

---

**Laboratorio di Immagini**

**Esercitazione 5:**

**Denoising & Segmentation**

---

Mauro Zucchelli  
21/04/2016

---

# Denoising

---

- Una delle principale applicazioni dei filtri è il denoising
  - Il rumore è uno dei principali artefatti che danneggia le immagini naturali
  - Esistono diversi tipi di rumore:
    - Gaussiano
    - Sale e pepe
-

# Rumore

---

original



gaussian noise



S&P noise



# Rumore in matlab

---

- Gaussiano
  - `img_noise = imnoise(img,'gaussian',0, 0.01);`
- Salt & Pepper
  - `img_noise = imnoise(img,'salt & pepper',0.05);`

# Denoising

---

- Le principali tecniche di rimozione del rumore sfruttano la proprietà che pixel vicini dovrebbero avere colore vicini
  - Il filtro media è uno dei più semplici
  - Ad ogni pixel della nuova immagine viene assegnato il valore della media del pixel stesso con i suoi vicini
-

# Denoising

---

```
function img2 = mean_filter(img,f_size)
```

```
[x,y] = size(img);
```

```
img2 = zeros(x,y);
```

```
% find centers
```

```
c = floor(f_size/2);
```

```
for i=(c+1):x-(c+1)
```

```
    for j=(c+1):y-(c+1)
```

```
        img_roi = img(i-c:i+c, j-c:j+c);
```

```
        img2(i,j) = mean(img_roi(:));
```

```
    end
```

```
end
```

```
img2 = uint8(img2);
```

```
end
```

---

# Denoising

---

- L'evoluzione del filtro mediano è il filtraggio Gaussiano
- Rispetto al precedente mantiene meglio i contorni dell'immagine

```
h = fspecial('gaussian', 5, 2.0);
```

```
den_gaus = imfilter(img_noise, h);
```

---

# Denoising

---

**original**



**noise**



**mean filter**



**gaus filter**



# Denoising

---

- Avendo creato artificialmente il rumore è possibile quantificare il miglioramento introdotto dal denoising

`sum(abs(img- img_noise))`

- Funzione **img\_difference**
-

# Denoising

---

- Funziona meglio il filtro media o Gaussiano?
  - Per quale tipo di rumore? Gaussiano o sale e pepe?
-

# Denoising

---

- Filtraggi Gaussiani o media su rumore S&P può addirittura peggiorare le cose!
  - L'ideale è utilizzare un filtro non lineare (che non ha equivalenti nel dominio delle frequenze)
  - Il più utile in questo caso è il filtro **mediano**
-

# Denoising: filtro mediano

---

- Create una nuova funzione `median_filter` che al posto dell'operatore di media abbia l'operatore di mediana
- Verificate i risultati in caso di rumore S&P e Gaussiano

# Denoising: filtro mediano

---

**original**



**mean filter**



**gaus filter**



**median filter**



# Fine dei filtraggi

---

**PAUSA!**

---

# Segmentazione

---

- La segmentazione è l'operazione che divide l'immagini in diverse sottoregione in cui i pixel sono accumulati da proprietà comuni
    - Colore
    - L'intensità in scala di grigio
    - La tessitura
    - Il fatto che le regioni sono delimitati da contorni
-

# Segmentazione

---

- Oggi vedremo un po' di segmentazioni basate sui livelli di grigio
    - Iterative thresholding
    - Otsu segmentation
-

# Iterative thresholding

---

1. Selezione una soglia  $T$  (a caso)
  2. Divido l'immagine in due regioni usando  $T$
  3. Calcolo le medie delle due regioni  $M1$  e  $M2$
  4. Aggiorno  $T$ :  $T=(M1+M2)/2$
  5. Ripeto dal passo 2 fino a che il processo non converge
-

# Iterative thresholding

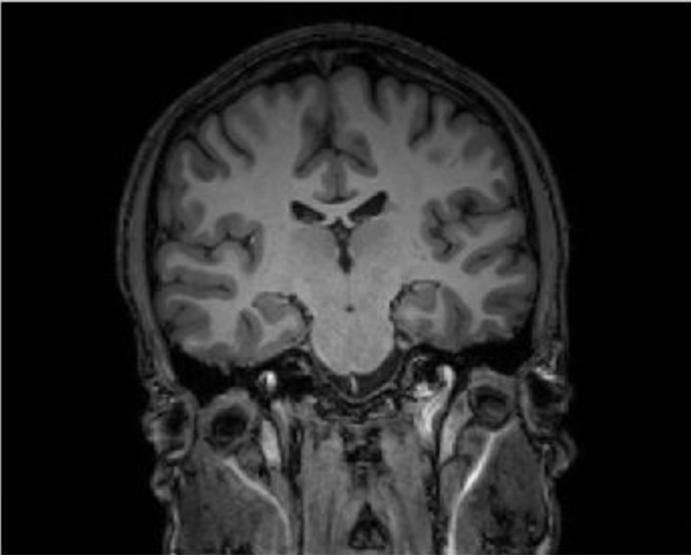
---

- Provate ad implementare l'iterative thresholding
  - Provatelo sulla vostra immagine preferita!
-

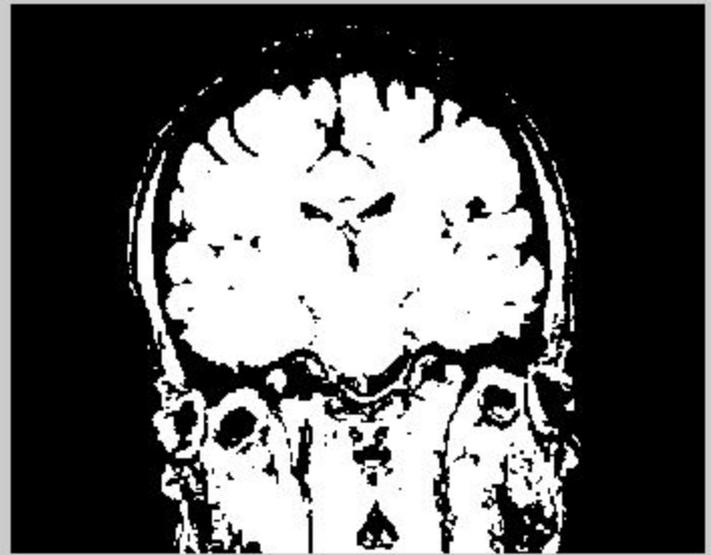
# Iterative thresholding

---

original



thresholed



# Iterative tresholding

---

- Il metodo Otsu minimizza la varianza intra classe, definita come somma pesata delle varianze delle due classi
- Otsu ha dimostrato che minimizzare la varianza intra classe equivale a massimizzare la varianza inter classe
- Otsu è facilmente estendibile a più classi

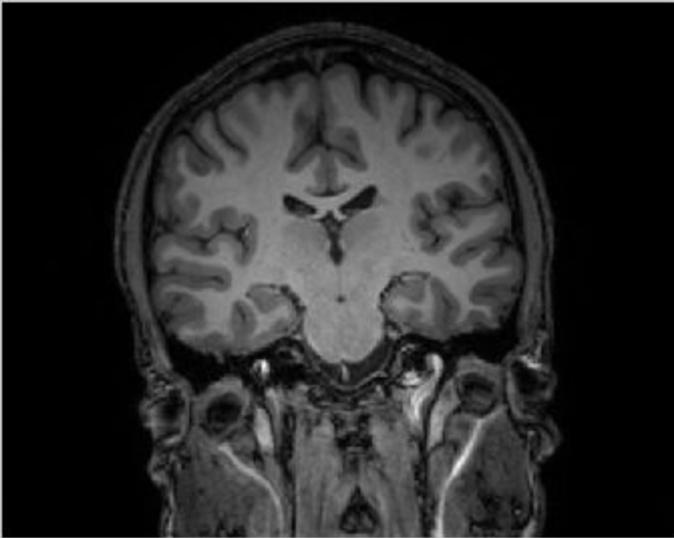
[cit] Wikipedia

---

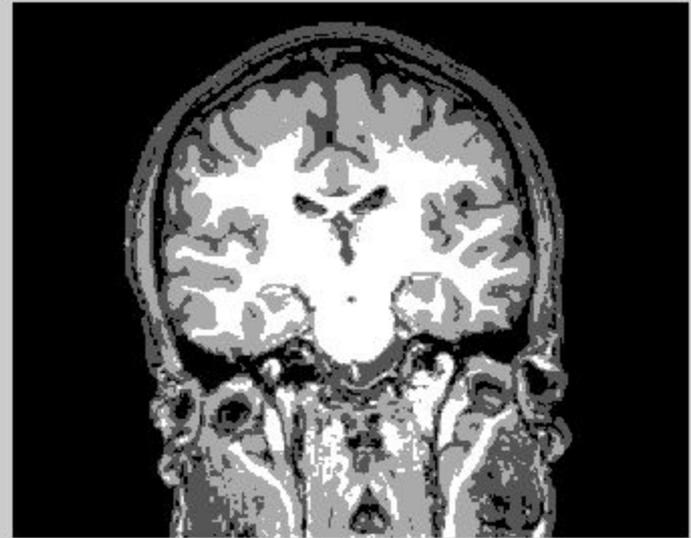
# Iterative thresholding

---

original



OTSU



**FINE!**

---

**The End**

**[http://profs.scienze.univr.  
it/~zucchelli/Esercitazione\\_5/resolution.jpg](http://profs.scienze.univr.it/~zucchelli/Esercitazione_5/resolution.jpg)**

---