Lab 7 of COMP 319

Instruction 1 to the second course project

Lab tutor:  Dennis Yang LIU
Email:  csygliu@comp.polyu.edu.hk
Lab 7:  Oct. 30, 2014
Summary of Lab 6

• General concepts of plotting
  - Plot container
  - Subplots

• Two-dimensional plotting
  - Simple 2-D plots
  - Parametric and other plots

• Three-dimensional plotting
  - Linear 3-D plots
  - Linear parametric and other 3-D plots
Outline of Lab 7

- Color image in Matlab
- DCT and IDCT in Matlab
- Sub-function definition
- Sub-block processing in Matlab
Outline of Lab 7

- Color image in Matlab
- DCT and IDCT in Matlab
- Sub-function definition
- Sub-block processing in Matlab
Color image in Matlab

- Since a color image requires three separate items of information for each pixel, a (true) color image of size $m \times n$ is represented in Matlab by an array of size $m \times n \times 3$: a three dimensional array.

```matlab
g >> x=imread('onion.png');
g >> imshow(x);
g >> size(x)
```

```
ans =

135   198     3
```
Color image in Matlab

• We can isolate each color component by the colon operator:
  – \( x(:, :, 1) \) is the first, or Red component.
  – \( x(:, :, 2) \) is the second, or Green component.
  – \( x(:, :, 3) \) is the third, or Blue component.

The color components can all be viewed with ‘imshow’ function

\[
\begin{align*}
  \text{>> figure, imshow(x(:, :, 1))} \\
  \text{>> figure, imshow(x(:, :, 2))} \\
  \text{>> figure, imshow(x(:, :, 3))}
\end{align*}
\]
Color image in Matlab

Red component  Green component  Blue component

The RGB components
Transform between Color Spaces

- $y = \text{rgb2hsv}(x)$;

The color components can all be viewed with ‘imshow’ function

- $\text{imshow}(y)$
- $\text{imshow}(y(:,:,1))$
- $\text{imshow}(y(:,:,2))$
- $\text{imshow}(y(:,:,3))$

- $z = \text{hsv2rgb}(y)$;
- $\text{imshow}(z)$;
Outline of Lab 7

- Color image in Matlab
- **DCT and IDCT in Matlab**
- Sub-function definition
- Sub-block processing in Matlab
1-D DCT and IDCT

- **DCT**: discrete cosine transform (one-dimensional).

- \( Y = DCT(X) \) returns the discrete cosine transform of \( X \). The vector \( Y \) is the same size as \( X \) and contains the discrete cosine transform coefficients.

- \( Y = DCT(X,N) \) pads or truncates the vector \( X \) to length \( N \) before transforming.
1-D DCT and IDCT

- **IDCT**: inverse discrete cosine transform (one-dimensional).

- $X = IDCT(Y)$ inverts the DCT transform, returning the original vector if $Y$ was obtained using $Y = DCT(X)$.

- $X = IDCT(Y,N)$ pads or truncates the vector $Y$ to length $N$ before transforming.
An example (1-D DCT)

```matlab
>> a = imread('onion.png');
>> a0 = a(1,1:8,1)
a0 =
    63   67   65   68   66   64   63   66
>> da0 = dct(double(a0))
da0 =
    184.5549   0.6646  -2.1184  -2.6833  1.4142  -1.8659  -1.4186  -1.9688
```
An example (1-D IDCT)

```matlab
>> ra0 = idct(da0)
ra0 =
   63.0000  67.0000  65.0000  68.0000  66.0000  64.0000  63.0000  66.0000

>> diff = ra0 - double(a0)
diff =
   1.0e-013 *
   -0.1421  -0.1421  -0.1421  -0.1421  -0.1421  -0.1421  -0.1421  -0.1421
```
2-D DCT and IDCT

- DCT2 computes 2-D discrete cosine transform.
- \( B = DCT2(A) \) returns the discrete cosine transform of \( A \). The matrix \( B \) is the same size as \( A \) and contains the discrete cosine transform coefficients.
- \( B = DCT2(A,[M N]) \) or \( B = DCT2(A,M,N) \) pads the matrix \( A \) with zeros to size \( M \)-by-\( N \) before transforming. If \( M \) or \( N \) is smaller than the corresponding dimension of \( A \), \( DCT2 \) truncates \( A \).
- This transform can be inverted using IDCT2.
2-D DCT and IDCT

- IDCT2 computes 2-D inverse discrete cosine transform.

- \( B = IDCT2(A) \) returns the two-dimensional inverse discrete cosine transform of \( A \).

- \( B = IDCT2(A,[M N]) \) or \( B = IDCT2(A,M,N) \) pads \( A \) with zeros (or truncates \( A \)) to create a matrix of size \( M \)-by-\( N \) before transforming.

- For any \( A \), \( IDCT2(DCT2(A)) \) equals \( A \) to within roundoff error.
An example (2-D DCT and IDCT)

```matlab
>> a = imread('onion.png');
>> a1 = a(1:8,1:8,1)

a1 =

63  67  65  68  66  64  63  66
61  61  64  66  67  67  66  66
61  61  64  65  66  65  65  67
65  63  64  65  63  66  64  65
62  60  60  65  66  64  64  64
66  64  63  64  67  67  68  65
62  64  64  67  66  63  64  65
65  60  61  66  66  61  63  66
```
An example (2-D DCT)

>> da1 = dct2(double(a1))

da1 =

515.1250  -6.9009   -4.4097   -0.1354   5.6250    0.7751  -1.8265    0.4425
 2.2310  -0.9758   -1.7370   -1.8775  -2.6824  -1.3261    1.0516  -1.0729
 1.2950    1.6965   -0.8687  -3.0164   1.8362  -1.5076  -0.1527  -1.4253
 2.7060    1.6665  -0.1457    1.5836  -2.0586  -1.2816  -1.3506  -1.4155
-2.6250    3.0794    1.0312  -0.4235    2.8750   0.4794  -1.1036  -0.5436
 1.9798    3.9761    0.6547  -2.9155  -1.5988  -0.7426  -1.2419 -0.3764
 1.4931    0.1617    2.0973    0.9814    0.1865  -0.4356   0.3687    0.4056
-3.0155  -1.2068   -1.9821  -2.1311    1.4400  -1.4382    0.0831 -1.3652
An example (2-D IDCT)

```matlab
>> a2=idct2(da1)

a2 =

63.0000   67.0000   65.0000   68.0000   66.0000   64.0000   63.0000   66.0000
61.0000   61.0000   64.0000   66.0000   67.0000   67.0000   66.0000   66.0000
61.0000   61.0000   64.0000   65.0000   66.0000   65.0000   65.0000   67.0000
65.0000   63.0000   64.0000   65.0000   65.0000   63.0000   66.0000   64.0000
62.0000   60.0000   60.0000   65.0000   66.0000   64.0000   64.0000   64.0000
66.0000   64.0000   63.0000   64.0000   67.0000   67.0000   68.0000   65.0000
62.0000   64.0000   64.0000   67.0000   66.0000   63.0000   64.0000   65.0000
65.0000   60.0000   61.0000   66.0000   66.0000   61.0000   63.0000   66.0000
```
An example (Result)

\[ \text{diff} = a1 - a2 \]

\[ \text{diff} = \]

\[ 1.0e-013 \times \]

\[
\begin{array}{cccccccccc}
0.1421 & 0.1421 & 0.1421 & 0.1421 & 0.1421 & 0.1421 & 0.1421 & 0.1421 \\
0.1421 & 0.2132 & 0.1421 & 0.1421 & 0.1421 & 0.1421 & 0.1421 & 0.2842 \\
0.2842 & 0.3553 & 0.2842 & 0.2842 & 0.2842 & 0.2842 & 0.4263 & 0.2842 \\
0.1421 & 0.1421 & 0.0711 & 0.1421 & 0.1421 & 0.0711 & 0.1421 & 0.1421 \\
0.2842 & 0.2842 & 0.1421 & 0.2842 & 0.2842 & 0.1421 & 0.2842 & 0.2842 \\
0.1421 & 0.1421 & 0.1421 & 0.0711 & 0.1421 & 0.1421 & 0.1421 & 0.1421 \\
0.1421 & 0.2132 & 0.0711 & 0.1421 & 0.1421 & 0.0711 & 0.2132 & 0.1421 \\
0.1421 & 0.1421 & 0.0 & 0.1421 & 0.1421 & 0.0 & 0.1421 & 0.1421 \\
\end{array}
\]
Outline of Lab 7

- Color image in Matlab
- DCT and IDCT in Matlab
- *Sub-function definition*
- Sub-block processing in Matlab
Direct definition

- The useful sub-function can be directly defined in the body of main function.

```matlab
function output = sqrSum(x,y)
output = square(x)+square(y);

%%%%%%sub-function %%%%%
function b = square(a)
b = a*a;
```

main function:
```matlab
sqrSum(x,y)
```

sub-function:
```matlab
square(a)
```
Nested function definition

- We also can define nested functions.

```matlab
function x = MainFun(p1,p2)
x1 = nestFun(p2);
    function y = nestFun(p3)
        y = 2 * p3;
    end
end
x = x1 + p1;
end
main function: MainFun(p1,p2)
sub-function: nestFun(p3)
```
Examples

$$\text{>> sqrSum}(3,4)$$

$$\text{ans} =$$

25

$$\text{>> MainFun (1,3)}$$

$$\text{ans} =$$

7
Outline of Lab 7

• Color image in Matlab
• DCT and IDCT in Matlab
• Sub-function definition
• *Sub-block processing in Matlab*
Block processing

- \( B = \text{BLKPROC}(A, [M \ N], \text{FUN}) \)
  - BLKPROC implements distinct block processing for image
  - processes the image \( A \) by applying the function \( \text{FUN} \) to each distinct \( M \)-by-\( N \) block of \( A \), padding \( A \) with zeros if necessary
  - \( \text{FUN} \) is a \text{FUNCTION\ HANDLE} that accepts an \( M \)-by-\( N \) matrix, \( X \), and returns a matrix, vector, or scalar \( Y \): \( Y = \text{FUN}(X) \)
  - \( \text{BLKPROC} \) does not require that \( Y \) be the same size as \( X \). However, \( B \) is the same size as \( A \) only if \( Y \) is the same size as \( X \)
An example

```plaintext
>> a1=

63  67  65  68  66  64  63  66
61  61  64  66  67  67  66  66
61  61  64  65  66  65  65  67
65  63  64  65  65  63  66  64
62  60  60  65  66  64  64  64
66  64  63  64  67  67  68  65
62  64  64  67  66  63  64  65
65  60  61  66  66  61  63  66
```
An example (DCT2 on sub-block)

FUN can be a FUNCTION_HANDLE created using @.

```matlab
>> fun = @dct2;
>> da1 = blkproc(a1, [4 4], fun)
```

```
da1 =

   255.7500   -5.2495    1.2500   -0.2610   261.5000    0.1913    2.0000    0.4619
    2.0951   -1.7803   -1.1713   -2.2374    0.7325    0.7071    1.2273   -1.9142
    4.2500    2.2537   -0.2500   -1.7453   -3.0000   -0.0793    0.5000    1.1152
    0.4852   -0.2374   -0.8678   -0.7197   -0.8446   -0.9142    1.6564   -0.7071
  253.2500   -2.2865    5.2500   -0.9471   259.7500    1.0920    2.7500    1.9831
   -1.6332    0.6339   -0.7093   -0.4445    1.8710    1.3839   -2.9069   -0.4268
   -3.7500   -0.5972    3.2500    0.5180   -2.7500   -0.3266    2.2500   -0.1353
   -0.6765   -3.4445   -1.0592   -1.1339   -2.6692    0.0732    1.4747   -0.3839
```
An example (IDCT2 on sub-block)

```
>> fun = @idct2;
>> a2 = blkproc(da1, [4 4], fun)

a2 =

63.0000  67.0000  65.0000  68.0000  66.0000  64.0000  63.0000  66.0000
61.0000  61.0000  64.0000  66.0000  67.0000  67.0000  66.0000  66.0000
61.0000  61.0000  64.0000  65.0000  66.0000  65.0000  65.0000  67.0000
65.0000  63.0000  64.0000  65.0000  65.0000  63.0000  66.0000  64.0000
62.0000  60.0000  60.0000  65.0000  66.0000  64.0000  64.0000  64.0000
66.0000  64.0000  63.0000  64.0000  67.0000  67.0000  68.0000  65.0000
62.0000  64.0000  64.0000  67.0000  66.0000  63.0000  64.0000  65.0000
65.0000  60.0000  61.0000  66.0000  66.0000  61.0000  63.0000  66.0000
```
An example (result)

```plaintext
>> diff = a2 - a1

diff =

1.0e-013 *

-0.2132   -0.2842   -0.2842   -0.2842   -0.2842   -0.2842   -0.2842   -0.2842
-0.1421   -0.2132   -0.2132   -0.2842   -0.2842   -0.2842   -0.2842   -0.2842
-0.1421   -0.1421   -0.2842   -0.2842   -0.2842   -0.2842   -0.2842   -0.2842
-0.2842   -0.2132   -0.3553   -0.2842   -0.2842   -0.2842   -0.2842   -0.2842
-0.1421   -0.2132   -0.2132   -0.2842   -0.2842   -0.2842   -0.2842   -0.2842
-0.1421   -0.1421   -0.1421   -0.2842   -0.2842   -0.2842   -0.2842   -0.2842
-0.1421   -0.0711   -0.0711   -0.1421   -0.2842   -0.2132   -0.3553   -0.2842
-0.2842   -0.2132   -0.2132   -0.2842   -0.2842   -0.2132   -0.3553   -0.2842
```

---

**Note:** The image contains a mathematical expression and a table, which is now represented in plain text. The expression `diff = a2 - a1` is followed by a table showing various numerical values resulting from the subtraction. The table includes values in the range of `-0.2132` to `0.2842` with a precision of `1.0e-013`. The table structure and values are maintained to ensure clarity and correct representation of the printed content.
Summary of Lab 7

- Color image in Matlab
  - Properties
  - Visualization
- DCT and IDCT in Matlab
  - 1-D DCT and 1-D IDCT
  - 2-D DCT and 2-D IDCT
- Sub-function definition
  - In main function
  - Nested function
- Sub-block processing in Matlab
Next lab course

- Quantization
- Zig-zag scan
- Run-length coding