

# Image Processing

## Intensity Transformation and Spatial Filtering (Part I)

Pattern Recognition and Image Processing Laboratory (Since 2012)

## Introduction

---

Transformation

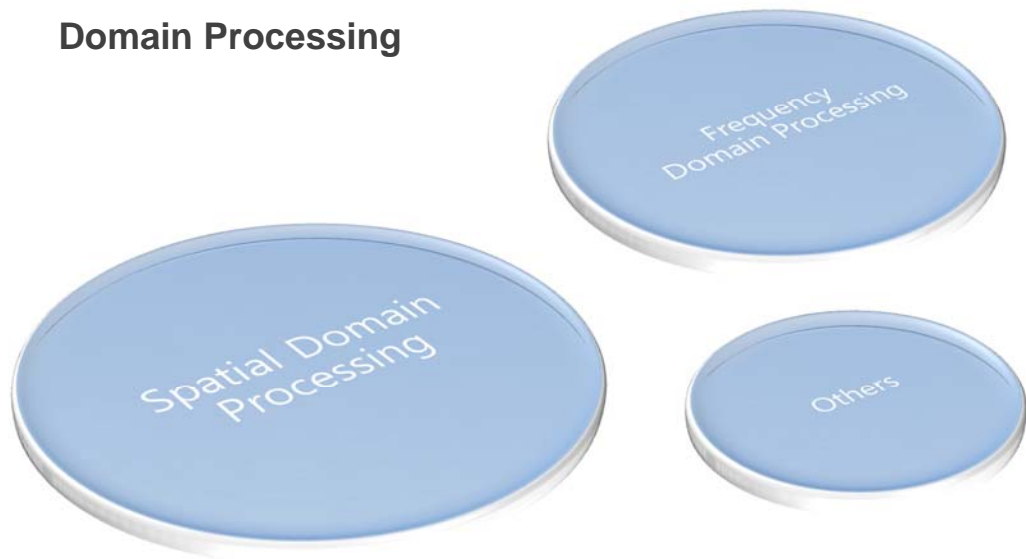




# Introduction

---

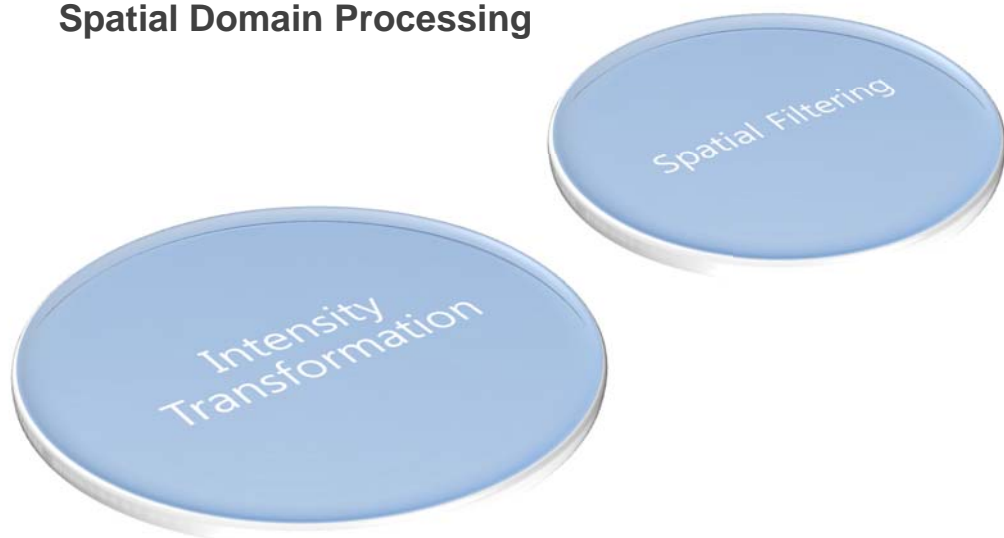
## Domain Processing



# Introduction

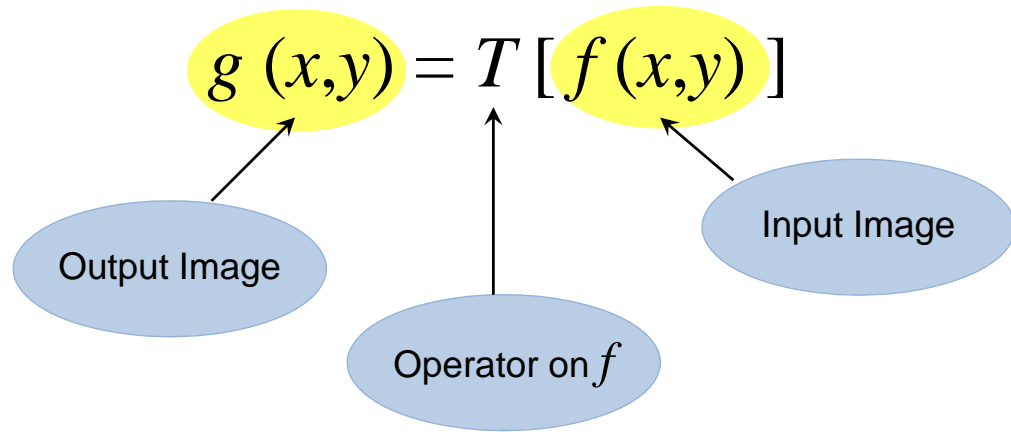
---

## Spatial Domain Processing



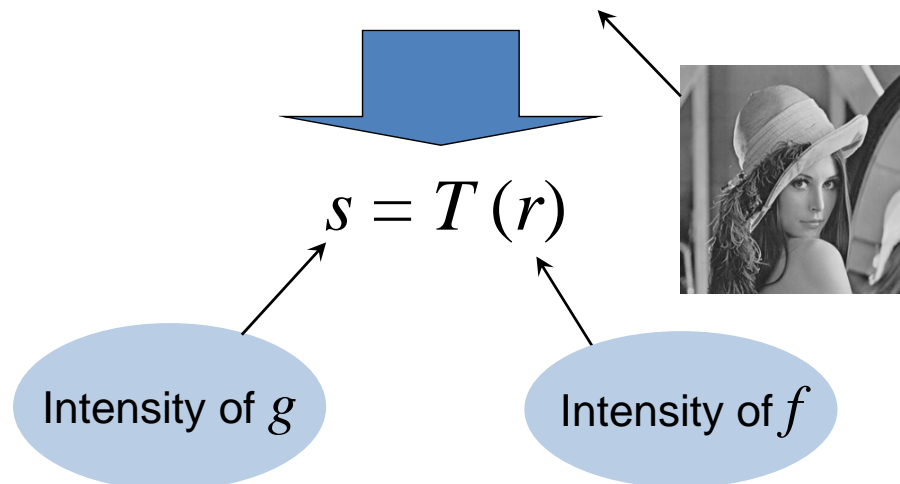
## Intensity Transformation Function

A spatial domain processing is denoted by the expression.



## Intensity Transformation Function

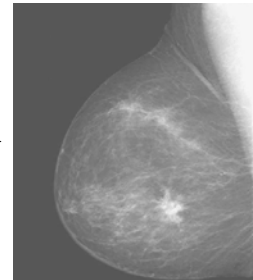
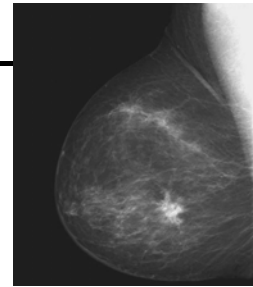
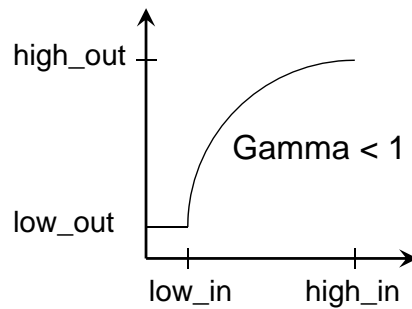
$$g(x,y) = T[f(x,y)]$$



## Intensity Transformation Function

Function `imadjust`

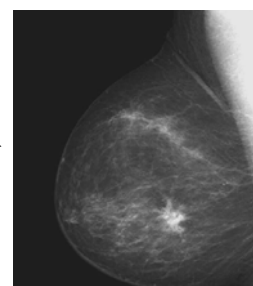
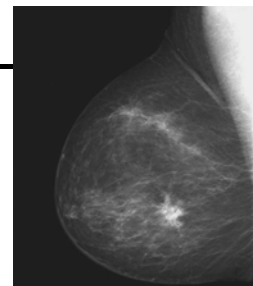
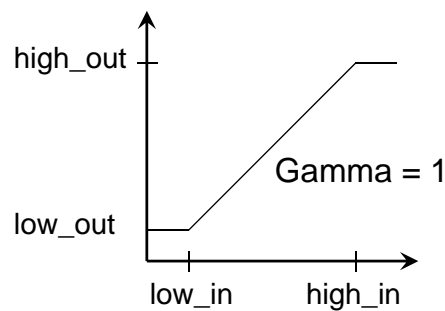
```
>> g = imadjust(f, [low_in high_in], ...  
                [low_out high_out], gamma)
```



## Intensity Transformation Function

Function `imadjust`

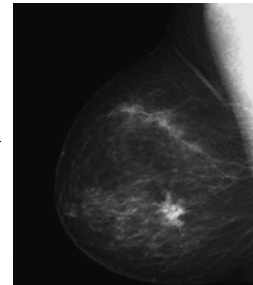
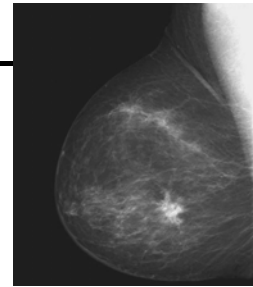
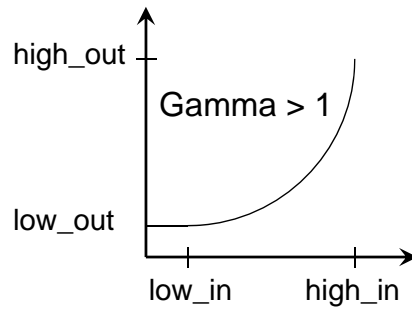
```
>> g = imadjust(f, [low_in high_in], ...  
                [low_out high_out], gamma)
```



## Intensity Transformation Function

Function `imadjust`

```
>> g = imadjust(f, [low_in high_in], ...  
                [low_out high_out], gamma)
```



## Intensity Transformation Function

```
>> ex3_01 % See demonstration
```



## Intensity Transformation Function

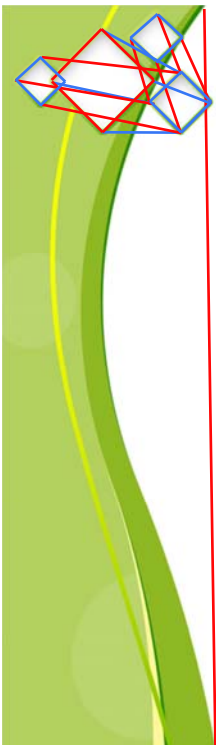
---

```
>> f = imread('breast.tif');  
>> g1 = imadjust(f, [0 1], [1 0]); % Neg. Image  
>> g11 = imcomplement(f);  
>> imshow(g1), figure, imshow(g11)
```

} same

---

**Note:** Compare the results between g1 and g11.



## Intensity Transformation Function

---

```
>> g2 = imadjust(f, [0.5 0.75], [1 0]);  
>> g3 = imadjust(f, [ ], [ ], 2);  
>> figure, imshow(g2)  
>> figure, imshow(g3)
```

---

**Note:** Compare the results between g2 and g3.



## Logarithm Transformation

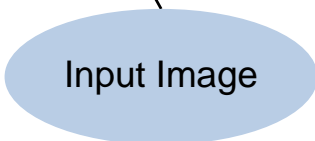
---

Logarithm transformation is implemented using the expression

$$>> g = c * \log(1 + \text{double}(f))$$



Constant



Input Image

---

**Note:** One of the principal uses of the log transformation is to suppress dynamic range.

เป็นการมองข้อมูลที่กระจายตัวกันให้อยู่ใน range เดียวกัน คล้ายๆกับการ normalize

เกลี่ยสีของรูปภาพให้อยู่ในระดับใกล้เคียงกัน (เช่นรูปที่ แยกสีขาวกับดำชัดเจน มันจะแยกให้ให้สีขาวเข้ม ขึ้น(เทา) และทำให้สีดำอ่อนลง(เทา))



## Logarithm Transformation

---

```
>> f = imread('pout.tif'); imshow(f);  
>> c = 1;  
>> gc = c*log(1+double(f));  
>> gc1 = im2uint8(mat2gray(gc)); % decompress  
                                % to the full range of the display.  
>> figure, imshow(uint8(gc)), figure, imshow(gc1);
```

---

**Note:** Compare the results between gc and gc1.

ถ้า c มากขึ้นรูปจะสว่างขึ้น

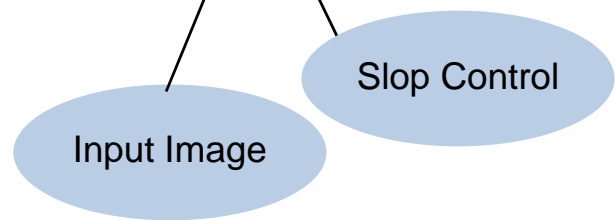
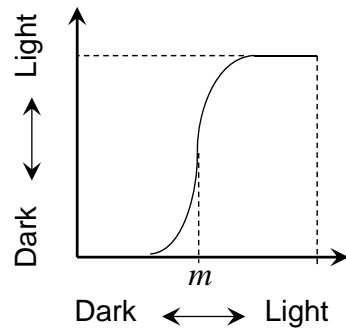




## Contrast-Stretching Transformation

A contrast-stretching transformation function is defined as

$$s = T(r) = \frac{1}{1 + (m/r)^E}$$



ถ้าตั้ง  $m$  ไปทางซ้ายก็จะมีสเกลสว่างที่เยอะขึ้น

ตรงข้ามกับ contrast

เน้นให้รูปภาพมีความแตกต่างกัน  
ชัดเจนมากขึ้น

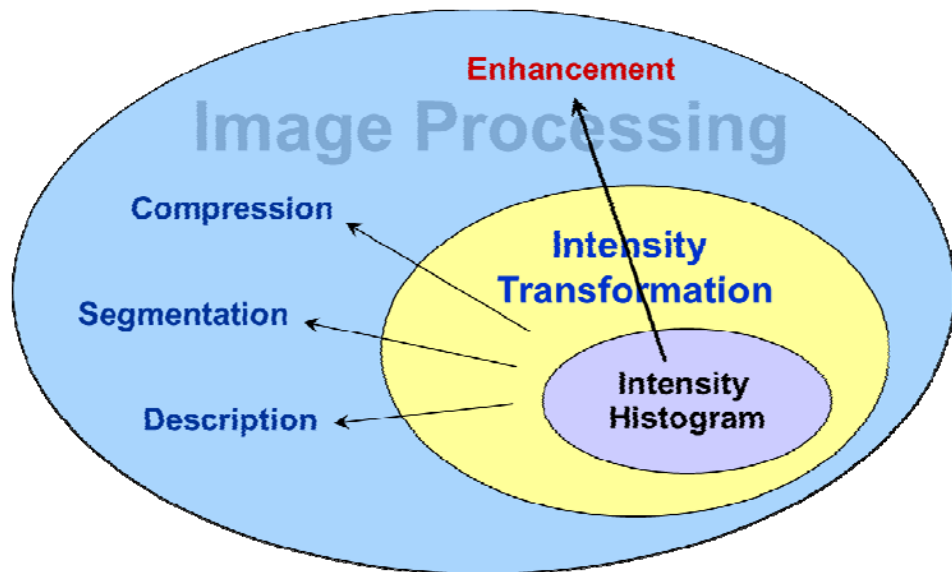


## Contrast-Stretching Transformation

>> ex3\_02 % See demonstration



## Intensity Histogram



## Intensity Histogram

The histogram of a digital image is defined as the discrete function

$$h(r_k) = n_k$$

where  $r_k$  is  $k^{th}$  intensity level in the interval  $[0, G]$  and  $n_k$  is the number of pixels in the image whose intensity level is  $r_k$ .



## Intensity Histogram

---

```
>> b = 256
>> h = imhist(f, b) % b is the number of bins,
>>                    % by default b = 256
>> p = imhist(f, b)/numel(f) % The normalized
>>                          % histogram
```

**numel = pixel ทั้งหมดที่อยู่ในภาพ**



## Intensity Histogram

---

### Histogram Equalization

```
>> h = imhist(f, b)    % b is the number of bins,
>>                    % by default b = 256.
>> p = imhist(f, b)/numel(f) % The normalized
>>                          % histogram.

>> ex3_03 % See demonstration
```

