Department of Computer Science
Faculty of Science
Prince of Songkla University

…28th.. March 2018
…10:00-11:50…

# LAB 9th **Geometric transformations**

## Objectives

1. Learn how to crop an image using the **imcrop** function.
2. Learn how to resize an image using the **imresize** function.
3. Learn how to flip an image upside down and left–right using **flipud** and **fliplr**.
4. Learn how to rotate an image using the **imrotate** function.
5. Explore interpolation methods for resizing and rotating images.

1) Open the peppers image and crop the image by the following processes:

```
J = imread('peppers.png');
figure, imshow(J);
impixelinfo
```

Use mouse points to the top left and bottom right positions that want to crop the image. When the mouse pointer points to the top left position, read $(x_1, y_1)$ from the lower left of the window, for the bottom right keep the position as $(x_2, y_2)$.

Assign the positions to parameters of **imcrop** function as the following commands:

```
xmin = x1; ymin = y1; width = x2-x1; height = y2-y1;
I3 = imcrop(J, [xmin ymin width height]);
figure, imshow(I3), title('Cropped Image');
```

**Question 1:** Which numbers did you record for the top left and bottom right coordinates and what do they mean?

2) Enlarge the cropped image by a scale factor of
3) By default, the function uses bicubic interpolation.

```
I_big1 = imresize(I3, 3);
figure, imshow(I3), title('Original Image');
figure, imshow(I_big1), title('Enlarged Image w/ bicubic interpolation');
```

4) Scale the image again using nearest-neighbor and bilinear interpolations.

```
I_big2 = imresize(I3, 3, 'nearest');
I_big3 = imresize(I3, 3, 'bilinear');
figure, imshow(I_big2), title('Resized w/ nearest-neighbor interpolation');
figure, imshow(I_big3), title('Resized w/ bilinear interpolation');
```

**Question 2:** Visually compare the three resized images. How do they differ?

5) Close any open figures.
6) Reduce the size of the peppers image by a factor of 0.25 in both dimensions.

```
[M,N,B] = size(J);
I_sm1 = J(1:2:M, 1:2:N, :);
figure, imshow(I_sm1);
```

**Question 3:** How did we scale the image?

**Question 4:** What are the limitations of this technique?

7) Shrink the image using the **imresize** function.

```
I_sm2 = imresize(J,0.5,'nearest');
I_sm3 = imresize(J,0.5,'bilinear');
I_sm4 = imresize(J,0.5,'bicubic');
figure, subplot(1,3,1), imshow(I_sm2), title('Nearest-neighbor Interpolation');
subplot(1,3,2), imshow(I_sm3), title('Bilinear Interpolation');
subplot(1,3,3), imshow(I_sm4), title('Bicubic Interpolation');
```

8) Flip the cameraman image upside down.
9) Flip the cameraman image from left to right.

```
I = imread('cameraman.tif');
J = flipud(I);
K = fliplr(I);
subplot(1,3,1), imshow(I), title('Original image')
subplot(1,3,2), imshow(J), title('Flipped upside-down')
subplot(1,3,3), imshow(K), title('Flipped left-right')
```

10) Close all open figures and clear all workspace variables.

11) Rotate the **eight** image by an angle of 35°.

```
I = imread('eight.tif');
I_rot = imrotate(I,35);
imshow(I_rot);
```

**Question 5:** Inspect the size (number of rows and columns) of I_rot and compare it with the size of I. Why are they different?

**Question 6:** The previous step rotated the image counterclockwise. How would you rotate the image 35° clockwise?

12) Rotate the same image using bilinear interpolation.

```
I_rot2 = imrotate(I,35,'bilinear');
figure, imshow(I_rot2)
```

**Question 7:** How did bilinear interpolation affect the output of the rotation?

13) Rotate the same image, but this time crop the output.

```
I_rot3 = imrotate(I,35,'bilinear','crop');
figure, imshow(I_rot3)
```

**Question 8:** How did the crop setting change the size of our output?