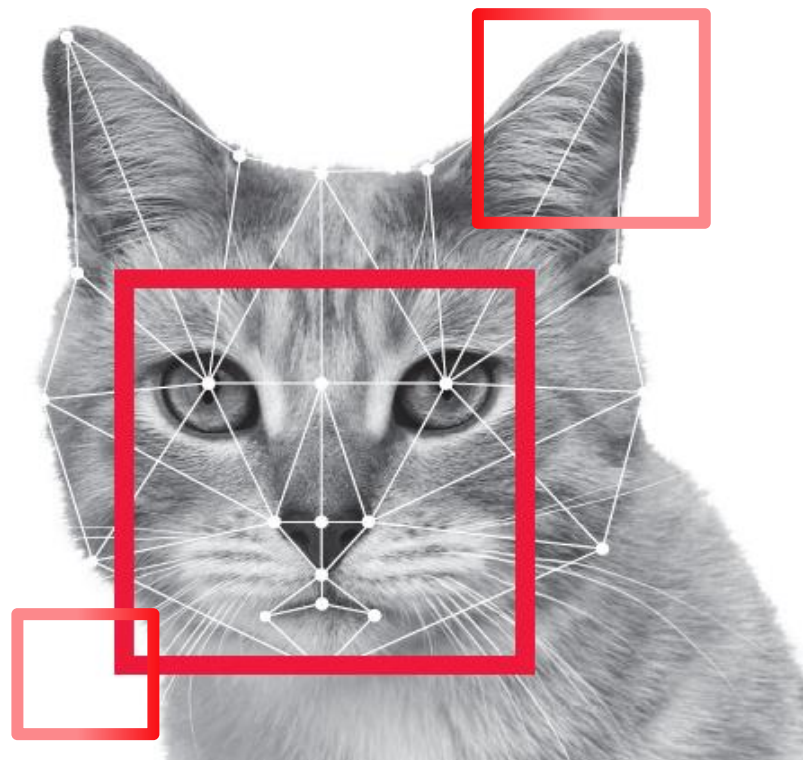


COMPUTER VISION 컴퓨터 비전

기본 개념부터 최신 모바일 응용 예까지



5장. 영상 분할

PREVIEW

■ 사람 뇌의 영상 분할



(a) 원래 영상

(b) 분할된 영상

그림 5-1 맨 왼쪽 영상을 보고 장면을 서술해 보자.

- “상자 위에 쌓여있는形形色색의 파프리카”라고 해석하는 과정에서 상자, 파프리카, 가격표, 글자와 같이 의미 있는 영역으로 분할
- 분할과 인식이 동시에 일어남

■ 컴퓨터 비전

- 현재는 분할 후 인식하는 순차 처리(동시 수행을 추구하는 연구도 있으나 초보 단계)
- 가장 어려운 문제 중 하나

각 절에서 다루는 내용

1. 영상 분할의 원리
2. 전통적 방법
3. 그래프 방법
4. 민시프트
5. 워터셰드
6. 대화식 물체 분할 - Graph Cut, Grab Cut
7. 알고리즘 선택

5.1 영상 분할의 원리

■ 분할의 정의

- 식 (5.1)은 엄밀성보다는 개념적인 정의

$$r_i \cap r_j = \emptyset, \quad 1 \leq i, j \leq n, i \neq j$$

겹칠 수 없다

$$\bigcup_{i=1,n} r_i = f$$

모든 영역은 전체를 덮어야한다

$$Q(r_i) = \text{참}, \quad 1 \leq i \leq n$$

같은 영역은 비슷한 성질

$$Q(r_i \cup r_j) = \text{거짓}, \quad 1 \leq i, j \leq n, \quad r_i \text{와 } r_j \text{는 이웃한 영역}$$

다른 영역은 다른 성질

■ 생각해 볼 점

- 적절한 분할이란? - 저분할과 과분할

- 사람 vs. 컴퓨터

- 사람은 선택적 주의집중과 능동 비전 기능을 가지며, 분할 과정에서 고급 지식 사용
 - 물체 모델, 지식, 의도 등
- 컴퓨터 비전은 그런 수준에 이르지 못함
 - 분할이 끝나야만 고급 지식을 이용하여 인식을 수행

5.1 영상 분할의 원리

■ 분할의 어려움

- 이웃 화소 몇 개를 보고 자신이 영역의 내부인지 경계인지 판단할 수 있을까?

→ 전역 연산 필요성

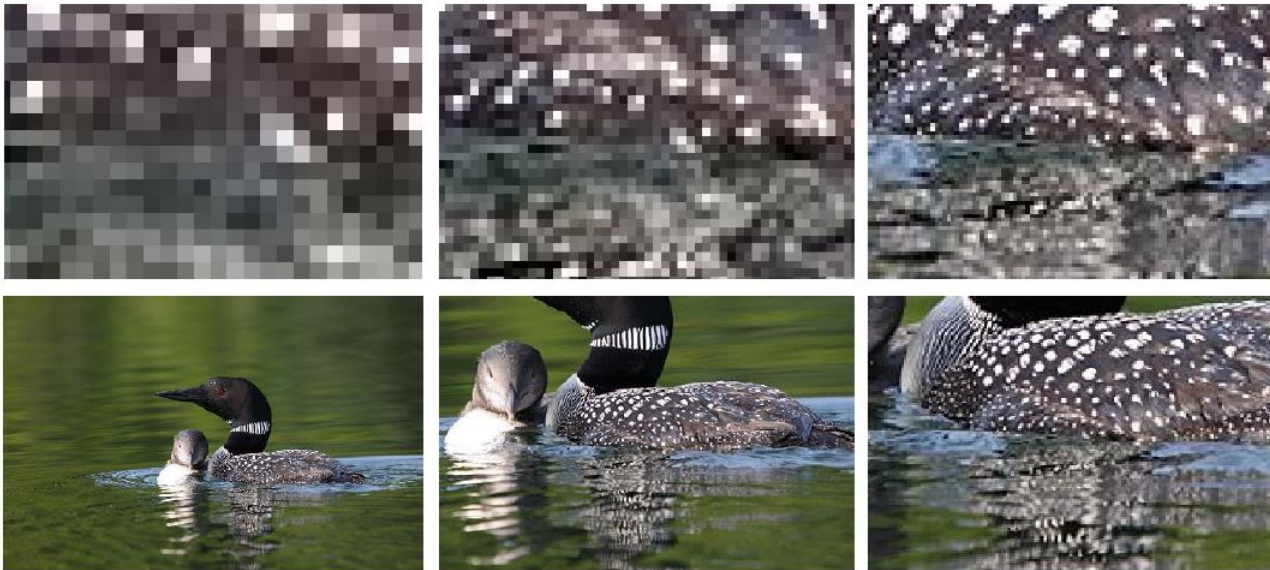


그림 5-2 영상 분할의 근원적인 어려움

시계방향으로 확대 → 축소

5.1 영상 분할의 원리

■ 에지(3장) vs. 영역

- 개념적으로 에지는 영역의 경계에 해당
- 하지만 에지 검출로는 한계
 - 거짓 긍정, 거짓 부정 (잘못된 에지) → 폐곡선을 이루지 못함

■ 영역 vs. 지역 특징(4장)

- 사람은 지역 특징보다 영역 분할에 훨씬 뛰어남
- 반면 컴퓨터 비전은 영역보다 지역 특징으로 문제 해결하는 사례 많음
- 미래는?

5.2 전통적 방법

■ 동작 조건

- 특수 조건 만족하거나 단순한 영상에서만 작동
 - 예) 공장 자동화, 문서 인식 등
 - 문제가 쉽다면 굳이 복잡한 알고리즘 쓸 필요 없음
- 자연 영상에서 매우 낮은 성능

5.2.1 임계화를 이용한 영역 분할

5.2.2 군집화를 이용한 영역 분할

5.2.3 분할합병

5.2.1 임계화를 이용한 영역 분할

■ 이진화를 이용한 영역 분할

- 문서 영상의 경우 Otsu 이진화(2.3.1절)는 훌륭한 영상 분할 알고리즘
- 하지만 명암 단계가 둘 이상인 경우는 오작동

■ 삼진화로 확장

- 이중 임계값 사용

$$v_{between}(t_1, t_2) = w_0(\mu_0 - \mu_g)^2 + w_1(\mu_1 - \mu_g)^2 + w_2(\mu_2 - \mu_g)^2 \quad \mu_g = \sum_{i=0, L-1} i\hat{h}_i : \text{전체 평균}$$

이때, $\mu_0 = \frac{1}{w_0} \sum_{i=0}^{t_1} i\hat{h}_i$, $\mu_1 = \frac{1}{w_1} \sum_{i=t_1+1}^{t_2} i\hat{h}_i$, $\mu_2 = \frac{1}{w_2} \sum_{i=t_2+1}^{L-1} i\hat{h}_i$ (5.2)

$$w_0 = \sum_{i=0}^{t_1} \hat{h}_i, \quad w_1 = \sum_{i=t_1+1}^{t_2} \hat{h}_i, \quad w_2 = \sum_{i=t_2+1}^{L-1} \hat{h}_i$$

$$(\hat{t}_1, \hat{t}_2) = \underset{0 < t_1 < t_2 < L-1}{\operatorname{argmax}} v_{between}(t_1, t_2) \quad (5.3)$$

세 영역이 다를수록 더 좋은 분할 : 분산 $v_{between}$ 이 클수록 좋다.

2.3.1 이진화와 Otsu 알고리즘

■ Otsu 알고리즘 [Otsu79]

- 이진화 했을 때 각 집합의 명함분포가 균일할수록 좋다는 원리에 근거
- 균일성은 분산으로 측정 (분산이 작을수록 균일성 높음)
- 분산의 가중치 합 $v_{within}(\cdot)$ 을 목적 함수(objective function, 혹은 비용 함수 cost function)로 이용한 최적화 알고리즘
- 모든 t 에 대해 검사

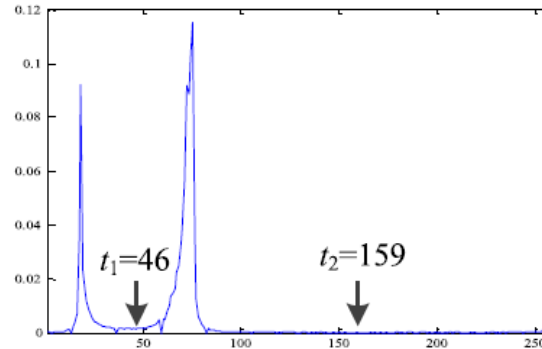
$$T = \underset{t \in \{0,1,\dots,L-1\}}{\operatorname{argmin}} v_{within}(t) \quad \begin{array}{l} w: \text{집합의 크기} \\ v: \text{분산} \end{array} \quad (2.6)$$

$$\begin{aligned} v_{within}(t) &= w_0(t)v_0(t) + w_1(t)v_1(t) \\ w_0(t) &= \sum_{i=0}^t \hat{h}(i), & w_1(t) &= \sum_{i=t+1}^{L-1} \hat{h}(i) \\ \mu_0(t) &= \frac{1}{w_0(t)} \sum_{i=0}^t i \hat{h}(i), & \mu_1(t) &= \frac{1}{w_1(t)} \sum_{i=t+1}^{L-1} i \hat{h}(i) \\ v_0(t) &= \frac{1}{w_0(t)} \sum_{i=0}^t \hat{h}(i)(i - \mu_0(t))^2, & v_1(t) &= \frac{1}{w_1(t)} \sum_{i=t+1}^{L-1} \hat{h}(i)(i - \mu_1(t))^2 \end{aligned} \quad (2.7)$$

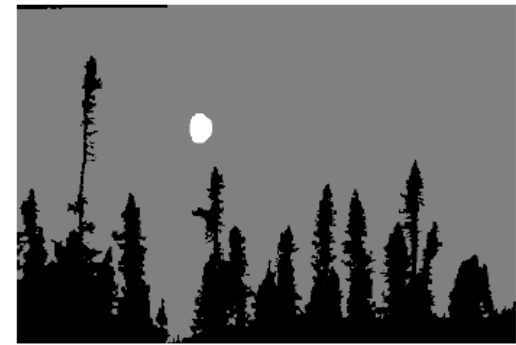
5.2.1 임계화를 이용한 영역 분할



(a) 원래 영상



(b) 명암 히스토그램



(c) 이중 임계값으로 분할한 영상

그림 5-3 이중 임계값 오츠크 알고리즘에 의한 영상 분할

알고리즘 5-1 이중 임계값 오츠크 알고리즘

입력 : 영상 $f(j, i)$, $0 \leq j \leq M-1$, $0 \leq i \leq N-1$

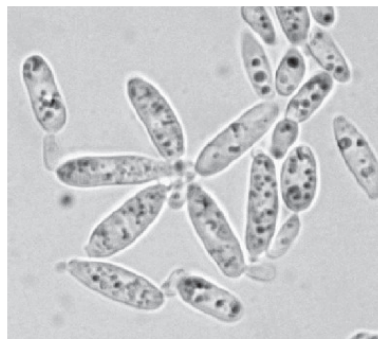
출력 : 삼진영상 $g(j, i)$, $0 \leq j \leq M-1$, $0 \leq i \leq N-1$ // 0, 1, 2 세 가지 값을 가진 영상

- 1 [알고리즘 2-1]을 이용하여 f 의 정규 히스토그램 h 을 만든다.
- 2 for($t_1=1$ to $L-3$)
- 3 for($t_2=t_1+1$ to $L-2$)
- 4 식 (5.2)를 이용하여 $v_{between}(t_1, t_2)$ 를 계산한다.
- 5 2~4행에서 가장 큰 $v_{between}$ 을 생성한 (t_1, t_2) 를 임계값 (\hat{t}_1, \hat{t}_2) 로 취한다.
- 6 (\hat{t}_1, \hat{t}_2) 로 f 를 삼진화하여 g 를 만든다.

5.2.1 임계화를 이용한 영역 분할

■ 적응적 임계화

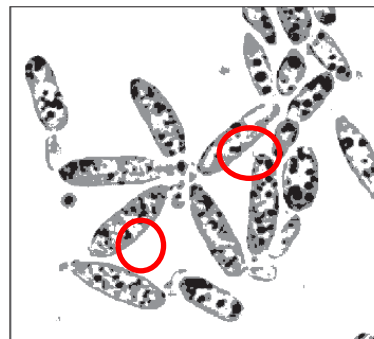
- 하나 또는 두 개의 임계값을 영상 전체에 적용하는 전역 방법의 한계
 - 지역적으로 명암 분포가 다른 경우 낮은 분할 품질



(a) 원래 영상



(b) [알고리즘 2-4]를 적용한 결과



(c) [알고리즘 5-1]을 적용한 결과

그림 5-4 효모 영상과 임계화한 영상

- 적응적 임계화로 해결 : 지역에 따라 적응적으로 임계값 결정
 - t 대신 $t(j, i)$: 지역에 따라 다른 임계값

$$b(j, i) = \begin{cases} 1, & f(j, i) \geq t(j, i) \\ 0, & \text{그렇지 않으면} \end{cases} \quad (5.4)$$

5.2.2 군집화를 이용한 영역 분할

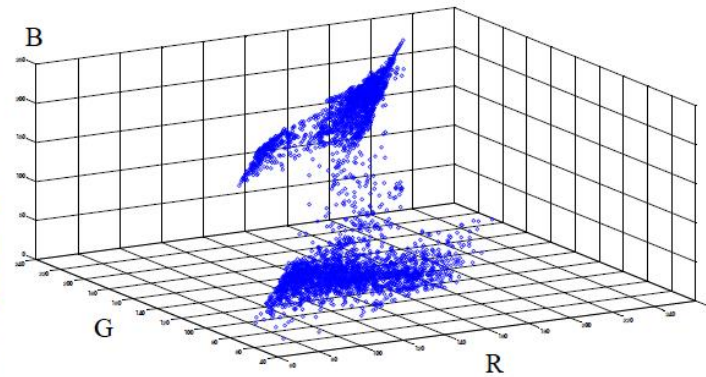
■ 군집화

- 화소를 RGB 3차원 컬러 공간으로 매핑한 후, k -means 로 군집화



(a) 원래 영상

그림 5-5 컬러 영상 화소를 RGB 공간에 매핑



(b) 3차원 공간에 매핑한 결과

5.2.2 *k*-means 유사 : 벡터 양자화

- 군집화Pattern Matching : 일치되는 것이 없으면 가장 가까운 값 선택
- Codebook의 index로 압축, 복원

벡터 양자화의 예)

키(cm)	몸무게(kg)
183	82
183	79
165	54
112	19
150	54
157	52
163	68
152	50
165	73
142	41
145	40
178	78

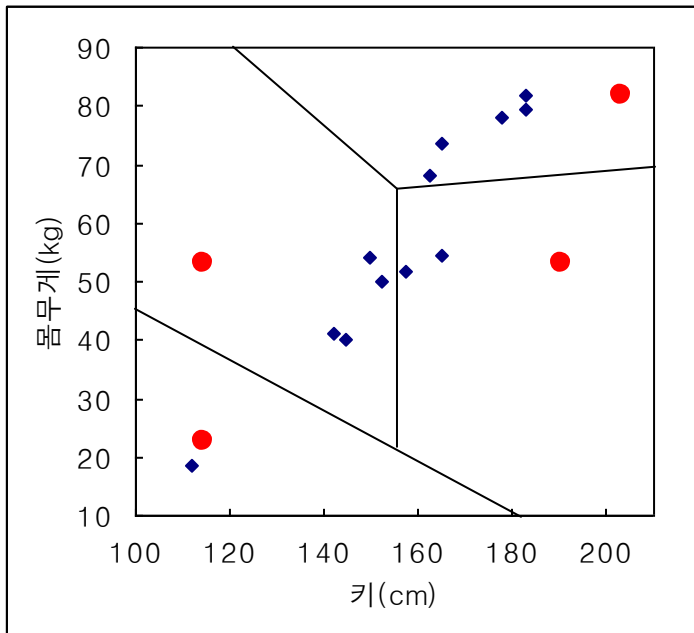
초기 중심점(centroid) 설정

키(cm)	몸무게(kg)
114	23
114	53
191	53
203	82

5.2.2 *k*-means 유사 : 벡터 양자화

첫 번째 단계

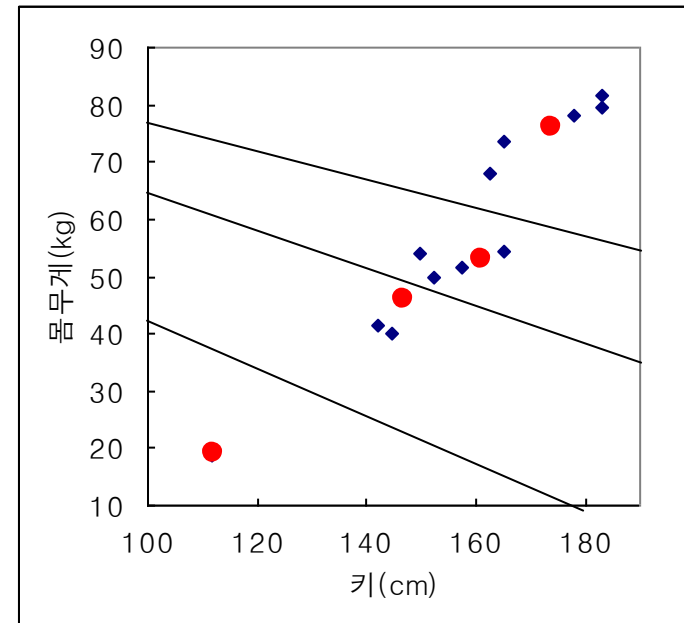
MSE = 348



두 번째 단계 - 평균에 의한 중심점 재계산

(112, 19), (147, 46), (161, 53), (174, 76)

MSE = 89

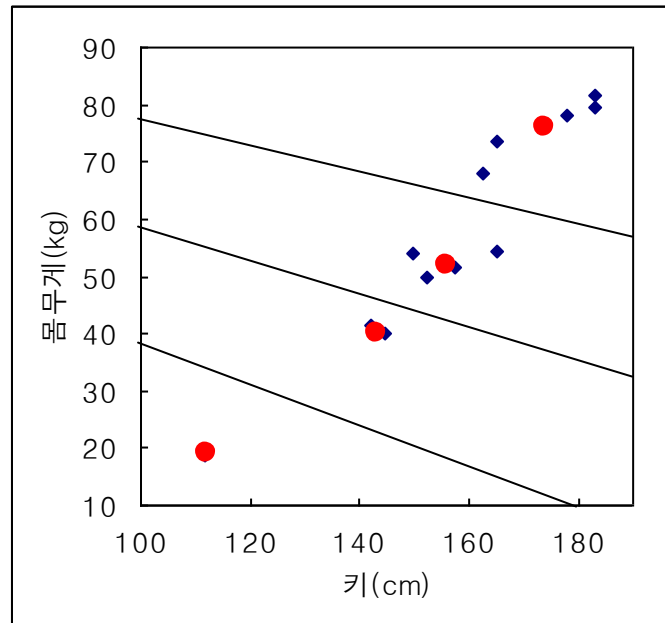


5.2.2 *k*-means 유사 : 벡터 양자화

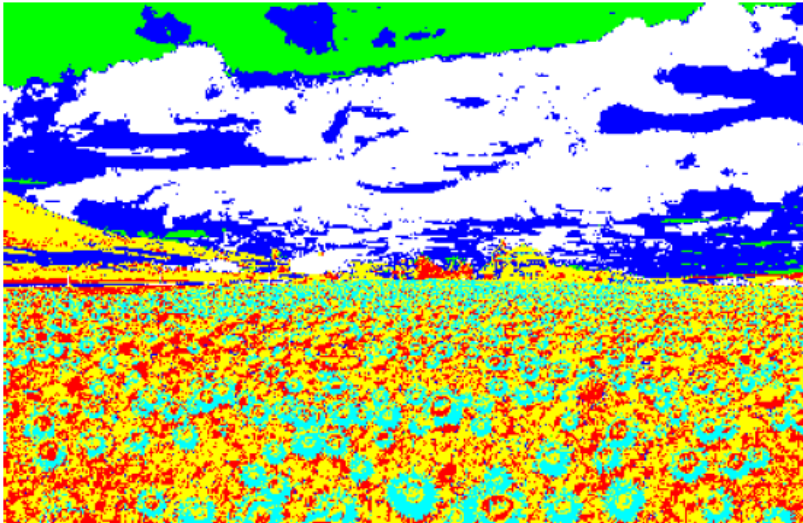
마지막 단계 - 중심점 재계산

(112, 19), (143, 40), (156, 52), (174, 76)

MSE = 72



5.2.2 군집화를 이용한 영역 분할



자연 영상에서 낮은 성능
← 5.4절은 진보한 군집화 알고리즘인
민시프트 사용

그림 5-6 k-means 군집화로 분할한 결과

5.2.3 분할 합병

■ 원리

- 영역의 균일성을 측정하는 $Q(r_i)$ 를 이용(있다고 치고!!!)하여 분할과 합병을 반복
 - $Q(r_i)$ 이 거짓이면 r_i 를 4개 영역으로 등분하고 재귀 반복
 - $Q(r_i \cup r_j)$ 가 참이면 r_i 와 r_j 를 합병
- 분할 결과는 4진 트리로 표현

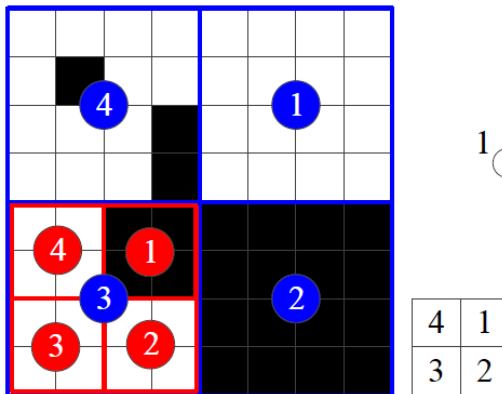
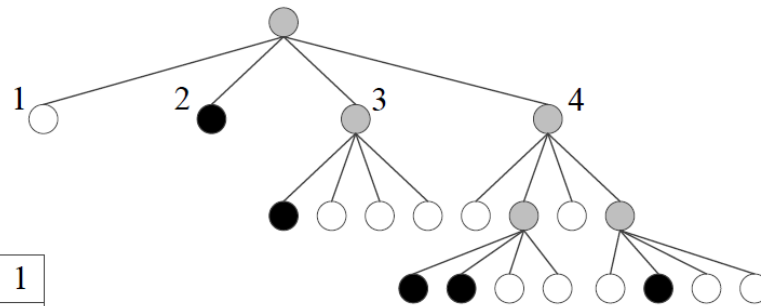


그림 5-7 4진 트리를 통한 영역 분할 결과



5.3 그래프(Graph) 방법

■ 그래프

- $G=(V, E)$
- 노드 집합 $V=\{v_1, v_2, \dots, v_n\}$
 - 화소 또는 슈퍼 화소(superpixel, 여러 화소가 모인 영역)가 노드
- 에지 집합 E
 - 이웃 노드 간에 에지 설정
 - 두 노드 v_p 와 v_q 를 연결하는 에지는 가중치 w_{pq} 를 가짐
 - 가중치는 유사도(같은 정도) 또는 거리(다른 정도)로 측정

$$\text{거리 } d_{pq} = \begin{cases} |f(v_p) - f(v_q)|, & v_q \in \text{Neigh}(v_p) \\ \infty, & \text{그렇지 않으면} \end{cases}$$

$$\text{유사도 } s_{pq} = \begin{cases} e^{-d_{pq}}, & v_q \in \text{Neigh}(v_p) \\ 0, & \text{그렇지 않으면} \end{cases} \quad (5.5)$$

이때, $\|\mathbf{x}(v_q) - \mathbf{x}(v_p)\| \leq r$ 이면 $v_q \in \text{Neigh}(v_p)$

5.3 그래프 방법

예제 5-1 영상의 그래프 표현

[그림 5-8(a)]는 10단계의 명암을 갖는 5×5 크기의 아주 간단한 영상을 보여준다. 그래프로 표현하기 위해 먼저 25개의 화소를 행 우선 순서에 따라 $v_0 \sim v_{24}$ 로 표기한다. 예를 들어 (1,2) 위치에 있는 화소는 노드 v_7 이 된다.

식 (5.5)에서 이웃을 규정하는 거리를 $r=1$ 로 설정하여 4-연결된 화소만 이웃이라고 가정하고, 에지의 가중치는 식 (5.5)의 맨 위에 있는 거리를 채택한다. 그래프를 인접 행렬로 표현하면 25×25 행렬이 되는데, 에지의 가중치 값을 계산하여 채우면 [그림 5-8(b)]와 같다.

	0	1	2	3	4
0	v_0	v_1	v_2	v_3	v_4
	3	-4	7	-5	2
	-0	1	7	6	1
1	v_5	v_6	v_7	v_8	v_9
	3	-3	6	-3	9
	-1	2	2	4	0
	...				
2					
	2	-6	8	-1	7
	-1	7	3	1	3
3					
	1	-0	1	-3	4
	-1	1	3	4	3
4					
	2	-0	2	-1	1

	v_0	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}	v_{12}	...	v_{23}	v_{24}
v_0	0	4	-	-	-	0	-	-	-	-	-	-	-	-	-	-
v_1	4	0	5	-	-	-	1	-	-	-	-	-	-	-	-	-
v_2	-	5	0	0	-	-	-	7	-	-	-	-	-	-	-	-
v_3	-	-	0	0	0	-	-	-	6	-	-	-	-	-	-	-
v_4	-	-	-	0	0	-	-	-	-	1	-	-	-	-	-	-
v_5	0	-	-	-	-	0	3	-	-	-	1	-	-	-	-	-
v_6	-	1	-	-	-	3	0	3	-	-	-	2	-	-	-	-
v_7	-	-	7	-	-	-	3	0	1	-	-	-	2	-	-	-
\vdots		
v_{24}	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0	0

(a) 입력 영상(화소를 잇는 값(파란색)은 에지 가중치) (b) 인접 행렬 표현

그림 5-8 명암 영상의 그래프 표현

이 그래프는 대각선 방향으로 세 줄의 띠를 형성하는데, 유효한 값이 드문 희소행렬(sparse matrix)이다. 이웃의 범위를 규정하는 r 을 크게 하면 띠의 폭이 늘어난다. 예를 들어, 8-이웃을 채택하면 10이었던 띠의 폭이 30이 된다.

- 가중치 값을 보기 쉽게 정리
- 정렬하면 작은 에지 위치를 알 수 있음

5.3 그래프 방법

■ 원리

- 유사도가 높은 노드 쌍은 같은 영역(연결요소), 낮은 노드 쌍은 다른 영역에 배치
- 유사도 낮은 에지가 분할선이 될 가능성 높음
- '가능성이 높다'라는 표현의 중요성
 - 지역적으로 유사도 낮더라도 전역적 판단에서 자르지 말아야 한다면 분할선으로 취하지 않음 → 전역 최적해 추구

■ 전역 최적화 문제의 구현

1. 어떤 분할의 좋은 정도를 측정하는 목적 함수 → 분할 품질 관련
2. 목적 함수를 최대화/최소화하는 최적해를 찾는 효율적인 탐색 알고리즘 → 속도 관련

5.3.1 최소 신장트리(Minimum Spanning Tree)

■ 원리

- 에지 가중치가 거리인 그래프로 표현
- 신장트리를 이용한 최적 분할 찾아냄 [Felzenszwalb2004]
 - 그림에서 회색 표시된 부분 그래프의 최소 신장트리는 빨간 에지

· 신장트리는 모든 노드를 포함하고 모든 노드 쌍이 연결되도록 하는 사이클이 없는 트리

v_0		v_1	v_2	v_3	v_4
3	-4	7	-5	2	0
0		1	7	6	1
v_5		v_6	v_7	v_8	v_9
3	-3	6	-3	9	-1
1		2	2	4	0
v_{10}		v_{11}	v_{12}	v_{13}	v_{14}
2	-6	8	-1	7	-3
1		7	3	1	3
v_{15}		v_{16}	v_{17}		
1	-0	1	-3	4	-1
1		1	3	4	3
2	-0	2	-1	1	-0
			...	v_{23}	v_{24}
				1	-0
				1	1

$$C = \{v_1, v_6, v_7, v_{11}, v_{12}\}$$

$$MST(C) = \{(v_1, v_6), (v_6, v_{11}), (v_{11}, v_{12}), (v_{12}, v_7)\}$$

$$intra(C) = \max_{e \in MST(C)} w_e \quad (5.6)$$

$$intra(C) = 2$$

둘 사이의 차이는 3이지만 같은 영역

어느 노드를 추가할까? v_{13} ? v_8 ?

그림 5-9 최소 신장트리

5.3.1 최소 신장트리

■ 두 연결요소의 균일성 측정

- 둘 중 하나라도 균일하지 않으면 큰 값 (균일하지 않음)
- k : 분할의 세밀함 조절하는 매개변수(작을수록 세밀하게 분할)

$$mult_intra(C_i, C_j) = \min(intra(C_i) + \tau(C_i), intra(C_j) + \tau(C_j)) \quad (5.7)$$

이때 $\tau(C) = \frac{k}{|C|}$ → 영역 안의 노드의 갯수

$$diff(C_i, C_j) = \min_{v_p \in C_i, v_q \in C_j} w_{pq} \quad (5.8)$$

2개의 연결요소가 붙어있는 에지들의 가중치 중 가장 작은 가중치

■ 두 연결요소가 적절히 분할되어 있는지 판정 (앞의 예제 참조)

- $D(C_i, C_j)$ 가 참이면 두 연결요소는 거칠지도 세밀하지도 않은 적당한 상태

$$D(C_i, C_j) = \begin{cases} \text{참,} & diff(C_i, C_j) > mult_intra(C_i, C_j) \\ \text{거짓,} & \text{그렇지 않으면} \end{cases} \quad (5.9)$$

5.3.1 최소 신장트리

■ 컬러 영상 f 의 분할

1. 3개의 채널을 독립적으로 분할한 후, 모두 같은 영역인 경우만 최종적으로 같은 영역
2. 3개의 채널을 하나의 벡터로 간주하여 다음의 거리 척도 사용

$$d_{pq} = \|\mathbf{f}(v_p) - \mathbf{f}(v_q)\| \quad (5.10)$$

이때 $\mathbf{f}(v) = (f_r(v), f_g(v), f_b(v))$

- f 를 r 개의 영역 $S = \{C_1, C_2, \dots, C_r\}$ 로 분할했을 때, 모든 쌍의 $D(C_i, C_j)$ 가 참이면 분할 S 는 거칠지도(저분할) 세밀하지도(과분할) 않은 적절한 상태
- 이런 분할을 어떻게 찾을 것인가? → 탐욕 알고리즘(greedy algorithm)으로 해결

5.3.1 최소 신장트리



(a) 해변 영상($\sigma=0.5$, $k=500$, $\min_area=50$)



(b) 콩 영상($\sigma=0.5$, $k=1,000$, $\min_area=100$)

그림 5-10 자연 영상의 분할 결과

5.3.2 정규화 절단

■ 유사도 그래프와 W_u 의 분할 품질 척도

- 거리 대신 유사도를 에지 가중치로 사용
- 예) 거리가 d 일 때 $9-d$ 로 유사도 계산

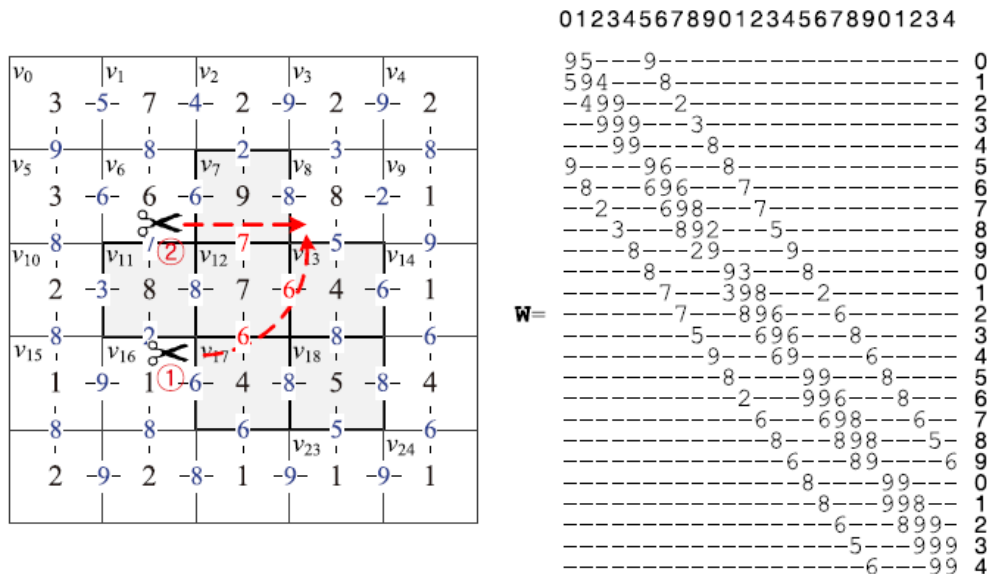


그림 5-11 유사도 그래프와 인접 행렬

- W_u 의 척도 cut [Wu93]

$$cut(C_1, C_2) = \sum_{p \in C_1, q \in C_2} w_{pq} \quad (5.11)$$

5.3.2 정규화 절단

■ Shi의 정규화 절단 [Shi2000]

- cut 의 문제점
 - 원래 연결요소를 작은 것과 큰 것으로 분할하는 경향
 - 두 연결요소 간의 에지 개수가 적으므로 cut 이 작아짐
 - 예) 분할 1의 cut 은 12, 분할 2는 7 → 하지만 직관적으로 분할 1이 우수
- Shi는 정규화 절단 $ncut$ 으로 확장하여 문제 해결

$$ncut(C_1, C_2) = \frac{cut(C_1, C_2)}{assoc(C_1, C)} + \frac{cut(C_2, C_1)}{assoc(C_2, C)} \quad (5.12)$$

이때 $assoc(C_i, C) = \sum_{p \in C_i, q \in C} w_{pq}$, $C = C_1 \cup C_2$

- $ncut$ 이 최소가 되는 분할을 찾는 문제는 NP-complete → 스펙트럴 군집화를 이용하여 근사해 구함

5.3.2 정규화 절단

■ 근사해 탐색 알고리즘

$$(\mathbf{D} - \mathbf{W})\mathbf{y}^T = \lambda \mathbf{D}\mathbf{y}^T \quad (5.13)$$

$$\begin{aligned} \mathbf{A}\mathbf{y}^T &= \lambda \mathbf{y}^T \\ \text{이때 } \mathbf{A} &= \mathbf{D}^{-\frac{1}{2}}(\mathbf{D} - \mathbf{W})\mathbf{D}^{-\frac{1}{2}} \end{aligned} \quad (5.14)$$

- \mathbf{W} 는 그림 5-11의 인접 행렬이고, \mathbf{D} 는 $d_{ii} = \sum_j w_{ij}$ 인 대각선 행렬

5.3.2 정규화 절단

■ 영상 분할

- 식 (5.14)의 고유값과 고유 벡터를 이용하여 영상 분할
- 실제로 쓰이는 유사도
 - 화소의 특징값(명암 또는 컬러)뿐 아니라 화소 사이의 거리도 같이 사용
 - 특징값이 비슷할수록 거리가 짧을수록 큼
 - 특징값이 크게 다르더라도 이웃이면 같은 영역이 된다거나 특징값이 비슷하더라도 멀면 다른 영역에 배정할 가능성 확보 ← 전역 최적화

$$S_{pq} = \begin{cases} e^{-\left(\frac{\|\mathbf{f}(v_p) - \mathbf{f}(v_q)\|^2}{\sigma_I} + \frac{\|\mathbf{x}(v_p) - \mathbf{x}(v_q)\|^2}{\sigma_X}\right)}, & \|\mathbf{x}(v_p) - \mathbf{x}(v_q)\| < r \text{ 이면} \\ 0, & \text{그렇지 않으면} \end{cases} \quad (5.15)$$

이때 $\mathbf{f}(v) = f(v)$ (명암 영상인 경우)

5.3.2 정규화 절단

■ 영상 분할 알고리즘

알고리즘 5-5 정규화 절단을 이용한 영상 분할

입력: 영상 $f(j, i)$, $0 \leq j \leq M-1$, $0 \leq i \leq N-1$

출력: 분할 결과 $S = \{C_1, C_2, \dots, C_p\}$ // C_i 는 연결요소

- 1 전체 노드 집합 V 를 하나의 연결요소 C 라고 한다.
- 2 C 의 유사도 행렬 W 를 계산한다.
- 3 식 (5.14)를 풀어, 고유값과 고유 벡터를 구한다.
- 4 두 번째 작은 고유값에 해당하는 고유 벡터를 이용하여 C 를 C_1 과 C_2 로 분할한다.
- 5 C_1 과 C_2 각각에 대해 추가 분할이 필요한지 판단하고, 그렇다면 그것을 C 로 놓고 2~5행을 재귀적으로 반복한다.

5.3.2 정규화 절단

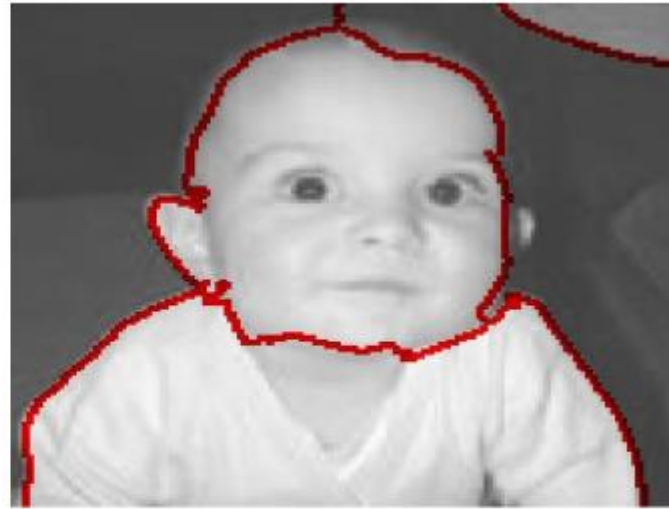


그림 5-12 정규화 절단을 적용한 영역 분할

논문 저자가 제공하는 Matlab 소스 코드를 <http://www.cis.upenn.edu/~jshi/software/>에서 받을 수 있다.

5.4 민시프트(Mean Shift)

■ 모드 탐색

- 샘플이 주어진 경우, 어떤 점이 속한 모드(봉우리)를 찾는 문제
- y 의 모드는?

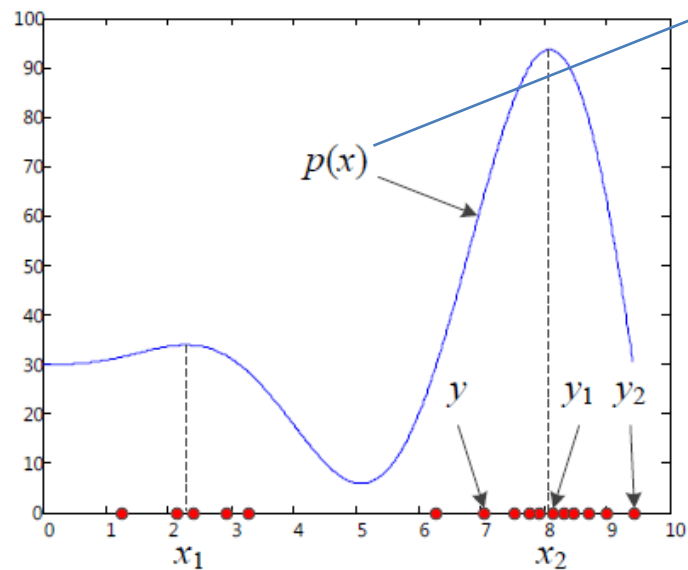
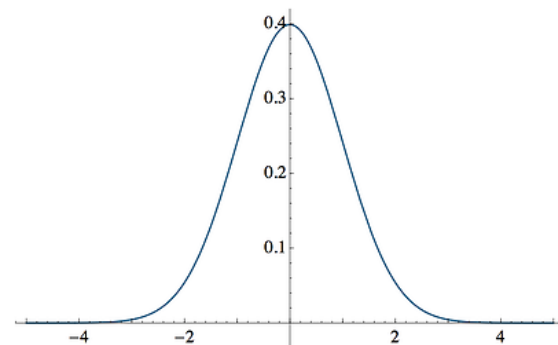


그림 5-13 모드 탐색

가우시안 분포의 확률밀도함수

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{\sigma^2}}$$



평균이 0이고 표준편차가 1인 정규분포의 확률밀도함수

5.4.1 군집화

■ 파젠 창(Pazen window)을 이용한 확률분포 추정

- h 는 커널의 폭 (클수록 많은 샘플이 창 안으로 들어와 매끄러운 함수가 생성됨)

$$p(\mathbf{x}) = \frac{1}{nh^d} \sum_{i=1}^n k\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right) \quad (5.16)$$

$$\text{평편한 커널: } k(\mathbf{x}) = \begin{cases} 1, & \|\mathbf{x}\| \leq 1 \\ 0, & \|\mathbf{x}\| > 1 \end{cases} \quad (5.17)$$

$$\text{가우시안 커널: } k(\mathbf{x}) = \begin{cases} e^{-\|\mathbf{x}\|^2}, & \|\mathbf{x}\| \leq 1 \\ 0, & \|\mathbf{x}\| > 1 \end{cases}$$

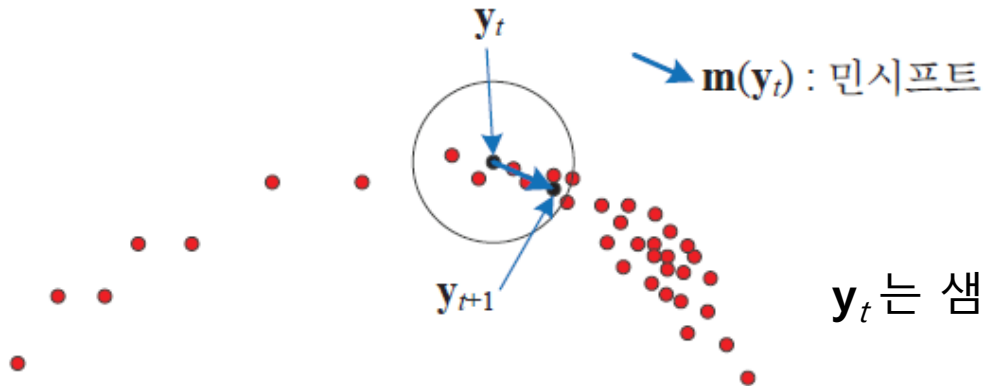
- 차원의 저주 발생
 - 차원 d 가 커질수록 기하급수적으로 메모리와 계산 시간 증가
- 명시적으로 확률분포 추정하려는 시도는 수학적으로 그럴싸하지만 비현실적임

5.4.1 군집화

■ 민시프트

- 우회적으로 소속(모드)을 결정하는 기법
- y 를 y_0 로 놓고 시작하여, 수렴할 때까지 $y_0 \rightarrow y_1 \rightarrow y_2 \rightarrow \dots$ 반복

$$y_{t+1} = \frac{\sum_{i=1}^n x_i k\left(\frac{x_i - y_t}{h}\right)}{\sum_{i=1}^n k\left(\frac{x_i - y_t}{h}\right)} \quad (5.18)$$



y_t 는 샘플이 밀집된 방향으로 이동

그림 5-14 현재 점 y_t 를 민시프트 $m(y_t)$ 가 y_{t+1} 로 이동시킴

5.4.1 군집화

- 바꾸어 쓰면,

$$\mathbf{y}_{t+1} = \mathbf{y}_t + \mathbf{m}(\mathbf{y}_t)$$

$$\text{이때 } \mathbf{m}(\mathbf{y}_t) = \mathbf{y}_{t+1} - \mathbf{y}_t$$

$$\begin{aligned} &= \frac{\sum_{i=1}^n \mathbf{x}_i k\left(\frac{\mathbf{x}_i - \mathbf{y}_t}{h}\right)}{\sum_{i=1}^n k\left(\frac{\mathbf{x}_i - \mathbf{y}_t}{h}\right)} - \mathbf{y}_t \\ &= \frac{\sum_{i=1}^n (\mathbf{x}_i - \mathbf{y}_t) k\left(\frac{\mathbf{x}_i - \mathbf{y}_t}{h}\right)}{\sum_{i=1}^n k\left(\frac{\mathbf{x}_i - \mathbf{y}_t}{h}\right)} \end{aligned} \quad (5.19)$$

- 민시프트를 이용한 군집화 알고리즘

- 모든 점에 대해 식 (5.19)로 수렴점 찾은 후, 군집 배정
- 뛰어난 군집화 알고리즘
- 군집 개수 자동 결정, 임의 모양 군집, 설정할 매개변수 적음

5.4.2 영상 분할과 스무딩

■ 민시프트를 영상 분할에 어떻게 적용할까? [Comaniciu2002]

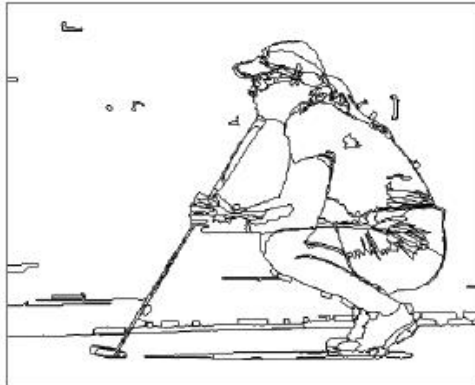
- 영상 f 를 샘플 집합 X 로 변형
 - 화소가 샘플이 됨

■ 몇 차원으로 표현할 것인가?

- 컬러만 표현하면 2.2절의 k -means와 비슷한 결과 (그림 5-6)
- 컬러 $\mathbf{x}^r = (r, g, b)$ 와 화소의 위치 $\mathbf{x}^s = (y, x)$ 를 같이 표현 \rightarrow 5차원 벡터 $\mathbf{x} = (r, g, b, y, x)$
- \mathbf{x}^r 과 \mathbf{x}^s 의 스케일 차이를 고려하여 커널을 별도로 표현

$$k(\mathbf{x}) = k\left(\frac{\mathbf{x}^s}{h_s}\right)k\left(\frac{\mathbf{x}^r}{h_r}\right) \quad (5.20)$$

5.4.2 영상 분할과 스무딩



(a) 원래 영상

(b) 영역의 경계를 표시

(c) 영역의 평균 컬러로 표시된 분할 영상

그림 5-15 민시프트(EDISON)로 분할한 영상

5.5 워터셰드(Watershed)

■ 워터셰드란?

- 워터셰드 (분수계) : 빗물이 안쪽과 바깥쪽 중 하나로 흐를 수 있는 점(붉은선)
- 유역 : 같은 호수로 빗물이 모이는 점들의 집합

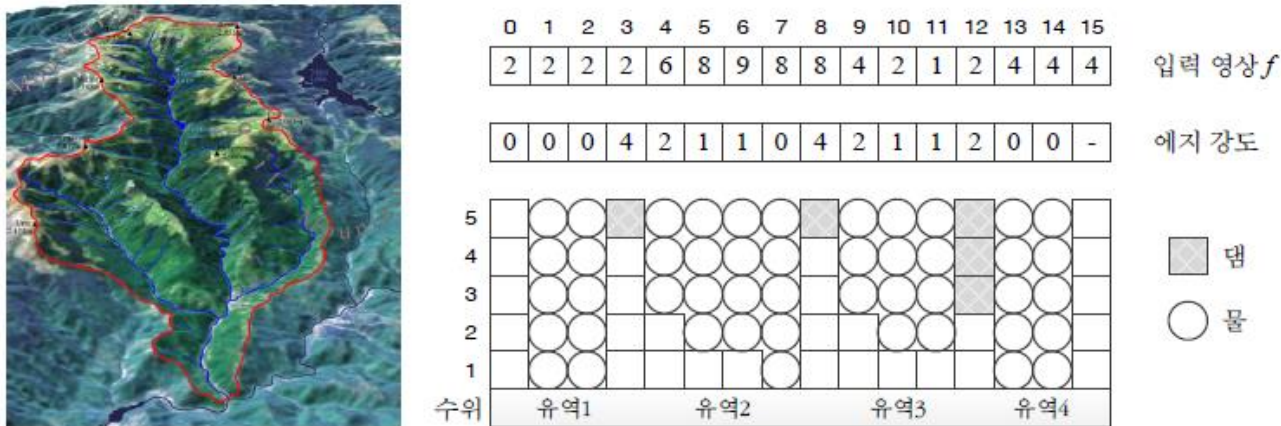


그림 5-16 워터셰드를 이용한 영상 분할의 원리

- 위치 3, 8, 12는 워터셰드, 1~2, 4~7, 9~11, 13~14는 유역

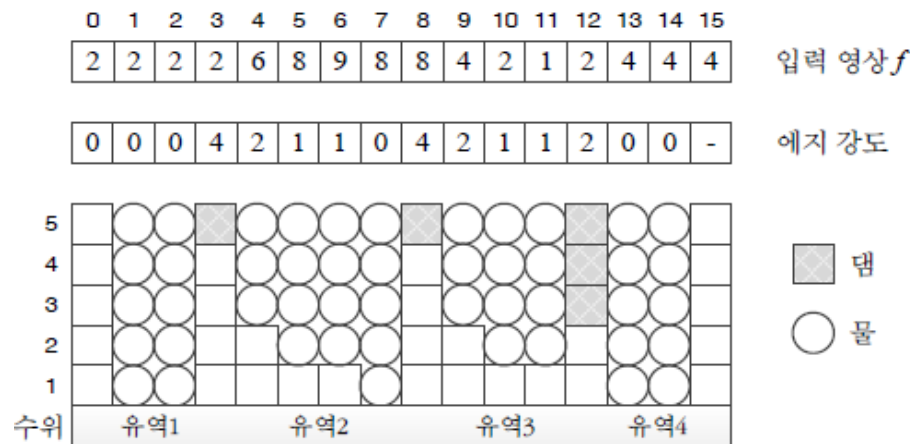
■ 영상 분할에 적용

- 에지 강도 맵의 워터셰드는 두 영역을 가르는 경계에 해당
- 에지 강도 맵에서 워터셰드를 어떻게 찾을 것인가?

5.5 워터셰드

■ 댐 건설 방법 [그림 5-16]

- 영상 분할 = 에지 강도 맵에서 워터셰드를 찾는 과정 → 댐 건설 방법
- 댐 = 워터셰드 = 영상 분할 경계면



- [그림 5-16]에서 물을 주입하면 수위 1에 물이 고인다 → 유역 1, 2, 4에 호수 생성
- 물을 더 넣어 수위 2까지 채우면 → 유역 3에 호수 생성
- 수위가 3이 되면 → 유역3과 4가 범람해 합쳐짐 → 댐이 필요!
-
- 수위가 5가 되면 → 유역 1과 2, 유역 2와 3, 유역 3과 4가 범람 → 각각 댐이 필요!
- 그리고 최고 수위가 되므로 알고리즘 멈춤

5.5 워터셰드

■ Meyer의 방법 [Meyer93]

- 우선순위 큐 (힙) 사용
 - 화소값이 낮을수록 우선순위 높음 (낮은 화소부터 조사가 이루어지도록)

알고리즘 5-9 워터셰드 영상 분할

입력 : 명암 영상 $f(j, i)$, $0 \leq j \leq M-1$, $0 \leq i \leq N-1$

출력 : 분할된 영상

```
1  f에 그래디언트 연산을 적용하여 에지 강도 맵을 구한다.
2  최저 영역을 찾아 각각에 서로 다른 번호를 매긴다. // 1, 2, 3, ...을 사용
3  번호가 매겨진 영역에 대해 이웃 화소를 에지 강도에 따라 우선순위 큐 q에 삽입한다.
4  while(q ≠ ∅) {
5      p = pop(q);
6      p의 이웃 중 번호가 매겨진 것을 조사한다.
7      if(그들의 번호가 모두 같으면) p에 그들과 같은 번호를 부여한다.
8      번호가 매겨지지 않은 p의 이웃을 모두 q에 삽입한다. // 이미 큐에 들어있는 것은 배제
9  }
10 번호가 매겨지지 않은 화소를 워터셰드로 취한다.
11 워터셰드로 영상을 분할한다.
```

5.6 대화식 물체 분할 (Interactive Object Segmentation)

■ 물체/배경 분할 (Object/Background Segmentation)



그림 5-17 물체/배경 분할의 응용

- 응용에 따라 관심있는 물체 하나만 잘라낼 필요
- 사용자가 초기 곡선 지정 → 대화식 시스템
- 사진에서 관심있는 물체를 오려내는 작업을 매팅(matting)
- 컴퓨터 비전과 컴퓨터 그래픽스의 협동

5.6.1 능동 외곽선

5.6.2 그래프 절단

5.6.1 능동 외곽선

■ 원리

- 초기 곡선에서 시작하여, 최적 상태(안정적인 에너지 상태)를 능동적으로 찾아감

■ 외곽선 표현

- 연속 공간에서는 $\mathbf{g}(s) = (y(s), x(s))$, $0 \leq s \leq 1$
- 디지털 공간에서는 $\mathbf{g}(s) = (y(s), x(s))$, $s = 0, 1, 2, \dots, n$

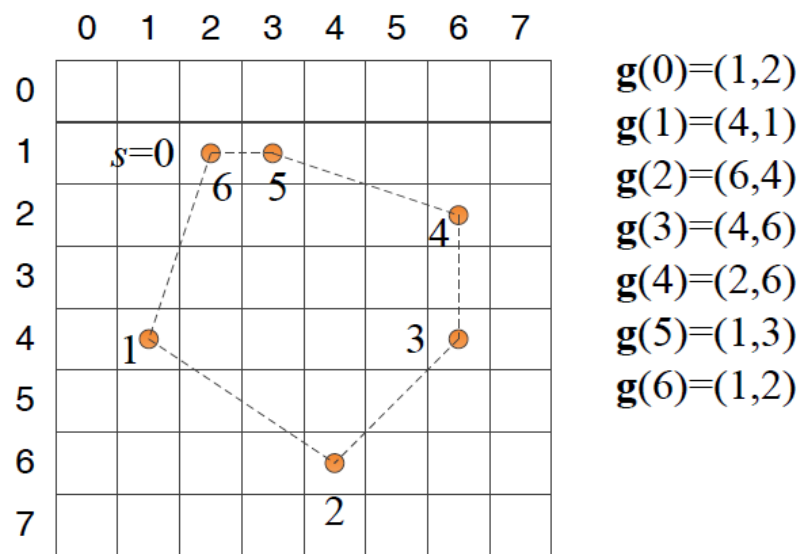


그림 5-18 스네이크를 표현하는 폐곡선 $\mathbf{g}(s)$

5.6.1 능동 외곽선

■ 스네이크 [Kass87]

- 최초의 능동 외곽선 기법
- 외곽선의 총 에너지 E^*

$$E^*(\mathbf{g}(s)) = \int_0^1 E(\mathbf{g}(s)) ds = \int_0^1 E_{internal}(\mathbf{g}(s)) + E_{image}(\mathbf{g}(s)) + E_{constraint}(\mathbf{g}(s)) ds \quad (5.21)$$

- E_{image} : 영상의 명암에 반응하는 항
 - 찾고자 하는 곳은 물체 경계이므로 에지 강도가 클수록 작은 값을 가짐
 - $E_{internal}$: 곡선의 내부 에너지
 - 물체의 경계는 매끄럽다는 전제하에, 곡률이 클수록 작은 값을 가짐
 - $E_{constraint}$: 사용자가 지정한 모양을 반영
- E^* 가 최소가 되는 곡선을 찾는 최적화 문제

$$\check{\mathbf{g}}(s) = \underset{\mathbf{g}(s)}{\operatorname{argmin}} E^*(\mathbf{g}(s)) \quad (5.25)$$

5.6.1 능동 외곽선

- 외곽선의 에너지 총합 $E^*(g(s))$ 의 계산

$$E_{image}(\mathbf{g}(s)) = -\|\nabla f(\mathbf{g}(s))\|^2 \quad (5.22)$$

$$E_{internal}(\mathbf{g}(s)) = \frac{\alpha(s)\|\mathbf{g}_s(s)\|^2 + \beta(s)\|\mathbf{g}_{ss}(s)\|^2}{2}$$

$$\begin{aligned} \circ \text{ 이때 } \|\mathbf{g}_s(s)\|^2 &\cong \|\mathbf{g}(s) - \mathbf{g}(s-1)\|^2 = (y(s) - y(s-1))^2 + (x(s) - x(s-1))^2 \\ \|\mathbf{g}_{ss}(s)\|^2 &\cong \|\mathbf{g}(s-1) - 2\mathbf{g}(s) + \mathbf{g}(s+1)\|^2 \\ &= (y(s-1) - 2y(s) + y(s+1))^2 + (x(s-1) - 2x(s) + x(s+1))^2 \end{aligned} \quad (5.23)$$

$$E^*(\mathbf{g}(s)) = \sum_{s=0}^n (E_{image}(\mathbf{g}(s)) + E_{internal}(\mathbf{g}(s))) \quad (5.24)$$

5.6.1 능동 외곽선

■ 최적해 탐색 알고리즘

- 초기 곡선 $\mathbf{g}_0(s)$ 에서 시작하여, 수렴할 때까지 $\mathbf{g}_0(s) \rightarrow \mathbf{g}_1(s) \rightarrow \mathbf{g}_2(s) \rightarrow \dots$ 반복
- 동적 프로그래밍 [Amini88]
- 탐욕 알고리즘 [Williams92]

■ Williams 알고리즘

- 개선된 에너지 수식 사용

$$E^*(\mathbf{g}(s)) = \sum_{s=0}^n (\alpha(s)E_{continuity}(\mathbf{g}(s)) + \beta(s)E_{curvature}(\mathbf{g}(s)) + \gamma(s)E_{image}(\mathbf{g}(s)))$$

$$\text{이때 } E_{continuity}(\mathbf{g}(s)) = \bar{d} - \|\mathbf{g}(s) - \mathbf{g}(s-1)\| (\bar{d} \text{는 평균 거리})$$

$$E_{curvature}(\mathbf{g}(s)) = \|\mathbf{g}(s-1) - 2\mathbf{g}(s) + \mathbf{g}(s+1)\|^2 \quad (5.26)$$

$$E_{image}(\mathbf{g}(s)) = \frac{min - mag}{max - min}$$

5.6.1 능동 외곽선

알고리즘 5-10 스네이크(탐욕 알고리즘)

입력 : 명암 영상 $f(j, i)$, $0 \leq j \leq M-1$, $0 \leq i \leq N-1$, 세 개의 임계값 T_1 , T_2 , T_3

출력 : 스네이크 곡선 $g(s)$, $s=0, 1, 2, \dots, n$

```
1 // 전처리
2 초기 곡선  $g(s)$ 를 설정한다. // 사용자 입력 또는 전처리에 의함
3 for( $s=0$  to  $n$ )  $\alpha(s)=\beta(s)=\gamma(s)=1.0$ ; // 가중치 설정
4
5 while(TRUE) {
6 // 스네이크의 이동
7    $moved=0$ ; // 스네이크가 이동한 양
8   for( $s=0$  to  $n$ ) { // 스네이크 곡선 상의 점 각각에 대해
9     for( $g(s)$ 의 이웃점 각각에 대해) // 자기 자신과 8-이웃을 포함한 9개 점
10       식(5.26)으로 에너지  $E^*$ 를 구한다.
11       if(8~9행에서 구한 최소 에너지 점이  $g(s)$ 와 다르면) {
12          $g(s)$ 를 그 점으로 이동시킨다.
13          $moved++$ ;
14       }
15   }
16   if( $moved < T_3$ ) break; // 뱀의 움직임이 아주 작으면 수렴으로 보고 끝낸다.
17
18 // 코너를 찾고, 다음 반복에서 코너의 곡률을 무시
19 for( $s=0$  to  $n-1$ )  $c_s = \left\| \frac{\mathbf{u}_s}{\|\mathbf{u}_s\|} - \frac{\mathbf{u}_{s+1}}{\|\mathbf{u}_{s+1}\|} \right\|^p$ ; // 곡률 계산( $\mathbf{u}_s = (g(s) - g(s-1))$ )
20 for( $s=0$  to  $n-1$ )
21   if(( $c_s > c_{s-1}$  and  $c_s > c_{s+1}$ ) and ( $c_s > T_1$ ) and ( $edge\_mag(g(s)) > T_2$ ))  $\beta(s)=0$ ; // 코너로 간주
22 }
```

$g_t(s) \rightarrow g_{t+1}(s)$

코너에 붙들어
두는 효과

5.6.1 능동 외곽선

■ 스네이크보다 진보한 현대적인 기법 등장

- 지능 가위: 편리한 사용자 인터페이스 제공 [Mortenson98]

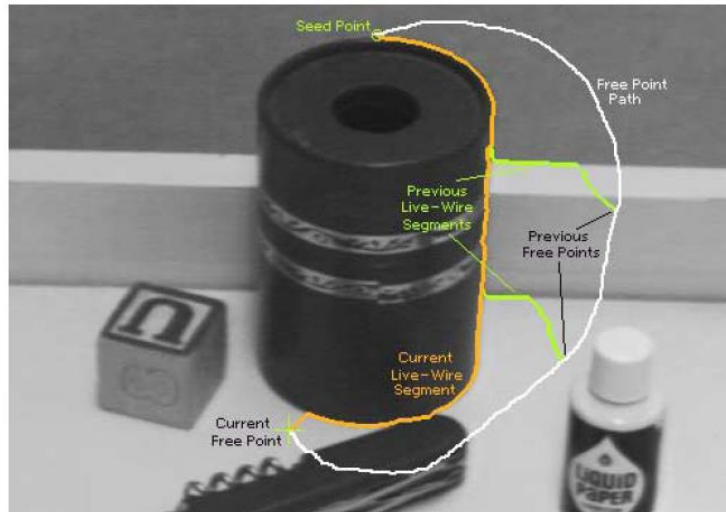


그림 5-19 지능 가위의 동작

- 레벨 셋
- Grab Cut

5.6.2 그래프 절단 (Graph Cut)

■ 네트워크 흐름 문제 [Ahuja93]

- 교통 흐름, 전기 흐름 등 많은 응용
- S 에서 출발하여 T 로 흐르는 최대 용량은?
 - 병목으로 인해 어떤 에지는 최대 용량 발휘 못함
 - 아래 그래프의 **최대 흐름**은 6

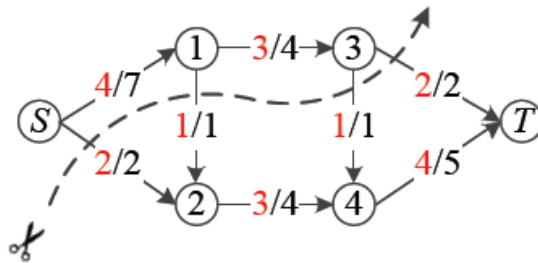
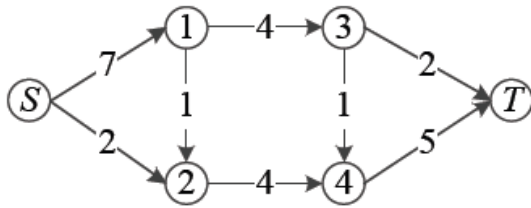


그림 5-20 네트워크 흐름과 그래프 절단

5.6.2 Graph Cut

■ 영상 분할과 어떻게 관련되나?

■ 영상을 그래프로 변환

- 영상의 화소를 노드로 간주하고, 유사도를 에지 가중치로 설정

- 예) [그림 5-20]의 절단은 최소 그래프 절단임

- $\{v_1, v_3\}$ 과 $\{v_2, v_4\}$ 로 분할

- 이 절단은 최적의 영상 분할임

■ 최소 절단(min-cut)을 어떻게 찾을 것인가?

- Ford의 정리[Ford62] : 최소 절단은 최대 흐름(max-flow)상태에서 포화된 에지와 같다.

- 최대 흐름 찾는 효율적인 알고리즘 존재

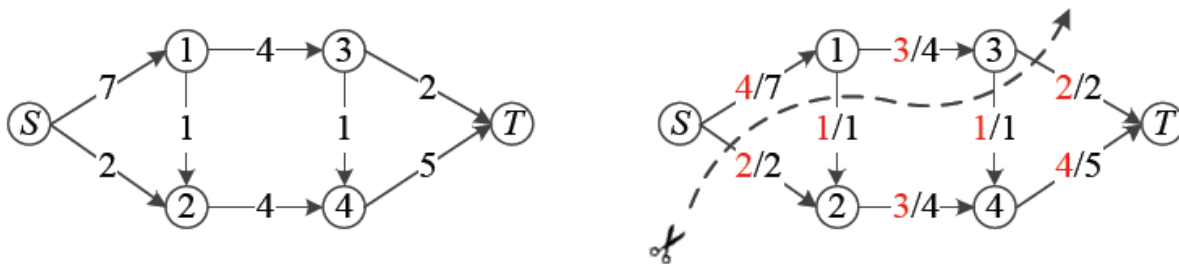


그림 5-20 네트워크 흐름과 그래프 절단

절단 비용 = 6,

다른 절단은 모두 6보다 크다

5.6.2 Graph Cut

■ 영상 분할 알고리즘

- 최대 흐름을 찾으면, Ford의 정리에 따라 그것이 바로 최소 절단이다.
- 최소 절단은 최적의 영상 분할 정보를 가진다.

알고리즘 5-11 그래프 절단을 이용한 영상 분할

입력 : 명암 영상 $f(j, i)$, $0 \leq j \leq M-1$, $0 \leq i \leq N-1$

출력 : 물체 $\{C_{object1}, C_{object2}, \dots\}$ 와 배경 $\{C_{background1}, C_{background2}, \dots\}$ // C 는 연결요소

- 1 f 로부터 그래프 G 를 구축한다.
- 2 G 로부터 최소 절단을 찾는다.
- 3 최소 절단을 영상 분할 결과인 C_{object} 와 $C_{background}$ 로 변환한다.

5.6.2 Graph Cut

- 그래프 $G=(V+\{S, T\}, E)$ 를 구축하는 방법 (알고리즘 5-11의 1행)
 - 에지 가중치(유사도)는 아래 수식 사용

$$w_{pq} = e^{-\frac{(f(v_p) - f(v_q))^2}{2\sigma^2}} d(v_p, v_q) \quad (5.27)$$

2	^O 8	3
2	7	9
1	1	^B 2

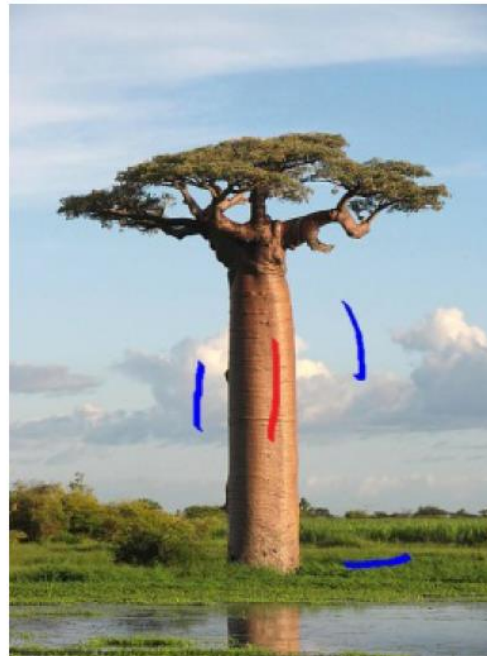
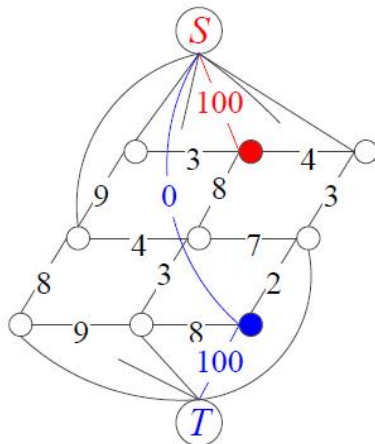


그림 5-21 영상 f 와 그래프 표현(에지 가중치는 [예제 5-2] 참고)

5.6.2 Graph Cut

■ 사용자가 지정한 물체(O)와 배경(B) 정보 반영

- O로 지정된 화소는 노드 S 와 큰 값, 노드 T 와 0
- B로 지정된 화소는 노드 S 와 0, 노드 T 와 큰 값
- 나머지 화소 p 는

$$\text{에지 } (p, S) \text{의 가중치 } w_{pS} = \lambda \cdot (-\ln \text{prob}(f(v_p) \mid \text{'background'})) \quad (5.28)$$

$$\text{에지 } (p, T) \text{의 가중치 } w_{pT} = \lambda \cdot (-\ln \text{prob}(f(v_p) \mid \text{'object'})) \quad (5.29)$$

5.6.2 Graph Cut

예제 5-2 에지 가중치 계산

[그림 5-21] 예에서는 화소를 연결하는 에지는 $9 - |f(v_i) - f(v_j)|$ 라는 가중치를 부여하였다. 예를 들어, 8과 7을 갖는 두 화소의 유사도는 $9 - |8 - 7| = 8$ 이다. 식 (5.27)을 사용하여 계산한 결과는 다음과 같이 0.6065이다. 이때 두 화소 간의 거리는 4-이웃이므로 1이고, $\sigma = 1.0$ 을 사용하였다. 2와 8을 갖는 화소의 에지는 0.0이 되어 앞의 에지에 비해 유사도가 훨씬 낮음을 알 수 있다.

$$e^{-\frac{(8-7)^2}{2\sigma^2}} d(v_p, v_q) = e^{-\frac{1}{2}} = 0.6065$$

S와 O(물체)로 지정된 빨간색 노드 간의 에지는 충분히 큰 100이라는 값을 부여하였다. 실제로는 계산 효율을 위해 포화 상태가 되지 않는 수준을 약간 넘는 값으로 설정한다. S와 B로 지정된 파란색 노드를 연결하는 에지의 값은 0이다.

명암7을 갖는 중앙에 있는 화소를 v_p 라고 하고, 에지 (v_p, T) 의 가중치 w_{pT} 를 계산해 보자. 이를 위해서는 식 (5.29)의 $prob(f(v_p) | \text{'object'})$ 를 알아야 한다. [그림 5-21]에서 사용자가 빨간색으로 표시한 화소들의 히스토그램을 분석하는 방법 등을 이용하여 $prob(7 | \text{'object'}) = 0.3$ 이 되었다고 가정하자. 이때 $\lambda = 1.0$ 으로 설정하면, $w_{pT} = 1.204$ 가 된다. 이제 v_p 를 S에 연결해 주는 에지 (v_p, S) 의 가중치 w_{pS} 를 계산해 보자. 적절한 방법으로 $prob(7 | \text{'background'}) = 0.01$ 이 되었다면, $w_{pT} = 4.6052$ 가 된다. 명암7은 배경보다 물체가 될 확률이 훨씬 크다. 그런 면에서 확률 0.01은 0.3보다 훨씬 작으므로 비교적 합리적이다. 명암7을 갖는 중앙에 위치한 화소는 S와 4.6052, T와 1.204라는 가중치로 연결된다.²⁰ 결국 최소 절단 알고리즘은 T와 연결된 에지를 끊을 가능성이 높다.



GMM

5.6.2 Graph Cut

■ 최소 절단 탐색 (2행)

- [Ford62]와 같은 일반적인 알고리즘 대신, 영상의 특성을 감안한 [Boykov2004] 이용
- 예) 그림 5-22에서 얇은 선이 최소 절단에 해당하는 에지

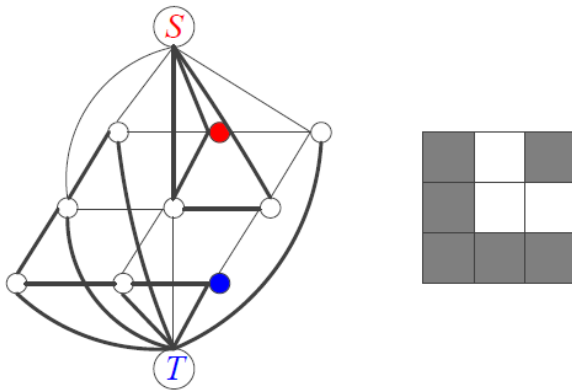


그림 5-22 최소 절단(굵은 선은 절단 되지 않은 에지)과 영상 분할 결과

■ 최소 절단을 영상 분할로 변환 (3행)

- 모든 화소는 S 와 T 중의 하나에만 굵은 선으로 연결됨
- S 에 굵은 선으로 연결된 노드는 물체, T 에 굵은 선으로 연결된 노드는 배경으로 구분

5.6.2 Graph Cut

■ 그래프 절단 알고리즘의 특성

- 두 종류의 에지는 상호 보완적으로 동작
 - $\{S, T\}$ 와 연결된 에지는 사용자가 지정한 정보에 충실
 - 화소 노드를 연결하는 에지는 영상이 가진 정보에 충실

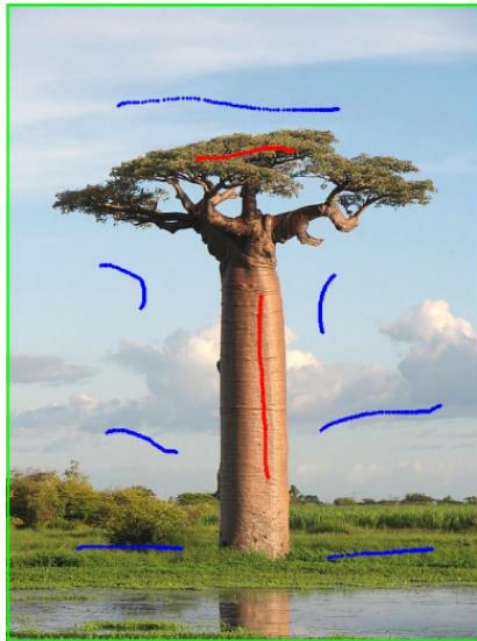


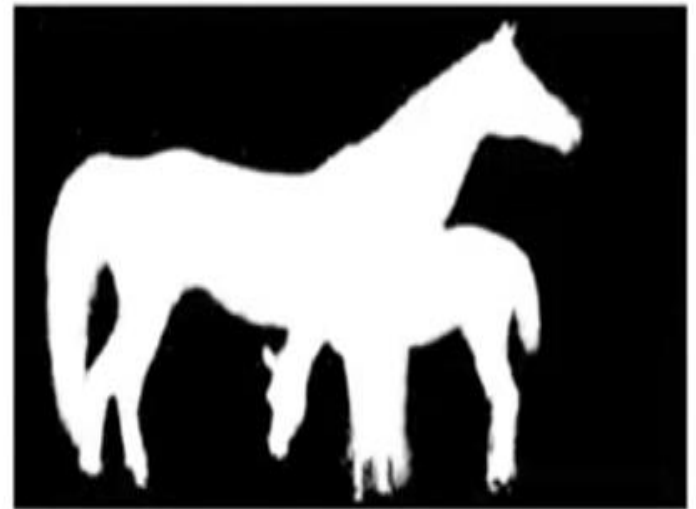
그림 5-23 GrabCut으로 오려낸 물체

5.6.2 Graph Cut

- 같은 영역(배경 또는 전경)에 속하는 픽셀들의 값의 유사도는 높다는 점에서 착안



원본 영상
(파랑색 : FG, 초록색 : BG)

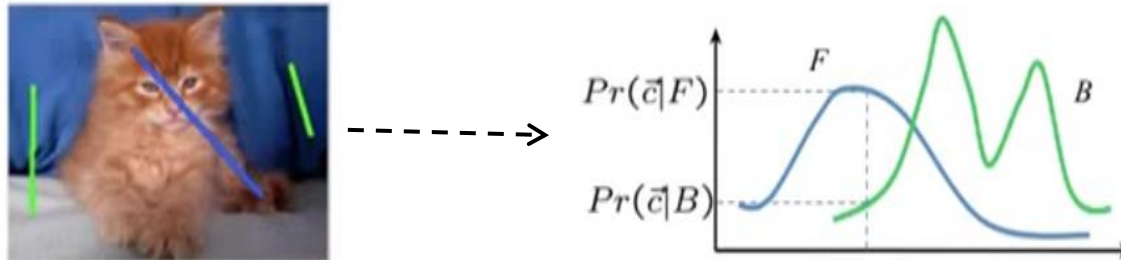


Alpha-Matte

5.6.2 Graph Cut

■ Step 1 - Feature Distribution Estimation

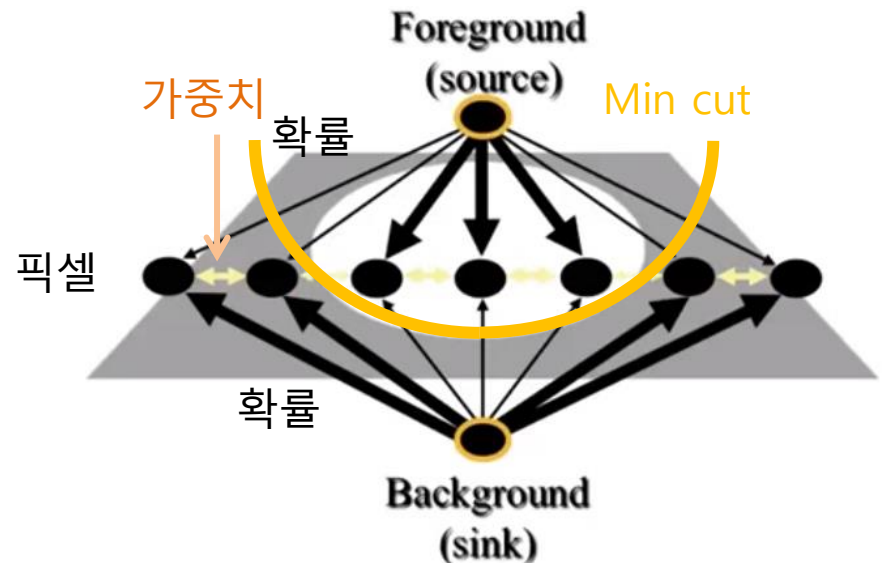
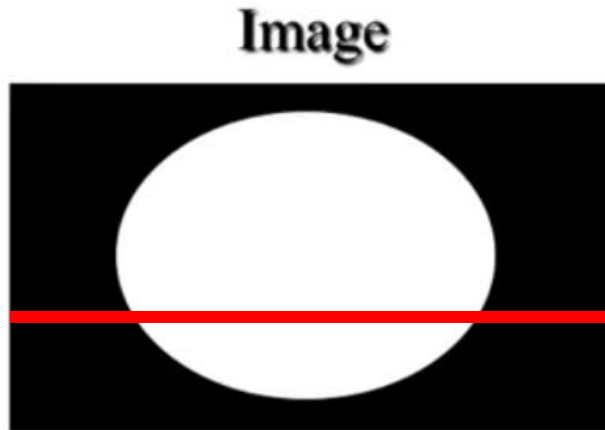
- 사용자가 입력한 데이터에 대한 grey-level 분포를 계산함
- 계산된 분포를 이용하면 각각의 픽셀에 전경 또는 배경일 확률이 할당됨



$$P_F(x) = \frac{Pr(\vec{c}_x|F)}{Pr(\vec{c}_x|F) + Pr(\vec{c}_x|B)}$$

5.6.2 Graph Cut

- Step 2 - Weight Estimation and Min Cut
- 픽셀 간의 가중치
 - 같은 영역에 속하는 픽셀 값은 유사하다는 점에서 착안
 - 픽셀 값의 유사도가 낮을수록 가중치는 낮아짐

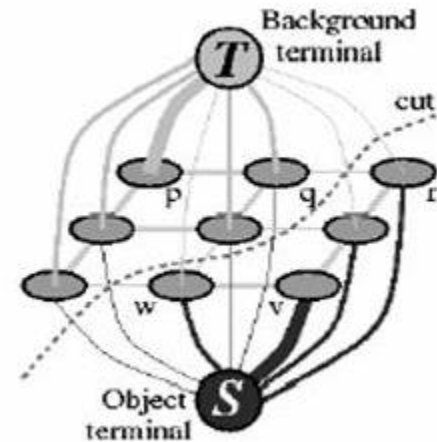
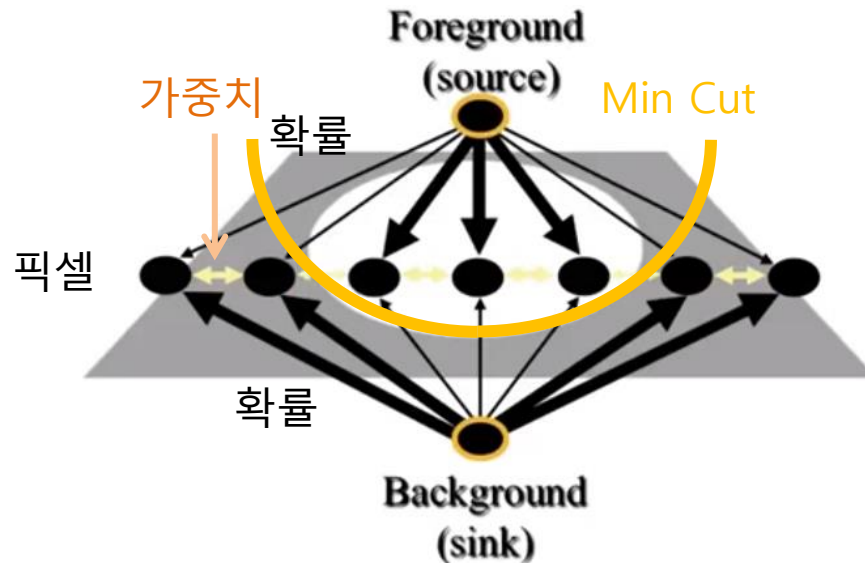


5.6.2 Graph Cut

■ Step 2 - Weight Estimation and Min Cut

■ Min Cut

- 에지를 지날 때 에지 값들의 총합이 가장 작은 값이 되도록 함
- 총합이 가장 작은 것이 가장 잘 된 segmentation



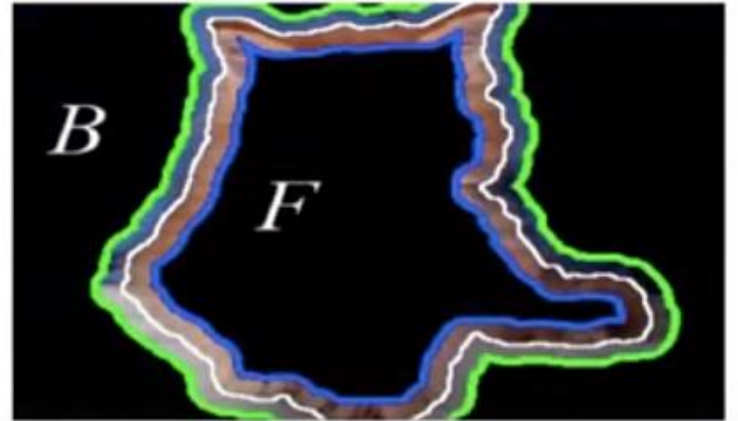
5.6.2 Graph Cut

■ Step3 – Refine

- Step2 결과 경계선에서 안쪽을 전경, 바깥쪽을 배경 데이터
- 새로운 데이터로 분포를 업데이트
- 새롭게 만들어진 분포로 경계 영역의 픽셀의 확률 계산 후 경계선 생성



Step2 결과



Step2 결과를 이용한 수정

5.6.2 Graph Cut



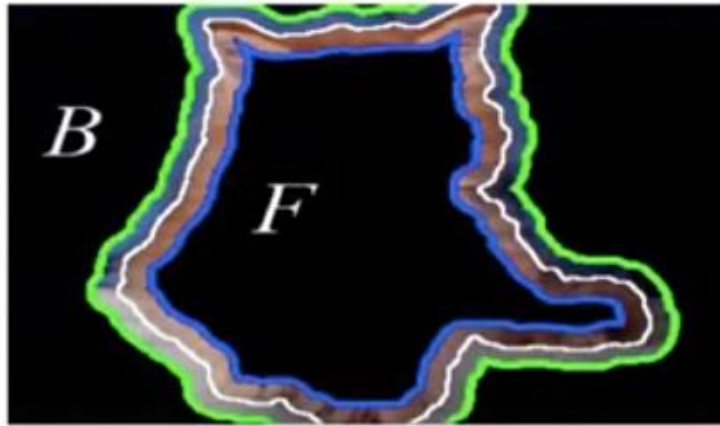
원본 영상
(파랑색 : FG, 초록색 : BG)

The result image of
Alpha-Matte

The composite image

5.6.2 Graph Cut

- 위의 그림처럼 FG 경계선의 내부를 전부 FG로 판단
- FG 경계선 안쪽에 배경이 포함되어 있더라도 FG로 판단될 수 있음
- Graph Cut에 경우, 오른쪽 사진의 FG 경계선의 내부 배경(노란색 원)까지 객체로 판단될 수 있음



5.6.2 Grab Cut

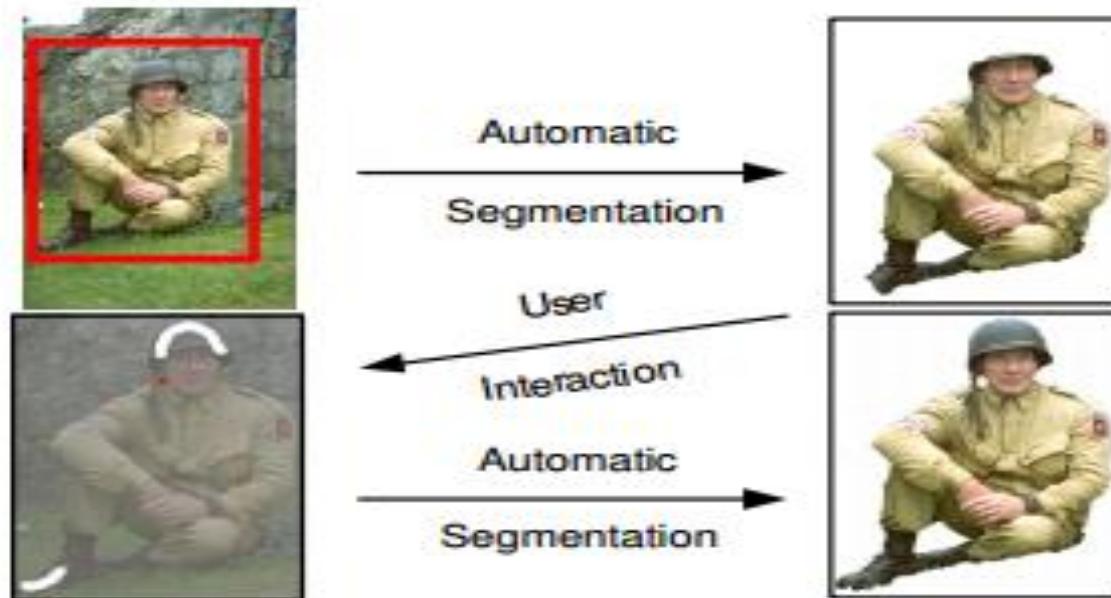
- Based on Graph Cut
- Graph Cut과 다른 UI를 통해 전경과 배경에 대한 더 많은 정보를 제공
- 불완전한 분할인 경우, 사용자 입력으로 상호작용 가능



5.6.2 Grab Cut

■ 간단한 사용자 상호작용

- 1) 전경이 포함된 사각형 그리기
- 2) 결과가 좋지 않을 경우 전경 또는 배경 브러싱



5.6.2 Grab Cut 사용방법

1) 사용자 관심 영역 정의

- 사각형 영역을 마우스 드래깅으로 손쉽게 정의, 보정 영역 스케치 가능
- 사각형 외부의 모든 픽셀은 배경으로 판단
- 사용자 입력은 hard-labeling으로 변하지 않음

2) 초기 분할

- 사용자가 입력한 데이터에 의존하여 초기 labeling 실행
- 분할 결과가 만족스럽지 못할 경우 3단계 실행



5.6.2 Grab Cut 사용방법

3) 보정 영역 스케치

- 사용자가 전경과 배경 부분을 대략적으로 스케치
- 사용자 입력은 hard-labeling으로 변하지 않음

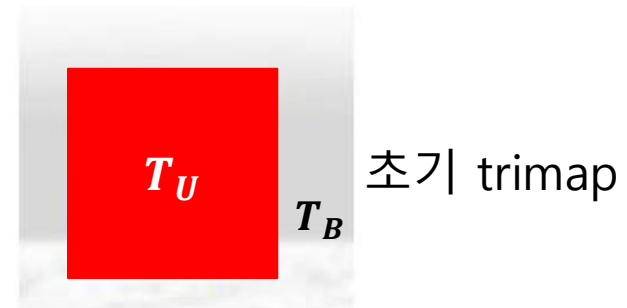
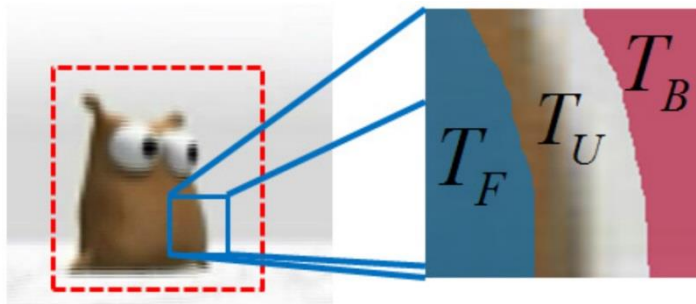
4) 보정 분할

- 분할 결과가 만족스럽지 못할 경우 3단계 반복



5.6.2 Grab Cut 초기화

- 사용자 입력 데이터에 의존하여 초기화
 - trimap 형성 : 전경(T_F), 배경(T_B), 알 수 없는 영역(T_U)으로 구성
 - 사각형 외부 T_B 영역에 포함되는 픽셀들 $\alpha_n = 0$
 - 배경으로 고정
 - 사각형 내부 T_U 영역에 포함되는 픽셀들 $\alpha_n = 1$
 - 초기 전경, 업데이트
 - 각 T_B, T_U 영역에 대한 GMM 초기화

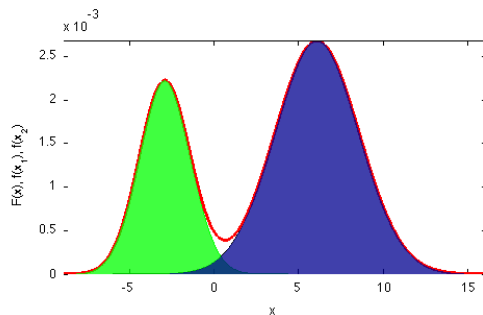


5.6.2 Grab Cut 반복 알고리즘

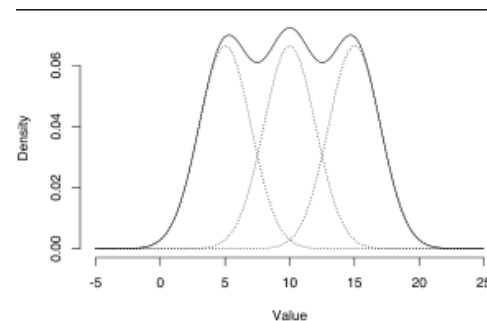
1) 전경과 배경에 대한 칼라 분포 GMM 재구성

- 초기 1회에는 이미 구성되어있으므로 생략함
- Step - 2를 거친 T_U 영역의 픽셀들을 포함하여 GMM 재구성
 - 주로, $K=5$ 사용

2) T_U 영역의 각 픽셀에 대해 $(K_B + K_F)$ 개의 가우시안 요소 중 소속 확률이 가장 높은 곳에 픽셀을 할당



배경 분포



전경 분포

5.6.2 Grab Cut 반복 알고리즘

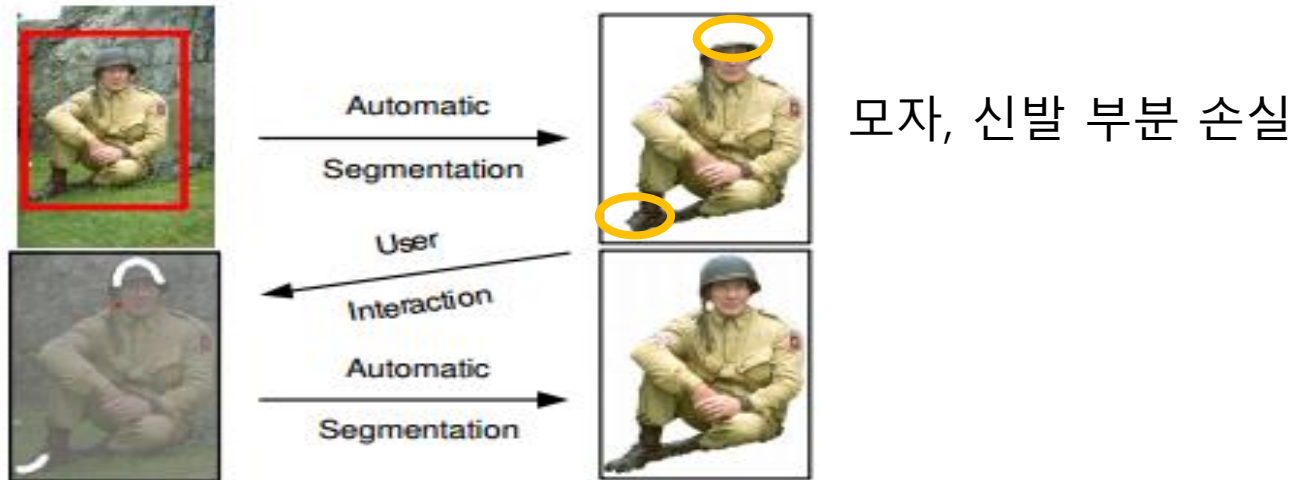
3) Step - 1, 2 에서 추정된 값으로 Min Cut 수행

- 에너지 함수(E)를 최소화하는 분할을 추정
- 수식 : $\min_{\{\alpha_n: n \in T_U\}} \min_{\mathbf{k}} \mathbf{E}(\underline{\alpha}, \mathbf{k}, \underline{\theta}, \mathbf{z}).$
- 에너지 함수의 최소값이 수렴할 때까지 Step - 1~3반복

5.6.2 Grab Cut 사용자 상호작용

■ Segmentation의 결과가 불완전할 경우

- Graph Cut의 사용자 입력과 같이 드래깅
- 사용자 입력으로 알게 된 데이터를 포함하여 반복 알고리즘을 실행



5.6.2 Grab Cut 결과 영상

사용자 입력



사용자 입력

No User
Interaction



결과 영상



결과 영상 확대



5.6.2 Grab Cut 실제 영상 테스트

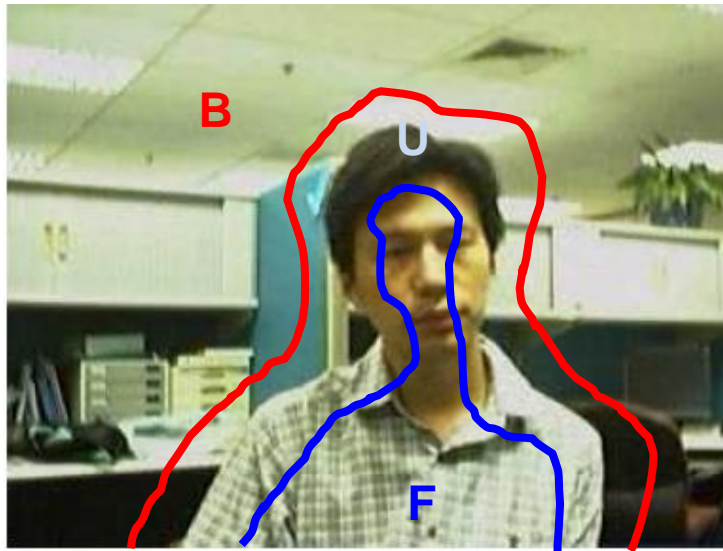


5.6.2 Background Cut

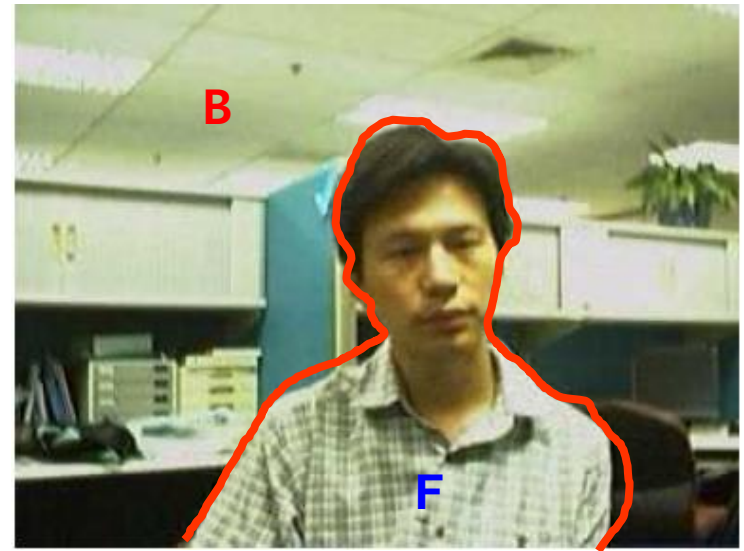
- 전제 : 동영상의 여러 프레임을 이용하여 배경 이미지를 생성할 수 있음
- Gibbs energy 수식 최소화

$$E(X) = \underbrace{\sum_{r \in \mathcal{V}} E_1(x_r)}_{\text{Color Term}} + \lambda \underbrace{\sum_{(r,s) \in \mathcal{E}} E_2(x_r, x_s)}_{\text{Contrast Term}}$$

- Color term : 2개의 GMM을 이용 전경과 배경 분리
 - Contrast term : 거리 및 유사도 이용, 불확실한 부분 분리
- } min-cut 사용



(1) Color term 역할

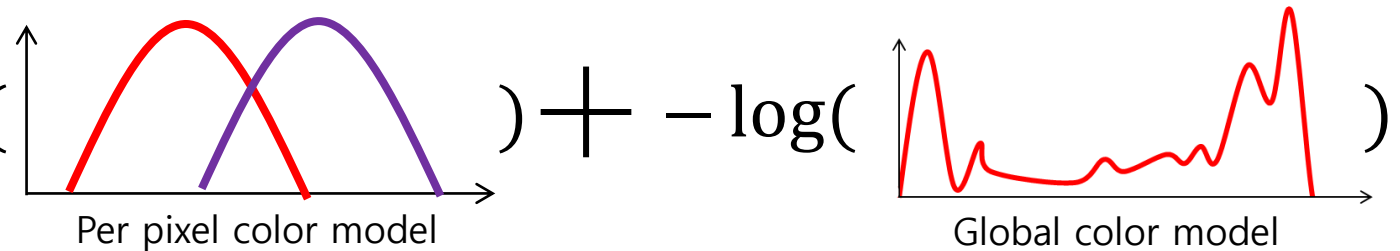


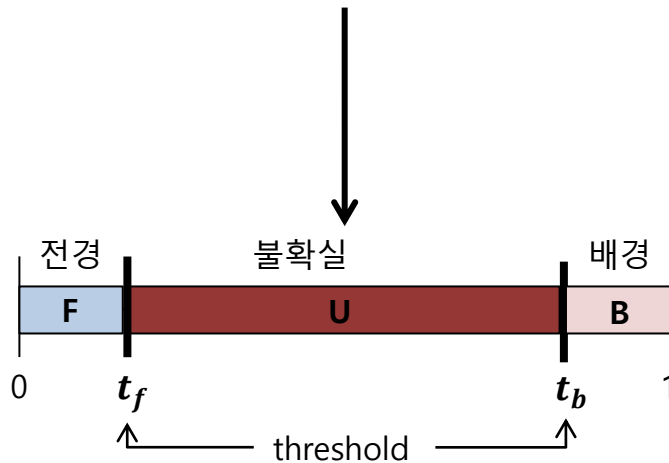
(2) Contrast term 역할

5.6.2 Background Cut – Color term

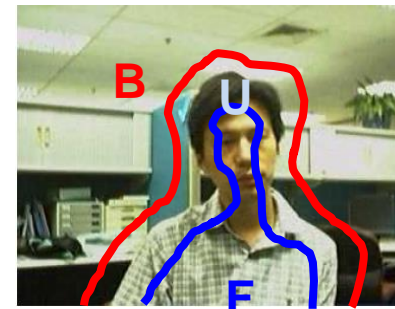
■ 배경 확률 모델 생성

- 각 픽셀에 대해 확률을 부여함

$$E_1 = -\log(\text{Per pixel color model}) + -\log(\text{Global color model})$$




- Global **background** color model
 - 배경 이미지로 학습
- Global **foreground** color model
 - 명확히 전경인 픽셀들로 학습



라벨링

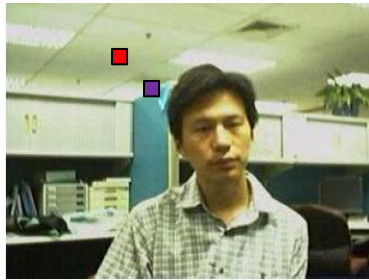
5.6.2 Background Cut – Color term

■ Per pixel color model

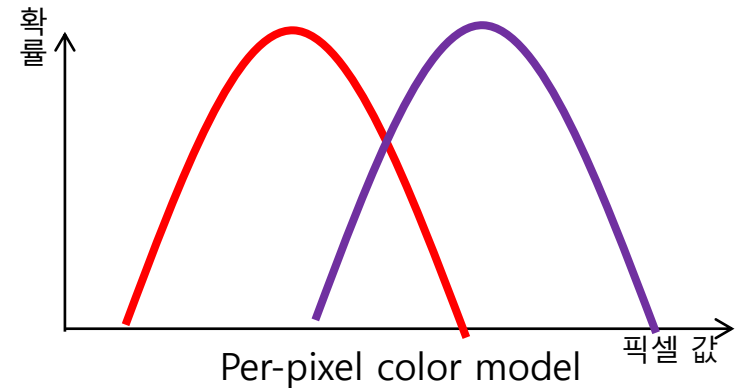
- 각각의 픽셀에 대해 (같은 위치, 시간축 상) 픽셀들을 Gaussian 분포로 표현 – Temporal GMM
- 입력 영상의 각각 픽셀에 대해 temporal GMM을 적용해 확률 계산



배경

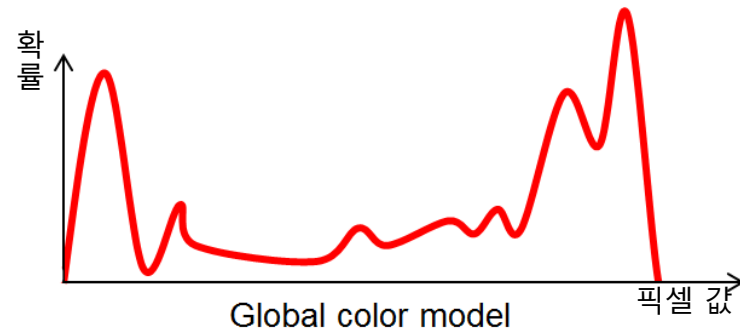
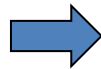
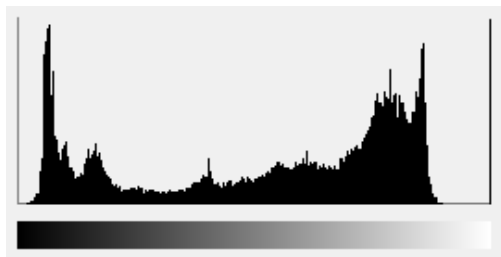


입력



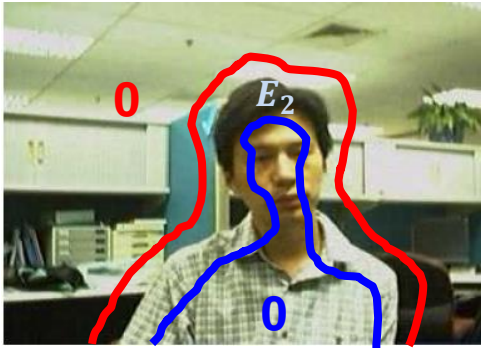
■ Global color model

- Temporal GMM에 의해 만들어진 배경 이미지의 픽셀들을 Gaussian 분포로 표현 - Spatial GMM
- 입력 영상의 각각 픽셀에 대해 spatial GMM을 적용해 확률 계산



5.6.2 Background Cut – Contrast term

- E_2 : 경계 영역에서 값 존재



E_2 의 최소값 선택

이웃 픽셀과의 차이 값 이용

If 차이 값 \uparrow , $E_2 \downarrow$

분류 정확성 \uparrow

- 전경/배경 경계 영역의 여러 경우에 대해 값을 계산함

- 경우 1 : 이웃 픽셀과 값이 비슷할 때



차이 값 \downarrow , $E_2 \uparrow$

선택 확률 \downarrow

5.6.2 Background Cut – Contrast term

- 배경, 전경 구분선의 모든 경우를 구함

- 경우 2 : 이웃 픽셀과 값이 비슷할 때



차이 값 \downarrow , $E_2 \uparrow$
선택 확률 \downarrow

- 경우 3 : 이웃 픽셀과 값이 차이가 클 때



차이 값 \uparrow , $E_2 \downarrow$
선택 확률 \uparrow

5.6.2 Background Cut – 예제

205	190	200	190
190	140	210	165
110	140	140	135
35	50	60	55

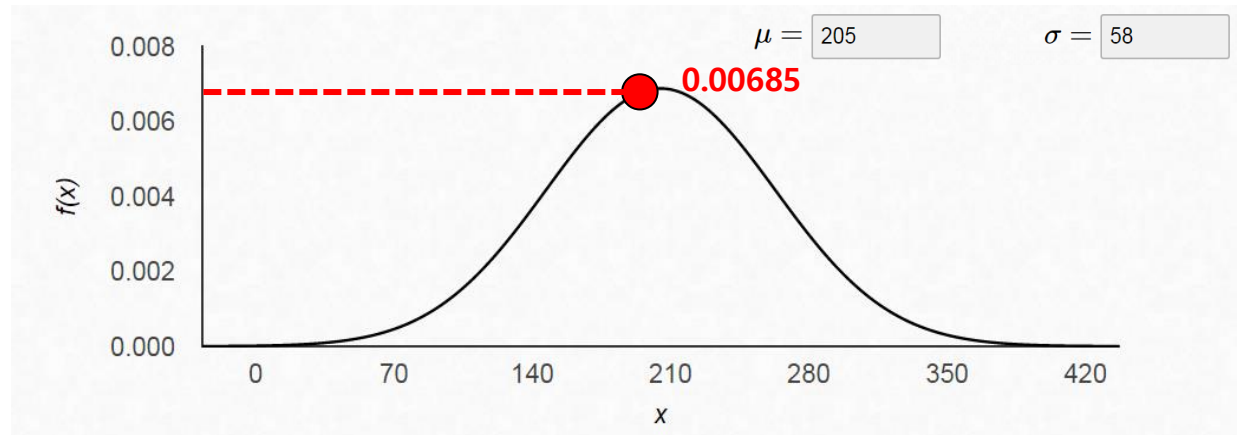
배경 영상

- 표준 편차 : 58

200	190	195	195
180	150	200	170
115	145	40	140
30	100	180	60

입력 영상

정답			
	배	경	
	전	경	



per-pixel color model

5.6.2 Background Cut – 예제

205	190	200	190
190	140	210	165
110	140	140	135
35	50	60	55

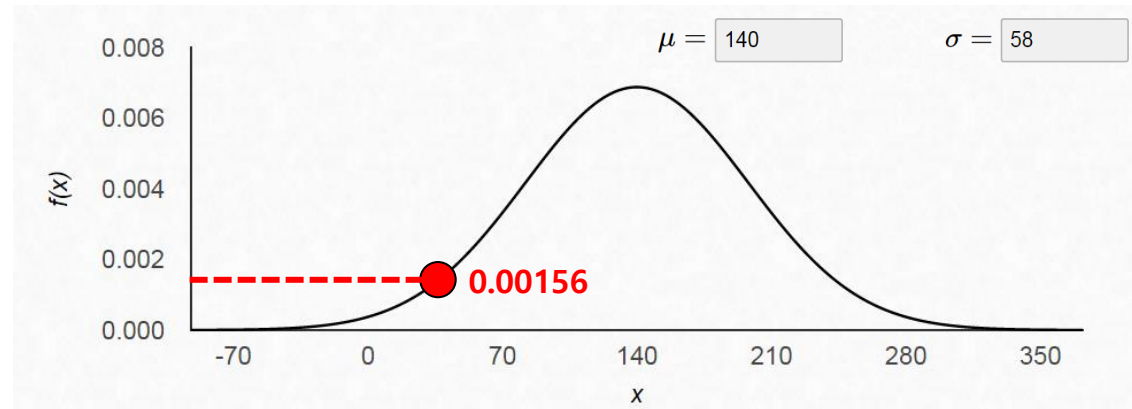
배경 영상

- 표준 편차 : 58

200	190	195	195
180	150	200	170
115	145	40	140
30	100	180	60

입력 영상

정답			
	배	경	
	전	경	



per-pixel color model

5.6.2 Background Cut – 예제

205	190	200	190
190	140	210	165
110	140	140	135
35	50	60	55

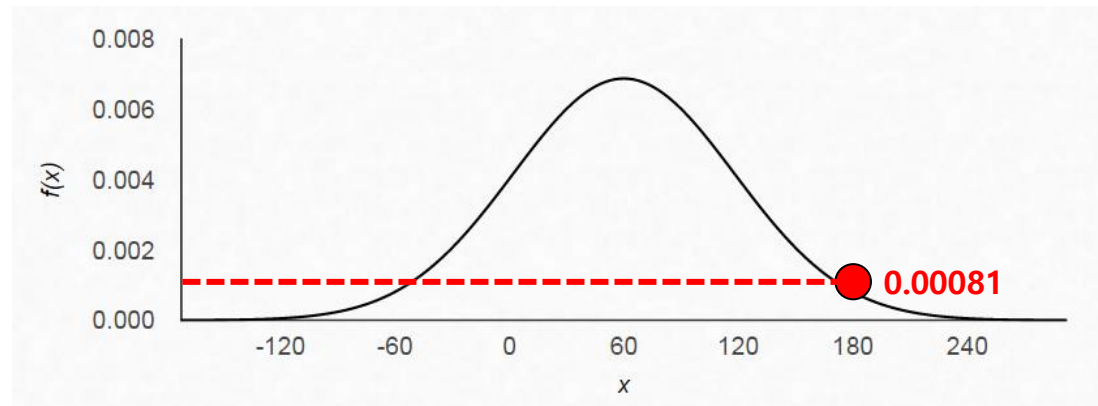
배경 영상

- 표준 편차 : 58

200	190	195	195
180	150	200	170
115	145	40	140
30	100	180	60

입력 영상

정답			
	배	경	
	전	경	



per-pixel color model

5.6.2 Background Cut – 예제

205	190	200	190
190	140	210	165
110	140	140	135
35	50	60	55

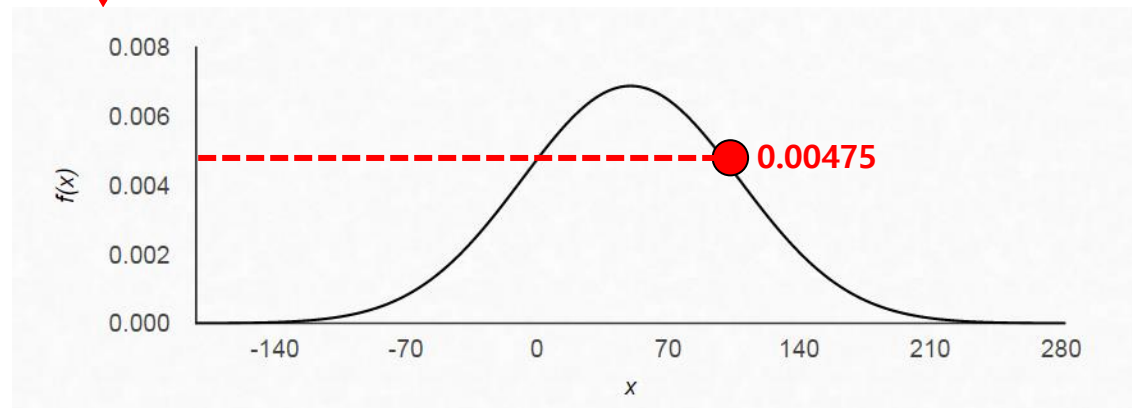
배경 영상

- 표준 편차 : 58

200	190	195	195
180	150	200	170
115	145	40	140
30	100	180	60

입력 영상

정답			
	배	경	
	전	경	



5.6.2 Background Cut – 예제

205	190	200	190
190	140	210	165
110	140	140	135
35	50	60	55

배경 영상

200	190	195	195
180	150	200	170
115	145	40	140
30	100	180	60

입력 영상

정답			
	배	경	
	전	경	

- 표준 편차 : 58

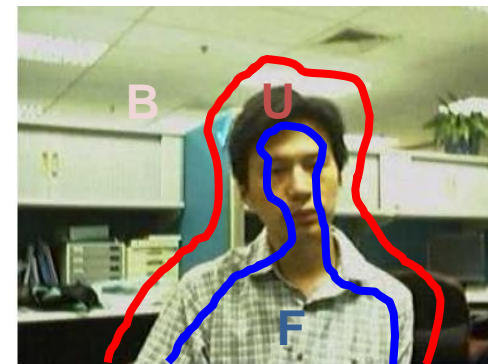
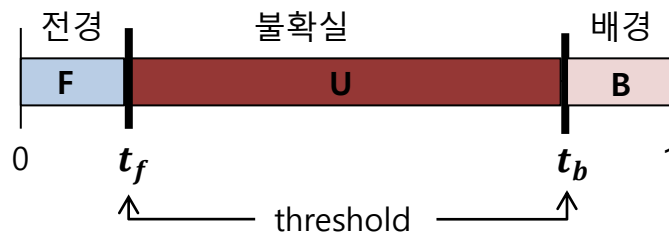
0.00685	0.00688	0.00685	0.00688
0.00678	0.00678	0.00678	0.00688
0.00688	0.00688	0.00156	0.00688
0.00688	0.00475	0.00081	0.00688

입력 영상의 per-pixel 확률

5.6.2 Background Cut – 예제

0.00685	0.00688	0.00685	0.00688
0.00678	0.00678	0.00678	0.00688
0.00688	0.00688	0.00156	0.00688
0.00688	0.00475	0.00081	0.00688

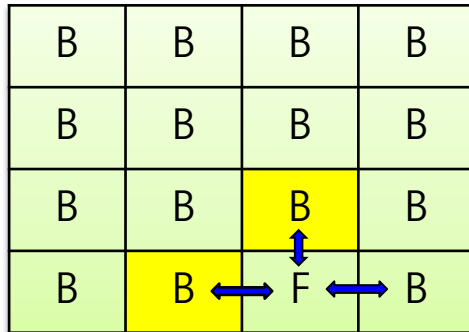
입력 영상의 per-pixel 확률



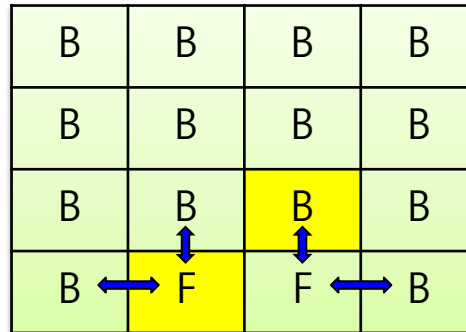
라벨링 이미지

5.6.2 Background Cut – Contrast term

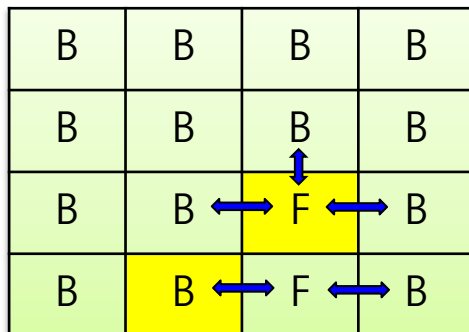
- U 영역에 대해 여러 경우로 labeling



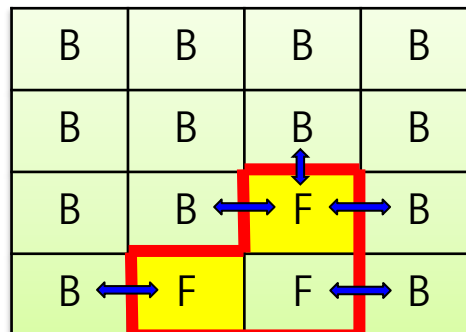
$E_2 \uparrow$, Energy \uparrow



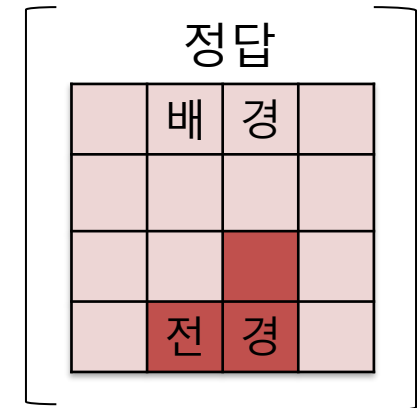
$E_2 \uparrow$, Energy \uparrow



$E_2 \uparrow$, Energy \uparrow



$E_2 \downarrow$, Energy \downarrow



참고

E_1 : Color term

E_2 : Contrast term

Energy : $E_1 + E_2$

5.6.2 Background Cut – Color term

- U 영역을 모두 배경으로 판단할 경우 (Worst case)

고정된 값

$$E_1 = \alpha \left(\sum_{n=1}^{13} GB_n + \sum_{n=1}^1 GF_n \right) + GB_\alpha + GB_\beta + (1 - \alpha) \sum_{n=1}^{16} PB_n$$

• 라벨링에 따라서 배경일 확률이나 전경일 확률을 얻음

- 회색 : 명확히 배경/전경 (불변)
- 노란색 : 불확실한 영역

참고

GB_n : Global background color model

GF_n : Global foreground color model

PB_n : Per-pixel background color model

} = $-\log(\text{확률})$

5.6.2 Background Cut – Color term

- U 영역을 모두 배경으로 판단할 경우 (Worst case)

0.00685	0.00688	0.00685	0.00688
0.00678	0.00678	0.00678	0.00688
0.00688	0.00688	0.00156	0.00688
0.00688	0.00475	0.00081	0.00688

per-pixel background 확률

0.00594	0.00579	0.00620	0.00597
0.00627	0.00658	0.00668	0.00718
0.00698	0.00578	0.00245	0.00508
0.00699	0.00386	0.00055	0.00518

global background 확률

0.00078	0.00099	0.00065	0.00088
0.00073	0.00078	0.00678	0.00068
0.00048	0.00068	0.00809	0.00046
0.00014	0.00765	0.00912	0.00066

global foreground 확률

$$E_1 = \alpha \left(\sum_{n=1}^{13} GB_n + \sum_{n=1}^1 GF_n \right) + GB_\alpha + GB_\beta + (1 - \alpha) \sum_{n=1}^{16} PB_n$$

α 값을 0.5라 가정 했을 때

$$E_1 = 1.02$$

참고

GB_n : Global background

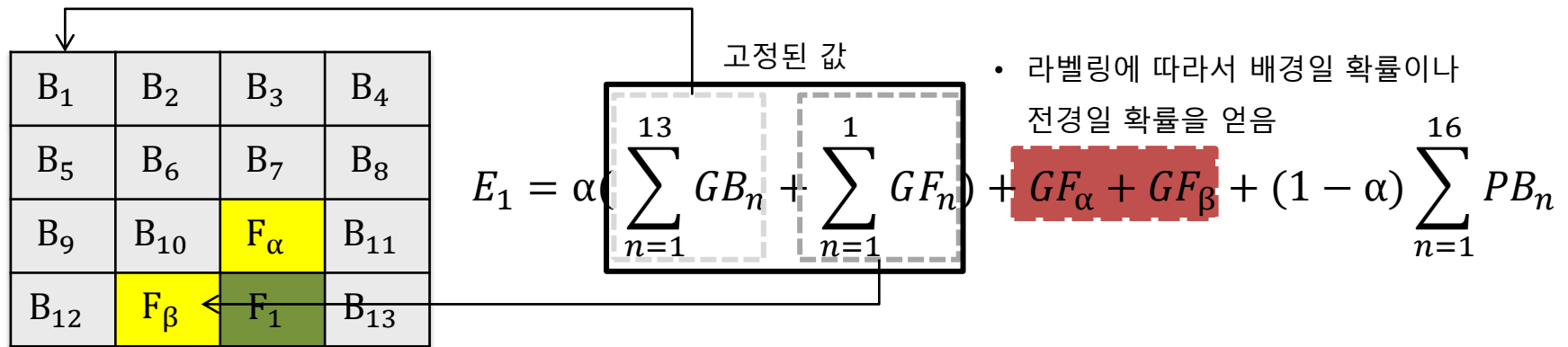
GF_n : Global foreground

PB_n : Per-pixel background

} = $-\log(\text{확률})$

5.6.2 Background Cut – Color term

- U 영역을 모두 전경으로 판단할 경우 (Best case)



- 회색 : 명확히 배경/전경 (불변)
- 노란색 : 불확실한 영역

참고

GB_n : Global background color model

GF_n : Global foreground color model

PB_n : Per-pixel background color model

} = $-\log(\text{확률})$

5.6.2 Background Cut – Color term

- U 영역을 모두 전경으로 판단할 경우 (Best case)

0.00685	0.00688	0.00685	0.00688	0.00594	0.00579	0.00620	0.00597	0.00078	0.00099	0.00065	0.00088
0.00678	0.00678	0.00678	0.00688	0.00627	0.00658	0.00668	0.00718	0.00073	0.00078	0.00678	0.00068
0.00688	0.00688	0.00156	0.00688	0.00698	0.00578	0.00245	0.00508	0.00048	0.00068	0.00809	0.00046
0.00688	0.00475	0.00081	0.00688	0.00699	0.00386	0.00055	0.00518	0.00014	0.00765	0.00912	0.00066
per-pixel background 확률				global background 확률				global foreground 확률			

$$E_1 = \alpha \left(\sum_{n=1}^{13} GB_n + \sum_{n=1}^1 GF_n \right) + GF_\alpha + GF_\beta + (1 - \alpha) \sum_{n=1}^{16} PB_n$$

α 값을 0.5라 가정했을 때

$$E_1 = 0.91$$

참고

GB_n : Global background
 GF_n : Global foreground
 PB_n : Per-pixel background

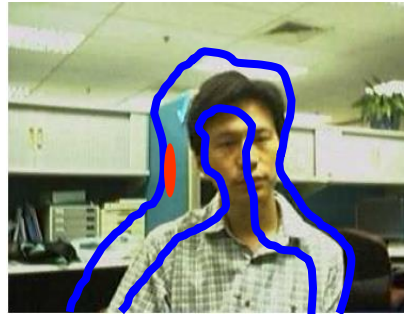
} = $-\log(\text{확률})$

5.6.2 Background Cut – Contrast term 문제점

- 문제점 1 : 배경에 강한 에지가 있는 경우



배경

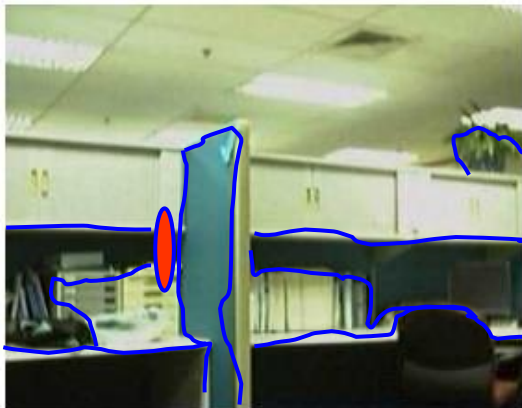


입력



결과

- 배경에 강한 에지가 있을 때, E_2 값 작아짐



배경에 강한 에지 존재

이웃 픽셀과 차이 값 \uparrow

배경과 전경 오분류

모델 변경 필요

5.6.2 Background Cut – Contrast term 문제점

■ 문제점 2 : 배경과 전경의 픽셀 값이 비슷한 경우



배경



입력

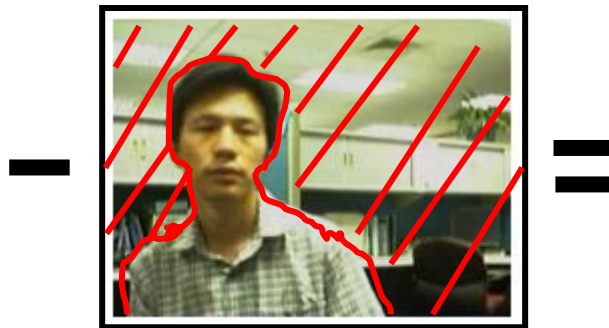


결과

■ 배경과 입력 영상의 픽셀 값 비교



배경



입력

=

배경과 입력의 차이 값 이용
모델 변경

If 차이 값 ↓

구분이 잘될 확률 ↓

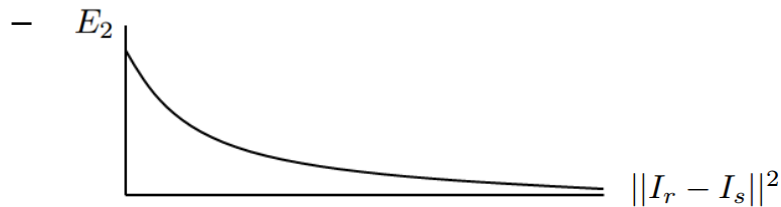
E_2 ↑

선택 확률 ↓

5.6.2 Background Cut – Contrast term

■ Basic model

- $E_2(x_r, x_s) = \underline{|x_r - x_s|} \cdot \exp(-\beta d_{rs})$
 - 이웃한 두 픽셀의 labeling이 다를 경우 값을 가짐
 - $d_{rs} = ||I_r - I_s||^2$



- 배경에 강한 에지, 배경과 전경 픽셀 값이 비슷한 경우
 - » 분할 오류 발생

5.6.2 Background Cut – Contrast term

■ 수정된 Contrast term

$$\blacksquare d''_{rs} = \|I_r - I_s\|^2 \cdot \frac{1}{1 + \left(\frac{\|I_r^B - I_s^B\|}{K} \right)^2 \exp\left(-\frac{z_{rs}^2}{\sigma_z}\right)}$$

추가된 부분

- $z_{rs} = \max\{\|I_r - I_r^B\|, \|I_s - I_s^B\|\}$

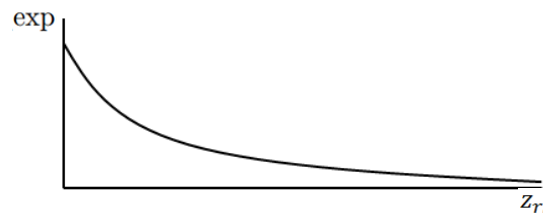
- 입력 영상과 배경 영상의 차이 값

- z_{rs} 값이 작을 경우

- » 배경에 속할 확률이 높음

- » $(\exp(-z_{rs}^2/\sigma_z) \rightarrow 1)$

- » contrast 감소 정도는 높아짐



5.6.2 Background Cut – Contrast term

■ 수정된 Contrast term

$$\blacksquare d''_{rs} = \|I_r - I_s\|^2 \cdot \frac{1}{1 + \left(\frac{\|I_r^B - I_s^B\|}{K} \right)^2 \exp\left(-\frac{z_{rs}^2}{\sigma_z}\right)}$$

$$\bullet z_{rs} = \max\{\|I_r - I_r^B\|, \|I_s - I_s^B\|\}$$

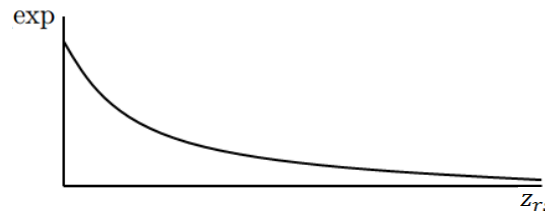
– 입력 영상과 배경 영상의 차이를 의미함

– z_{rs} 값이 클 경우

» 전경/배경 경계에 속할 확률이 높음

» $(\exp(-z_{rs}^2/\sigma_z) \rightarrow 0)$

» contrast 감소 정도는 낮아짐



5.6.2 참고자료

■ Graph Cut

- 컴퓨터 비전 책
- 유튜브 설명 영상
- <https://youtu.be/HMGX8HXskKk>

■ Grab Cut

- http://docs.opencv.org/3.1.0/d8/d83/tutorial_py_grabcut.html
- <http://bitsearch.blogspot.kr/2014/03/understanding-grabcut-interactive.html?view=timeslide>
- <http://www.dbpia.co.kr/Journal/PDFViewNew?id=NODE01634658&prevPathCode=>
- Segmentation QA.pdf

5.7 알고리즘 선택

■ 어떤 알고리즘을 선택해야 하나?

- 뚜렷한 가이드라인 없음
- 자신의 응용 환경에서 성능 실험하여 알아냄

■ 선택을 도우려는 노력: 데이터베이스

- UC 버클리, 바이츠만 연구소



(a) UC 버클리²²



(b) 바이츠만 연구소²³

5.7 알고리즘 선택

■ 선택을 도우려는 노력: 성능 분석

- 성능 측정 방법 [Unnikrishnan]
- 성능 벤치마킹 [Estrada2005, Arbelaez2011]

■ 표준 데이터베이스가 있어 좋지만, 주의할 점도 있다!

- 데이터베이스에 집착하여 과적합 문제를 야기할 수 있음
- 즉, 일반화 능력이 낮은 알고리즘을 낳을 수도 있기 때문에 주의해야 함
- 이러한 문제를 해결하는 한 방법으로 데이터베이스의 크기를 꾸준히 키워나가는 중