

# Segmentasi Citra

---

- ❑ Proses untuk memisahkan citra menjadi bagian-bagian pembentuknya (region)
  - Merupakan fase penting dalam analisis citra otomatis → pengenalan objek
- ❑ Pendekatan algoritma segmentasi:
  - Berdasar discontinuity → perubahan warna mendadak → deteksi titik, garis, dan tepi
  - Berdasar similarity
    - ❑ Pengelompokan berdasar distribusi properti pixel (warna), contoh: thresholding
    - ❑ Mencari region secara langsung berdasar 'persamaan' karakteristik suatu area, contoh: region growing, split & merge

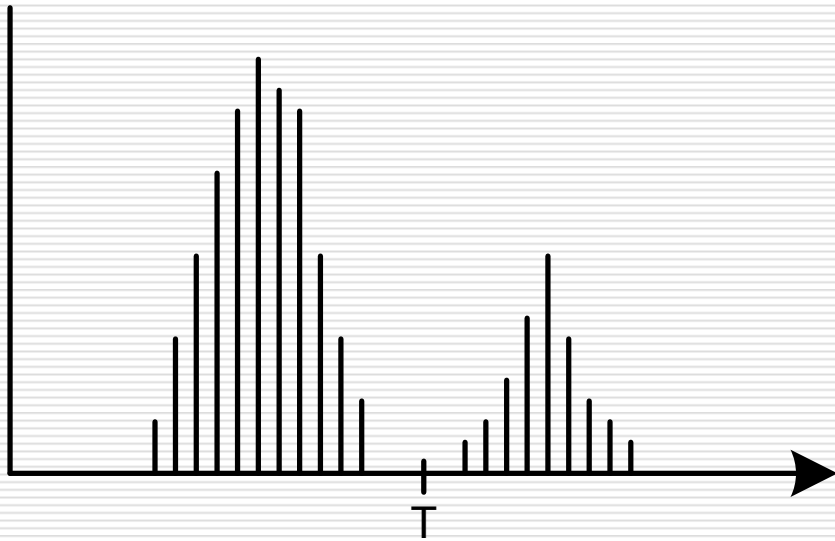
# Thresholding

---

- Asumsi:
  - antar objek yang akan dipisahkan memiliki intensitas warna yang berlainan
  - masing-masing objek memiliki warna yang hampir seragam
- Operasi: menempatkan satu atau lebih threshold pada sumbu datar histogram untuk memisahkan kelompok warna pixel yang diduga sebagai penyusun objek

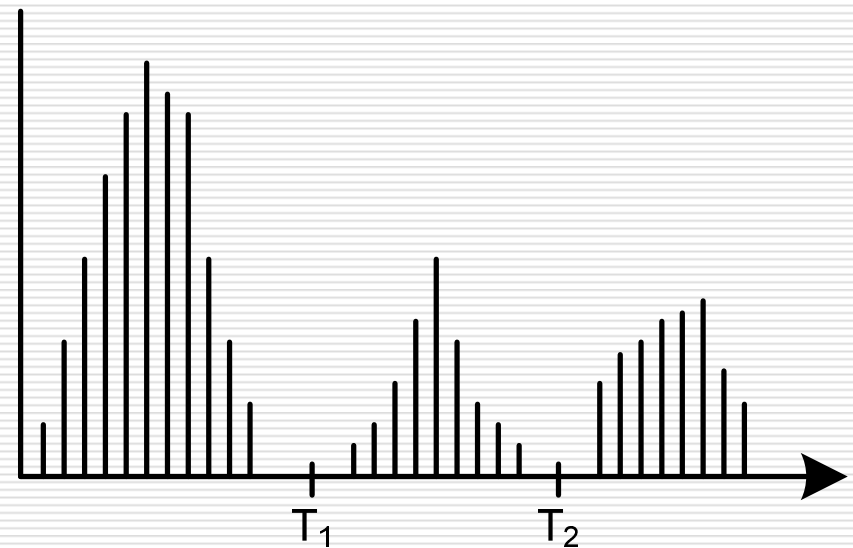
# Contoh thresholding dgn $g(x,y)=konstanta$

---



$$g(x,y) = \begin{cases} 1 & \text{if } f(x,y) > T \\ 0 & \text{if } f(x,y) \leq T \end{cases}$$

---

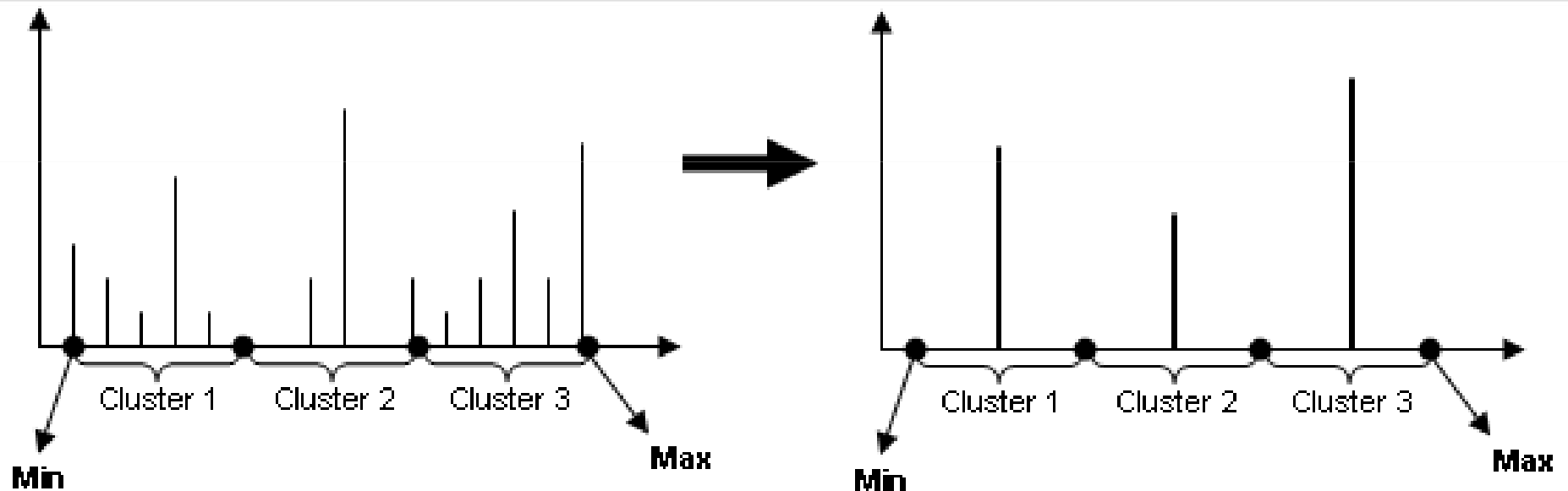


$$g(x,y) = \begin{cases} 0.0 & f(x,y) \leq T_1 \\ 0.5 & T_1 < f(x,y) \leq T_2 \\ 1.0 & T_2 < f(x,y) \end{cases}$$

# Contoh thresholding dgn

$g(x,y) = \text{rata-rata warna per cluster}$

---



# Jenis threshold

---

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

**f(x,y):** gray level pada titik (x,y)

**p(x,y):** properti lokal dari titik (x,y); misal: gray level rata-rata dari area ketetanggaan yang berpusat di (x,y)

- ❑ Global → T hanya tergantung dari f(x,y)
  - ❑ Local → T dipengaruhi oleh f(x,y) dan p(x,y)
  - ❑ Dynamic → T tergantung dari koordinat spasial titik (x,y)
-

# Kelemahan thresholding

---

- ❑ Penentuan nilai threshold yang tepat
  - ❑ Bermasalah jika kemunculan tiap warna dalam citra cenderung sama → tidak bisa diprediksi batas antar objek
-

# Pendekatan lain dalam proses segmentasi

---

- Segmentasi berorientasi daerah (region)
- Jika  $R$  adalah daerah keseluruhan citra  $\rightarrow$  segmentasi membagi  $R$  menjadi  $R_1, R_2, \dots, R_n$  sedemikian sehingga tercapai syarat segmentasi:

$$(a) \bigcup_{i=1} R_i = R$$

(b)  $R_i$  region yang terhubung,  $i = 1, 2, \dots, n$

(c)  $R_i \cap R_j = \phi$  untuk semua  $i$  dan  $j, i \neq j$

(d)  $P(R_i) = TRUE, i = 1, 2, \dots, n$

(e)  $P(R_i \cup R_j) = FALSE, i \neq j$

---

# Region Growing

---

- ❑ Prosedur yang mengelompokkan pixel atau sub-region menjadi region yang lebih besar
  - ❑ Pendekatan paling sederhana: *pixel aggregation*
    - Mulai dengan sekumpulan titik 'benih' (*seed*)
    - Dari titik-titik tsb region diperluas dengan menambahkan titik-titik tetangganya yang memiliki properti yang sama (misal: gray level, tekstur, warna)
    - Jika tidak ada lagi titik tetangga yang dapat ditambahkan lagi, maka proses untuk region tersebut dihentikan
-



# Ilustrasi

---

Seed

0	0	5	6	7
1	1	5	6	7
0	1	6	7	7
2	0	7	6	6
0	1	5	6	5

Citra asli

a	a	b	b	b
a	a	b	b	b
a	a	b	b	b
a	a	b	b	b
a	a	b	b	b

Hasil segmentasi;  
perbedaan warna  
absolut dg seed < 3

a	a	a	a	a
a	a	a	a	a
a	a	a	a	a
a	a	a	a	a
a	a	a	a	a

Hasil segmentasi;  
perbedaan warna  
absolut dg seed < 8

---

# Masalah dg region growing

---

- Penentuan lokasi seeds yang tepat
  - Tergantung aplikasi
  - Misal: warna yang sering muncul, warna terang dll
- Penentuan properti yang tepat untuk mengelompokkan titik menjadi region
  - Tergantung masalah dan data citra yang tersedia
  - Misal: intensitas, tekstur, data multispektral dll
- Kondisi penghenti
  - Dasar: jika tidak ada lagi titik tetangga yang memenuhi syarat
  - Tambahan: ukuran region, bentuk dll

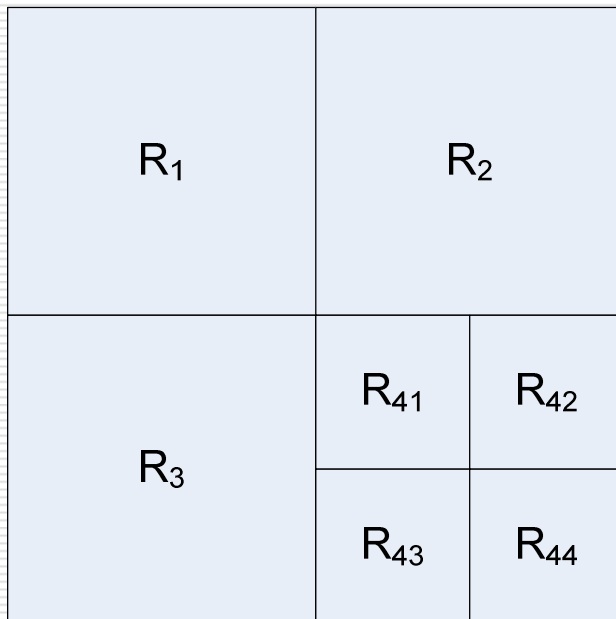
# Split & Merge

---

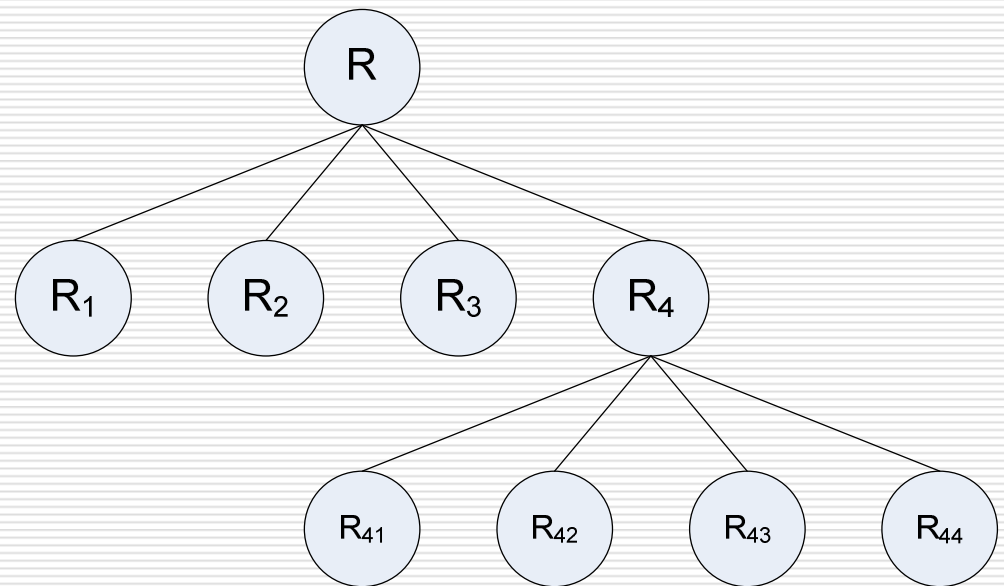
- ❑ Membagi citra menjadi sekumpulan region acak yang disjoint kemudian menggabungkan atau kembali membaginya hingga terpenuhi syarat segmentasi (a) – (e)
  - ❑ Algoritma bersifat rekursif
  - ❑ Memanfaatkan *quadtree*
-

# Ilustrasi

---



Citra terpartisi



Representasi Quadtree

---

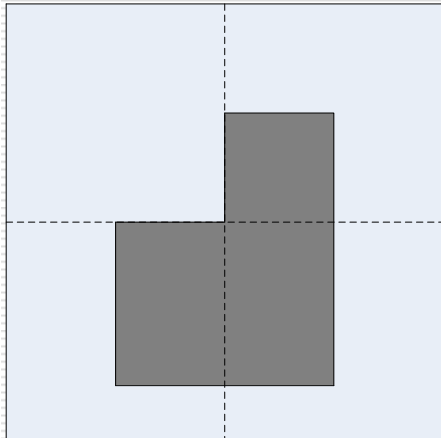
# Algoritma rekursif

---

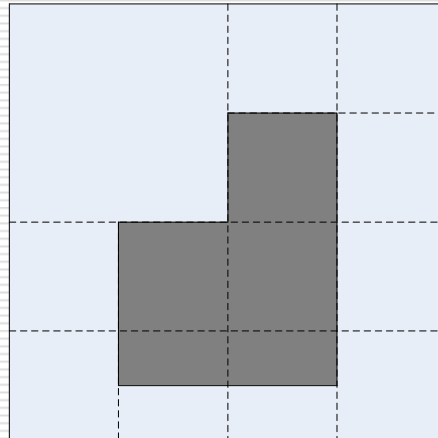
1. Bagi (*split*) setiap region  $R_i$  menjadi 4 quadran disjoint jika  $P(R_i)=FALSE$
  2. Gabung (*merge*) setiap region  $R_j$  dg  $R_k$  jika  $P(R_j \cup R_k)=TRUE$
  3. Berhenti jika tidak ada *split* maupun *merge* yang bisa dilakukan
-

# Contoh

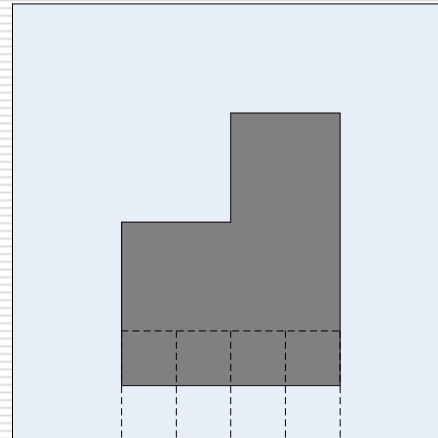
- ❑ Objek tunggal dg intensitas warna konstan
- ❑ Intensitas latar belakang konstan
- ❑  $P(R_i) = \text{TRUE}$  jika semua pixel dalam  $R_i$  memiliki intensitas warna yang sama



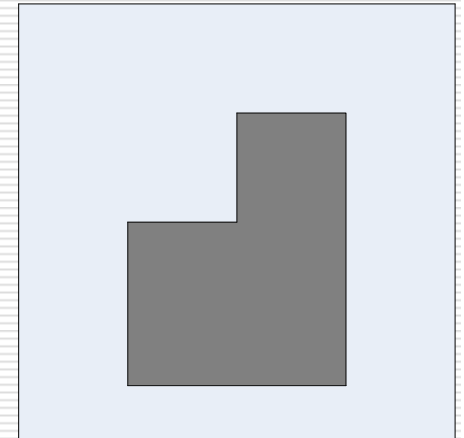
Operasi: split



Operasi: split



Operasi: split & merge



Operasi: merge

# Aplikasi segmentasi: magic wand

