSB steganography
Image Standardization

- Face detection and extraction, which is a more common way of image standardization are used in most authentication applications used in government and private offices.
- A second type can be taken as RST i.e. Rotation, scaling, translation analysis of signature templates used for signature verification mostly in banks.
- When a person signs, two consecutive signatures from the same person are bound to differ in terms of size, rotation, and position from assumed origin.
About correlation

• It is a mathematical function that gives the level of similarity between two data table i.e. voice image etc.

• Adjacent image shows that when correction is complete, the correlation graph is maximum (peak value)
Image Standardization (RST correction)

Algorithm to correct rotation

- Obtain user image and the reference image.
- Carry out pre-processing by converting both images to grayscale and performing normalization.
- Trim the reference signature to remove any excess background; this will act as the template.
- Starting with the angle as $-60^\circ$, in increments of $5^\circ$, record normalized correlation values between pre-processed reference image and user image.
Image Standardization\(\text{(RST correction)}\)

Algorithm (contd.)

- If angle is less than or equal to 60°, go to step 4.
- Maximum correlation value corresponds to angle within a 5° range. Let this angular value be \(x°\).
- Starting with the angle as \((x - 3)°\), in increments of 1°, record normalized correlation values between the preprocessed reference image and the user image.
- If angle is less than or equal to \((x + 3)°\), go to step 7.
- Correct the user image by the obtained angle and proceed for further correction, if required.
Algorithm (contd.)

• Once rotation correction is performed crop additional background to get translation correction

• Simply resize the resulting image by scalefactor (ratio of size of original image/ratio of size of image after R and T correction) to correct for scaling
Thank You!

for queries contact
Aman Chadha 993 055 6583
Divya Jyoti 8879 100 684