9장

클러스터링에 의한 세분화

중간 수준의 시각에서 결정적인 문제는 간결하면서도 표현력이 풍부한 이미지를 표현하는 것과 관련이 있습니다. 이러한 표현은 시각적 처리의 첫 번째 단계에서 사용할 수 있는 정보를 요약하고 전달해야 합니다. 초기 시력은 방대한 양의 정보를 생성하기 때문에 요약이 필요합니다. 사용 가능한 표현의 풍부함은 중요한 것을 압도하는 경향이 있습니다. 유용한 요약은 픽셀 또는 픽셀 그룹에서 계산할 수 있습니다. 또한 로컬 패턴 요소에서 계산할 수도 있습니다. 예를 들어 선이나 원에 있거나 복잡한 기하학적 구조에 가까운 것처럼 보이는 가장자리 점을 함께 수집합니다. 핵심 아이디어는 픽셀 또는 패턴 요소를 함께 수집하여 중요하거나 흥미롭거나 독특한 속성을 강조하는 요약 표현으로 만드는 것입니다.

이러한 표현을 얻는 것은 세분화, 그룹화, 지각 조직 또는 피팅으로 다양하게 알려져 있습니다. 기술은 다를 수 있지만 이러한 모든 활동에 대한 동기는 동일하기 때문에 광범위한 활동에 세분화라는 용어를 사용합니다. 무엇이 흥미롭고 무엇이 그렇지 않은지는 응용 프로그램에 따라 다르기 때문에 세분화에 대한 포괄적인 이론이 있을 수 있다고 보기는 어렵습니다.

이 글을 쓰는 시점에는 세분화에 대한 포괄적인 이론이 없으며 이 용어는 분기마다 다른 방식으로 사용됩니다.

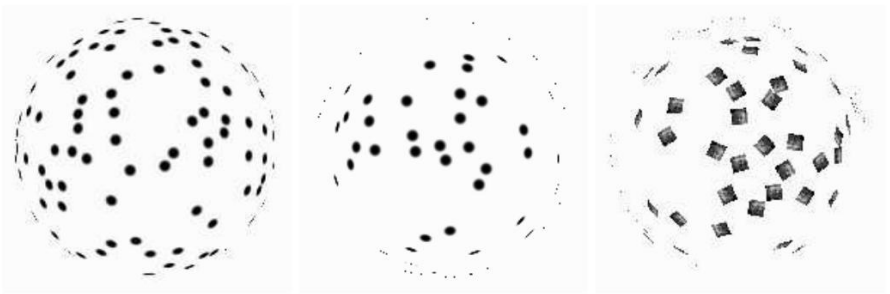


그림 9.1: 이 이미지에서 알 수 있듯이 시각의 중요한 구성 요소는 이미지 정보를 의미 있는 조합으로 구성하는 것과 관련이 있습니다. 인간의 시각 시스템은 오히려 잘 작동하는 것 같습니다. 이 세 이미지 각각에서 얼룩이 함께 구성되어 페이지 밖으로 튀어나온 것처럼 보이는 텍스처 표면을 형성합니다(반구처럼 느껴질 수 있음).

블롭은 "표면을 형성하기 때문에" 조립된 것으로 보이며, 거의 만족스러운 설명이 아니며 어려운 계산 문제를 야기합니다. 그들이 함께 같은 질감을 형성하기 때문에 조립되었다고 말하는 것도 질문을 하게 합니다(어떻게 알 수 있습니까?). 왼쪽 표면의 경우, 하나의 일관된 질감을 인식할 수 있는 프로그램을 작성하는 것이 상당히 어려울 수 있습니다. 이 구성 프로세스는 다양한 종류의 입력에 적용될 수 있습니다.

요약 표현이 무엇이어야 하는지에 대한 세부 사항은 작업에 따라 다르지만 매우 일반적으로 바람직한 기능이 많이 있습니다. 첫째, 일반적인 그림에 대해 계산된 표현에는 상대적으로 적은 수의 구성 요소가 있어야 합니다(즉, 최신 알고리즘이 처리할 수 있는 것보다 많지 않아야 함). 둘째, 이러한 구성 요소는 암시적이어야 합니다. 우리가 찾고 있는 물체가 있는지 여부는 이러한 구성 요소에서 꽤 분명해야 합니다.

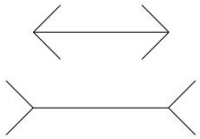


그림 9.2: 유명한 Mul-ler-Lyer 착시; 아래쪽 그림에 속하는 것이 더 길어 보이지만 수평선은 사실 길이가 같습니다. 분명히 이 효과는 각각의 분리된 세그먼트의 속성이 아니라 전체를 형성하는 관계의 일부 속성(형태성질)에서 발생합니다.

세분화에는 완전히 다르지 않은 두 가지 중요한 스레드가 있습니다. 첫 번째에서 요약은 항목 간의 로컬 관계에 초점을 맞춘 클러스터링 방법을 통해 순전히 로컬로 조립됩니다. 여기서 우리는 서로 닮은 아이템을 조립하려고 합니다. 예를 들어 이 접근 방식을 사용하면 유사하게 보이는 픽셀 덩어리를 함께 조립할 수 있습니다. 이러한 덩어리를 일반적으로 영역이라고 합니다.

일반적으로 이 접근법은 클러스터링 방법을 사용하며 이 장의 초점입니다.
두 번째 접근 방식에서는 전체 관계를 기반으로 항목을 함께 조립합니다(예: 직선에 있는 모든 항목). 그림 9.1은 작은 픽셀 그룹 모음을 보여줍니다. 이 그림을 볼 때 이러한 픽셀 그룹은 함께 속한 것처럼 보이며 함께 취하면 표면의 존재를 암시하기 때문일 가능성이 큽니다. 이 접근 방식에서 우리는 토큰 또는 픽셀 그룹의 픽셀을 함께 수집할 수 있는 방법에 관심이 있습니다. 이 접근법은 데이터 풀에서 파라메트릭 모델을 식별할 수 있는 방법을 강조합니다. 10장에서 그러한 방법을 설명합니다.

9.1 인간의 시각: 그룹화 및 게슈탈트

인간 시각 시스템의 주요 특징은 상황이 사물이 인식되는 방식에 영향을 미친다는 것입니다(예: 그림 9.2의 착시 참조). 이러한 관찰로 인해 게슈탈트 심리학파는 자극에 대한 반응 연구를 거부하고 그룹화를 시각적 인식을 이해하는 열쇠로 강조했습니다. 그들에게 그룹화는 그림의 일부 구성 요소를 함께 조합하고 함께 인식하는 시각 시스템의 경향을 의미했습니다(이는 위에서 사용된 컨텍스트라는 단어에 다소 대략적인 의미를 제공합니다). 예를 들어 그룹화는 그림 9.2의 M¨uller-Lyer 착시를 일으키는 것입니다. 비전 시스템은 두 개의 화살표 구성 요소를 조립하고 수평선은 전체 구성 요소가 아니라 전체 구성 요소로 인식되기 때문에 서로 다르게 보입니다. 라인보다. 또한 많은 그룹화 효과는 인지 입력에 의해 중단될 수 없습니다. 예를 들어 화살표를 그룹화하지 않기로 결정하면 그림 9.2의 선 길이가 동일하게 보이도록 만들 수 없습니다.

분할의 일반적인 경험은 이미지가 그림(일반적으로 중요하고 중요한 대상)과 근거로 분해될 수 있는 방식입니다.

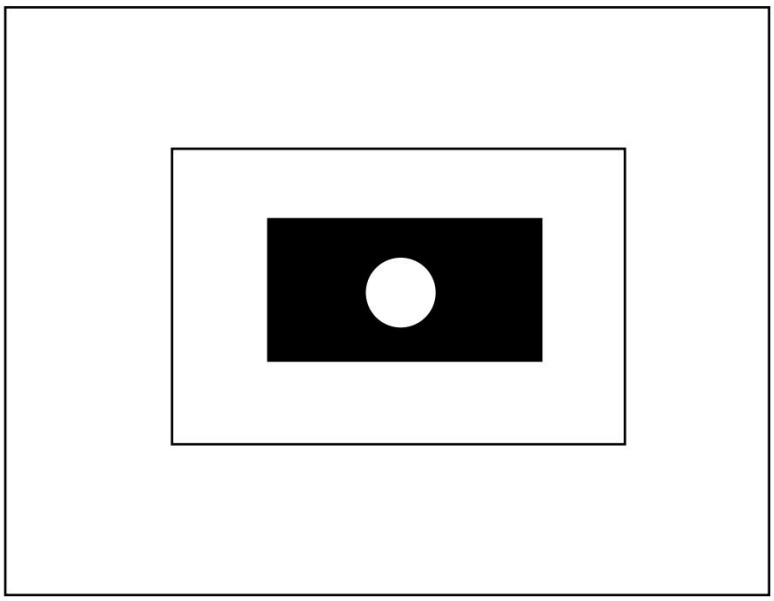


그림 9.3: 분할에 대한 한 가지 관점은 이미지의 어떤 구성 요소가 그림을 형성하고 어떤 요소가 배경을 형성하는지 결정한다는 것입니다. 이 그림은 이러한 관점에서 발생하는 모호성의 한 형태를 보여줍니다. 흰색 원은 검은색 직사각형 바탕에 도형으로 보이거나 원형 구멍이 있는 검은색 직사각형 모양이고 바닥이 흰색 정사각형인 바탕으로 볼 수 있습니다.

인물이 놓인 배경. 그러나 그림 9.3에서 알 수 있듯이 그림과 근거는 매우 모호할 수 있으며 이는 더 풍부한 이론이 필요함을 의미합니다.

게슈탈트 학파는 게슈탈트(전체 또는 그룹) 개념과 전체를 만드는 내부 관계 집합(예: 그림 9.2)을 아이디어의 중심 구성 요소로 사용했습니다. 그들의 작업은 이미지 요소가 함께 연관되고 그룹으로 해석되는 일련의 규칙을 기록하려는 시도로 특징지어집니다. 순전히 역사적인 관심사인 알고리즘을 구성하려는 시도도 있었습니다(그들의 작업을 넓은 맥락에서 소개하는 설명은 Gordon(1997) 참조).

게슈탈트 심리학자들은 일련의 요소를 식별했으며, 이들은 그룹화할 일련의 요소를 미리 배치한다고 느꼈습니다. 이러한 요소는 인간의 시각 시스템이 어떤 방식으로든 사용한다는 것이 매우 분명하기 때문에 중요합니다. 또한 유용한 중간 표현으로 이어지는 토큰이 함께 속할 때 선호도 집합을 나타낼 것으로 기대하는 것이 합리적입니다.

다양한 요소가 있으며, 그 중 일부는 주요 게슈탈트 운동 이후의 것입니다.

- 근접성: 근처에 있는 토큰은 그룹화되는 경향이 있습니다.
- 유사성: 유사한 토큰은 함께 그룹화되는 경향이 있습니다.
- 일반적인 운명: 일관된 움직임을 가진 토큰은 다음과 같이 그룹화되는 경향이 있습니다.

그녀를 갖다.

- 공통 영역: 동일한 폐쇄 영역 내에 있는 토큰은 함께 그룹화되는 경향이 있습니다.
- 평행성: 평행 곡선 또는 토큰은 함께 그룹화되는 경향이 있습니다.
- 폐쇄: 폐쇄 곡선으로 이어지는 경향이 있는 토큰 또는 곡선은 함께 그룹화되는 경향이 있습니다.
- 대칭: 대칭 그룹으로 연결되는 곡선은 함께 그룹화됩니다.
- 연속성: 지속으로 이끄는 토큰
  형식적인 의미보다 곡선이 그룹화되는 경향이 있습니다.
- 친숙한 구성: 그룹화될 때 친숙한 개체로 이어지는 토큰은 함께 그룹화되는 경향이 있습니다.

이러한 법칙은 그림 9.4, 9.5, 9.7 및 9.1에 설명되어 있습니다.

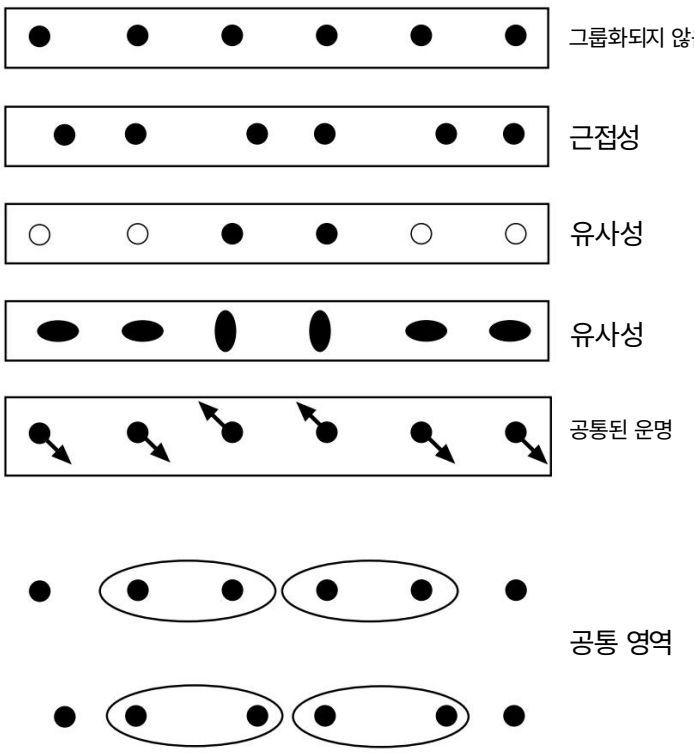


그림 9.4: 그룹화로 이어지는 게슈탈트 요인의 예(본문에 자세히 설명되어 있음).

이러한 규칙은 설명으로서 상당히 기능할 수 있지만 알고리즘을 구성하는 것으로 간주하기에는 불충분합니다. 게슈탈트 심리학자들은 심각한

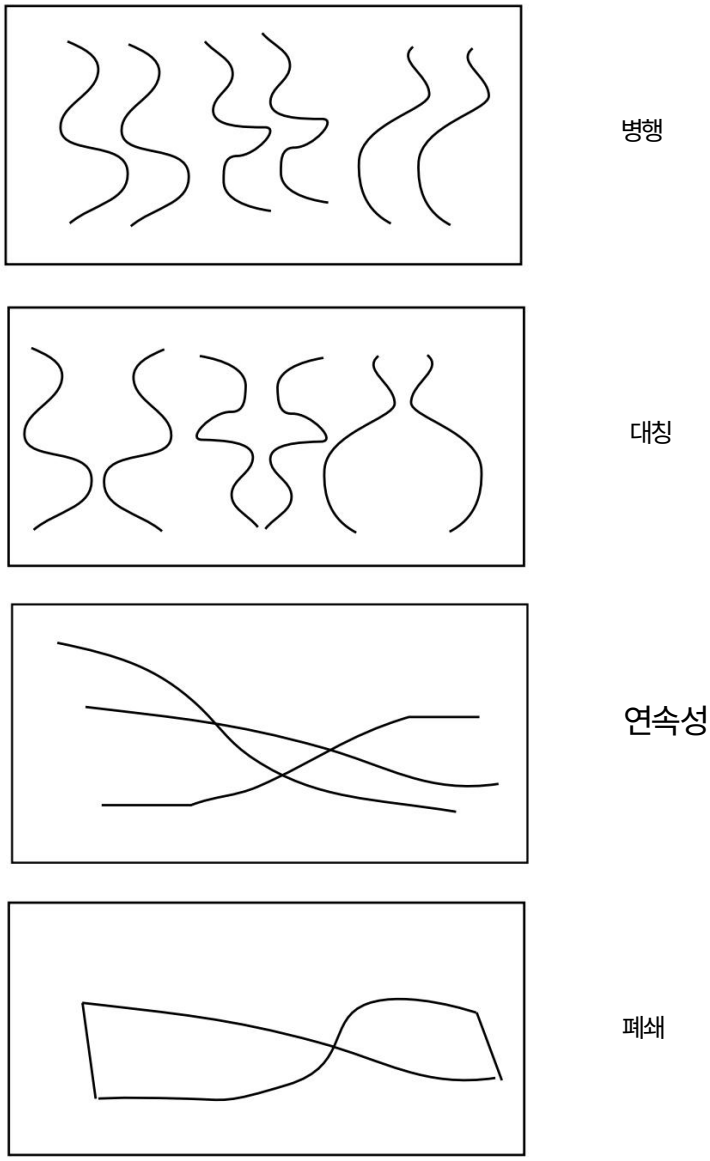


그림 9.5: 그룹화로 이어지는 게슈탈트 요인의 예(본문에 자세히 설명되어 있음).

하나의 규칙이 적용된 때와 다른 때와 같은 세부 사항에 어려움이 있습니다. 이러한 규칙을 사용하기 위한 만족스러운 알고리즘을 제공하는 것은 어렵습니다. 게슈탈트 운동은 극단 원리를 사용하려고 시도했습니다.

익숙한 구성이 특별한 문제입니다. 핵심 문제는 어떤 익숙한 구성이 문제에 적용되고 어떻게 선택되는지 이해하는 것입니다. 예를 들어, 그림 9.1을 보십시오. 블롭이 그룹화되는 이유는 다음과 같습니다.

그들은 구체를 산출합니다. 이 견해의 어려움은 이것이 어떻게 발생했는지 설명하는 것입니다. 구체가 존재한다는 가설은 어디에서 왔습니까? 모든 객체의 모든 보기를 통한 검색은 하나의 설명이지만 이 검색이 어떻게 구성되는지 설명해야 합니다. 우리는 모든 점의 패턴으로 모든 구체의 모든 보기를 확인합니까?

이것이 어떻게 효율적으로 이루어질 수 있습니까?

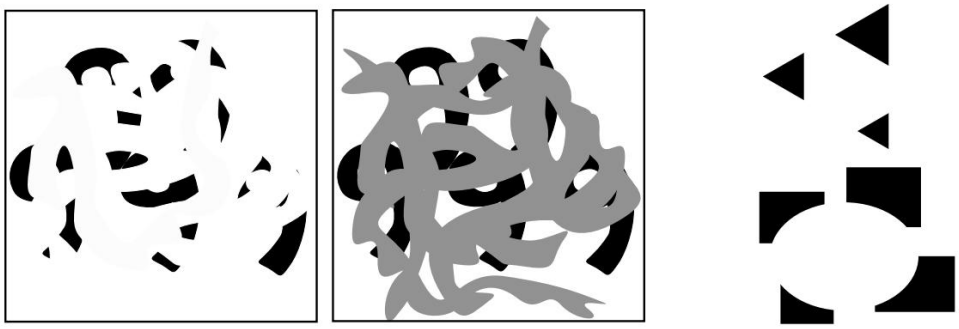


그림 9.6: 폐색은 그룹화에서 중요한 단서로 보입니다. 왼쪽의 패턴을 숫자 모음으로 볼 수 있습니다. 그 옆에 있는 패턴은 가려진 숫자입니다. 각 그림의 검은색 영역은 동일합니다. 두 그림의 중요한 차이점은 중첩된 회색 영역이 검은색 영역이 단순히 흩어진 검은색 영역이 아니라 이유 때문에 분리된 더 큰 물체의 구성 요소라는 증거를 제공한다는 것입니다. 오른쪽에는 경계가 이미지의 많은 부분과 대조되지 않는 가려진 물체의 존재를 암시하는 토큰으로 구성된 두 개의 그림이 있습니다. 폐색된 형상의 전체 윤곽의 위치에 대한 명확한 인상을 가지고 있음에 주목하십시오. 이러한 윤곽선은 착시 윤곽선으로 알려져 있습니다.

게슈탈트 규칙은 다양한 예에서 일어나는 일을 설명하기 때문에 어느 정도 통찰력을 제공합니다. 이러한 설명은 규칙이 현실 세계에서 일반적으로 발생하는 시각적 효과로 인해 제기되는 문제를 해결하는 데 도움이 된다고 제안하기 때문에 합리적인 것 같습니다. 즉, 생태학적으로 타당합니다. 예를 들어 연속성은 폐색으로 인한 문제에 대한 해결책을 나타낼 수 있습니다. 가려진 객체의 윤곽 섹션은 연속성에 의해 결합될 수 있습니다(그림 9.6 참조).

폐색으로 설명되는 해석을 선호하는 이러한 경향은 흥미로운 효과로 이어집니다. 하나는 그림 9.6에 나와 있는 착시 윤곽선입니다. 여기서 일련의 토큰은 개체의 존재를 암시하며 대부분의 윤곽에는 대비가 없습니다. 토큰은 폐색 개체의 존재에 대한 신호를 제공하기 때문에 함께 그룹화되는 것처럼 보입니다. 이는 이러한 토큰에 의해 매우 강력하게 제안되어 윤곽의 비대비 영역을 채울 수 있습니다.

이 생태학적 주장은 그것을 사용하여 대부분의 그룹화 요인을 해석할 수 있기 때문에 어느 정도 설득력이 있습니다. 공통된 운명은 물체의 구성 요소가 함께 움직이는 경향이 있다는 사실의 결과로 볼 수 있습니다. 마찬가지로 대칭 또는 대칭 윤곽에 가까운 실제 개체가 많기 때문에 대칭은 유용한 그룹화 신호입니다. 본질적으로 생태학적 주장에 따르면 토큰은 사람들이 접하는 시각적 세계에 도움이 되는 표현을 생성하기 때문에 그룹화됩니다. 생태학적 논증은 애매모호하지만 통계적 특징이 있기는 하지만 호소력이 있다. 우리의 관점에서 게슈탈트 요인은 흥미로운 정보를 제공합니다.

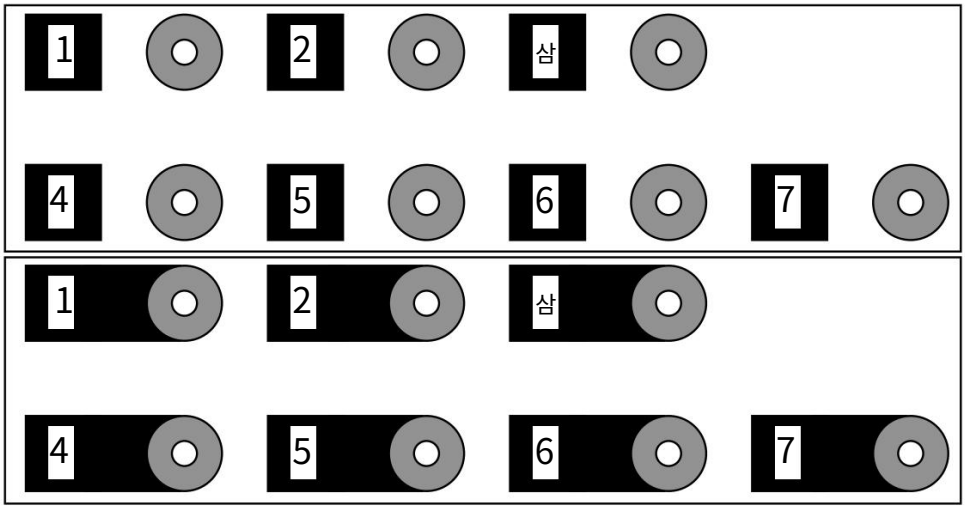


그림 9.7: 실생활에서 현상을 그룹화하는 예. UC Berkeley의 컴퓨터 과학 건물에 있는 엘리베이터의 버튼은 상단 그림과 같이 배치되어 있었습니다. 잘못된 층에 도착해서 잘못된 버튼을 눌렀기 때문이라는 것을 발견하는 것이 일반적이었습니다. 단추를 올바른 레이블로 명확하게 그룹화하기 어렵고 한 눈에 잘못된 그룹화를 쉽게 얻을 수 있습니다. 공신력 있는 개인이 하단 그림과 같이 숫자와 버튼 사이의 간격을 메웠고 근접 신호가 명확해지면서 혼란이 멈췄습니다.

그러나 프로세스 자체가 아니라 더 큰 그룹화 프로세스의 결과로 보아야 합니다.

9.2 중요한 애플리케이션

단순 분할 알고리즘은 종종 중요한 애플리케이션에 유용합니다. 일반적으로 간단한 알고리즘은 유용한 분해가 무엇인지 쉽게 알 수 있을 때 가장 잘 작동합니다. 두 가지 중요한 사례는 배경 빼기(알려진 배경처럼 보이지 않는 모든 것이 흥미로운 경우)와 샷 경계 감지(비디오의 상당한 변화가 흥미로운 경우)입니다.

다른 두 가지 매우 중요한 응용 프로그램에는 더 복잡한 알고리즘이 필요합니다. 대화형 분할에서 사용자는 분할 시스템을 안내하여 그림에서 개체를 잘라냅니다. 마지막으로 주요 목표는 이미지 영역을 형성하는 것입니다.

9.2.1 배경 빼기

많은 응용 프로그램에서 개체는 대체로 안정적인 배경에 나타납니다. 일반적인 예는 컨베이어 벨트에서 부품을 감지하는 것입니다. 또 다른 예는 도로의 오버헤드 뷰에서 자동차를 세는 것입니다. 도로는 외관상 꽤 안정적입니다. 덜 분명한 또 다른 예는 인간과 컴퓨터의 상호 작용입니다. 일반적으로 카메라는 고정되어(예: 모니터 상단에) 방을 봅니다. 방처럼 보이지 않는 보기의 거의 모든 것이 흥미롭습니다.

이러한 종류의 응용 프로그램에서 유용한 세분화는 종종 다음과 같이 얻을 수 있습니다.

그림 9.8: 이 그림은 무늬가 있는 소파에서 노는 어린이의 120개 프레임 시퀀스에서 5번째 프레임마다 표시됩니다. 프레임은 그림 9.10에서 설명하는 이유로 80 x 60 해상도에서 사용됩니다. 시퀀스 중에 자식이 프레임의 한쪽에서 다른 쪽으로 이동하는 것을 확인하십시오.

그림 9.9: 80 x 60 프레임을 사용하는 그림 9.8의 시퀀스에 대한 배경 빼기 결과. 배경을 계산하는 두 가지 방법을 비교합니다. (a) 모든 120 프레임의 평균. 아이가 소파의 한쪽에서 다른 쪽보다 더 많은 시간을 보냈기 때문에 평균적으로 희미한 흐림이 발생했습니다. (b) 평균과의 차이가 작은 임계값을 초과하는 픽셀. (c) 평균과의 차이가 다소 큰 임계값을 초과하는 것. 각 경우에 약간의 초과 픽셀과 일부 누락 픽셀이 있습니다.

이미지에서 배경 모양의 추정치를 빼고 결과에서 큰 절대값을 찾습니다. 주요 문제는 배경에 대한 좋은 평가를 얻는 것입니다. 한 가지 방법은 단순히 사진을 찍는 것입니다. 이 접근 방식은 배경이 일반적으로 시간이 지남에 따라 천천히 변하기 때문에 제대로 작동하지 않습니다. 예를 들어, 도로는 비가 오면 더 빛나고 날씨가 건조하면 덜 빛날 수 있습니다. 사람들은 방 안에서 책과 가구를 옮길 수 있습니다. 등등.

일반적으로 잘 작동하는 대안은 이동 평균을 사용하여 배경 픽셀의 값을 추정하는 것입니다. 이 접근법에서 우리는

그림 9.10: 정합은 특히 텍스처의 경우 배경 빼기에서 상당한 골칫거리가 될 수 있습니다. 이 그림은 160 x 120 프레임을 사용하여 그림 9.8의 시퀀스에 대한 결과를 보여줍니다. 배경을 계산하는 두 가지 방법을 비교합니다. (a) 전체 120개 프레임의 평균. 아이가 소파의 한 쪽에서 다른 쪽보다 더 많은 시간을 보냈고 평균적으로 희미한 흐릿함을 나타냈습니다. (b) 평균과의 차이가 작은 임계값을 초과하는 픽셀. (c) 평균과의 차이가 다소 큰 임계값을 초과하는 것. 소파의 패턴이 어린이로 오인된 문제 픽셀의 수가 눈에 띄게 증가했음을 알 수 있습니다. 작은 움직임으로 인해 소파의 높은 공간 주파수 패턴이 어긋나 큰 차이가 발생할 수 있기 때문입니다.

배경 견적 작성 B (0) . 각 프레임에서 F

일반적으로 wiB (n i) waF+ 로 배경 추정치를 업데이트합니다.

가중치 wa, wi = ——————

확장실

및 wc 의 선택을 위해 B (n+1)를 형성합니다.

에서 배경 추정치를 뺍니다.

프레임을 표시하고 차이의 크기가 일부 임계값보다 큰 각 픽셀의 값을 보고합니다.

끝

알고리즘 9.1: 배경 빼기.

이전 값의 가중 평균으로 특정 배경 픽셀. 일반적으로 먼 과거의 픽셀은 가중치가 0이어야 하며 가중치는 부드럽게 증가합니다. 이상적으로 이동 평균은 배경의 변화를 추적해야 합니다. 즉, 날씨가 빠르게 변하는 경우(또는 책을 옮기는 사람이 열광적일 경우) 상대적으로 적은 수의 픽셀이 0이 아닌 가중치를 가져야 하며 변화가 느린 경우 이전 픽셀의 수는 0이 아닌 가중치가 증가해야 합니다. 이것은 알고리즘 9.1을 생성합니다. 필터 장을 읽은 사람들에게 이것은 시간의 함수를 평활화하는 필터이며 배경에서 일반적인 변경 주파수보다 큰 주파수를 억제하고 해당 주파수 이하의 주파수는 통과시키기를 바랍니다. . 이 접근 방식은 매우 성공적일 수 있지만 그림 9.9 및 9.10에 나와 있는 것처럼 상당히 거친 규모의 이미지에 사용해야 합니다.

이미지 시퀀스의 각 프레임에 대해 이 프레임과 이전
프레임 사이의 거리를 계산합니다. 거리가 임계값보다 크면 프레임
을 샷 경계로 분류합
니다.

끝

알고리즘 9.2: 프레임간 차이를 사용한 샷 경계 검출.

9.2.2 샷 경계 감지

비디오의 긴 시퀀스는 샷으로 구성됩니다. 대체로 동일한 대상을 보여주는 훨씬 짧은 하위 시퀀스입니다. 이러한 샷은 일반적으로 편집 프로세스의 산물입니다. 샷 사이의 경계가 어디인지에 대한 기록은 거의 없습니다.

비디오를 샷 모음으로 나타내는 것이 유용합니다. 그런 다음 각 샷을 키 프레임으로 나타낼 수 있습니다. 이 표현은 비디오를 검색하거나 사용자가 비디오 또는 비디오 세트를 탐색할 수 있도록 콘텐츠를 캡슐화하는 데 사용할 수 있습니다.

이러한 샷의 경계를 자동으로 찾는 것(샷 경계 감지)은 간단한 분할 알고리즘의 중요한 실제 응용 프로그램입니다. 샷 경계 감지 알고리즘은 비디오에서 이전 프레임과 크게 다른 프레임을 찾아야 합니다. 우리의 중요성 테스트는 주어진 샷 내에서 물체와 배경이 모두 시야에서 움직일 수 있다는 사실을 고려해야 합니다. 일반적으로 이 테스트는 거리의 형태를 취합니다. 거리가 임계값보다 크면 샷 경계가 선언됩니다(알고리즘 9.2).

거리를 계산하기 위한 다양한 표준 기술이 있습니다.

- 프레임 차이 알고리즘은 시퀀스의 각 두 프레임 사이의 픽셀별 차이를 취하여 차이의 제곱을 합산합니다. 이러한 알고리즘은 속도가 느리고(차이가 많음) 카메라가 흔들릴 때 많은 샷을 찾는 경향이 있기 때문에 인기가 없습니다.

- 히스토그램 기반 알고리즘은 각 프레임에 대한 색상 히스토그램을 계산하고 히스토그램 사이의 거리를 계산합니다. 색상 히스토그램의 차이는 프레임에서 색상의 공간적 배열에 민감하지 않기 때문에 사용하기에 합리적인 측정입니다(예: 작은 카메라 흔들림은 히스토그램에 영향을 미치지 않음).

- 블록 비교 알고리즘은 프레임을 상자 그리드로 자르고 상자를 비교하여 프레임을 비교합니다. 이것은 왼쪽 하단 모서리에서 화면 밖으로 사라지는 빨간색 개체가 상단 가장자리에서 화면에 나타나는 찢어진 개체와 동일한 색상 토그램의 어려움을 피하기 위한 것입니다. 일반적으로, 이러한 블록 비교 알고리즘은 프레임간 거리에 대해 사용되는 것과 같은 방법을 사용하여 각각 계산되는 블록간 거리의 합성(최대값을 취하는 것이 자연스러운 전략임)인 프레임간 거리를 계산합니다.

- 에지 차이 알고리즘은 각 프레임에 대한 에지 맵을 계산한 다음 이러한 에지 맵을 비교합니다. 일반적으로 비교는 잠재적으로 대응하는 에지(근처, 유사한 방향,

그림 9.11: 이미지에서 개체를 잘라내려는 사용자(왼쪽)는 일부 전경 픽셀과 일부 배경 픽셀(가운데)을 표시한 다음 대화형 분할 방법을 사용하여 오른쪽에서 잘라낸 구성 요소를 가져올 수 있습니다. 이 방법은 표시된 픽셀에서 전경 및 배경 픽셀 모양의 모델을 생성한 다음 이 정보를 사용하여 그림 바닥 분할을 결정합니다. 이 그림은 원래 Protiere와 Sapiro가 "Interactive Image Segmentation via Adaptive Weighted Distances"의 그림 9, IEEE Transactions on Image Processing, 2007 c IEEE, 2007로 게시되었습니다.

등) 다음 프레임에서. 잠재적으로 대응하는 에지가 거의 없는 경우 샷 경계가 있습니다. 해당 모서리의 수를 변환하여 거리를 얻을 수 있습니다.

이러한 방법은 상대적으로 임시 방편이지만 종종 당면한 문제를 해결하기에 충분합니다.

9.2.3 대화형 세분화

사람들은 종종 이미지에서 개체를 잘라내어 다른 이미지로 옮기고 싶어합니다. 이렇게 하는 데는 여러 가지 이유가 있습니다. 섹션 6.3에서 소스 이미지의 결과 구멍을 채우는 방법을 설명하는 일부 스케치를 했습니다. 그러나 이 작업을 효율적으로 수행하려면 잘라내려는 개체를 선택하는 좋은 방법이 필요합니다. 객체의 픽셀이나 이미지의 경계를 칠하는 것은 너무 많은 작업입니다.

이것은 분명히 분할 문제이지만 전경과 배경이라는 두 개의 세그먼트가 있는 특수한 문제입니다. 전경 세그먼트는 일관성이 있어야 하지만 배경 세그먼트는 그렇지 않을 수 있습니다. 문제를 공격하는 다양한 방법은 다양한 유형의 인터페이스를 중심으로 구축됩니다. 지능형 가위 인터페이스에서 사용자는 개체의 경계에 상당히 가까운 곡선을 스케치합니다. 그런 다음 이 곡선은 로컬 정보(일반적으로 이미지 그래디언트 큐)를 사용하여 경계로 이동합니다. 페인팅 인터페이스에서 사용자는 전경 또는 배경 브러시로 일부 픽셀을 페인팅합니다. 이 픽셀은 전경과 배경의 모양 모델을 생성하는 데 사용됩니다. 차례로, 이러한 모델은 빠른

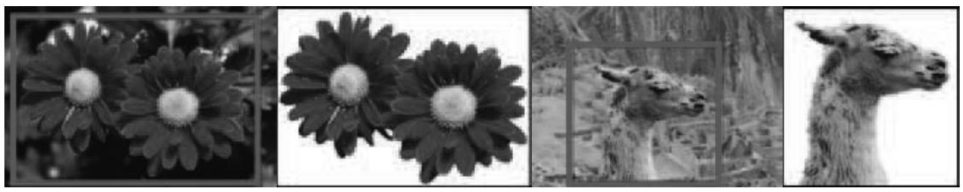


그림 9.12: 대화형 분할을 위한 Grabcut 인터페이스에서 사용자는 관심 대상 주위에 상자를 표시합니다. 그런 다음 클러스터링 방법으로 전경 및 배경 모델을 유추하고 객체를 분할합니다. 이 분할이 만족스럽지 않은 경우 사용자는 모델을 안내하는 데 도움이 되도록 픽셀에 전경 및 배경 획을 페인팅할 수 있습니다. 이 그림은 원래 C. Rother, V. Kolmogorov 및 A. 블레이크, ACM 트랜스. on Graphics(ACM SIGGRAPH Proc), Vol. 23:3 c 2004, ACM, Inc. http: // doi. 약. org/ 10. 1145/ 1186562. 1015720 허가를 받아 재인쇄됨.

그래프 기반 분할기(섹션 9.4.3). 그림 9.11은 프로세스를 보여줍니다. 마지막으로 Grabcut 인터페이스에서 사용자는 개체 주위에 상자를 그립니다. 이 상자는 전경 및 배경 픽셀의 초기 추정치를 생성하고 여기에서 초기 분할을 얻습니다. 이는 향상된 분할을 생성하는 전경 및 배경 모델을 생성합니다(그림 9.12).

종종 픽셀은 순수한 배경이나 순수한 전경이 아닙니다. 예를 들어, 얼굴 사진에서 머리카락 경계 주변의 픽셀은 다소 모호합니다. 여기에는 머리카락만 포함하거나 배경만 포함하는 픽셀이 거의 없습니다. 대신 픽셀은 렌즈를 통해 들어오는 빛을 평균화하기 때문에 대부분 머리카락과 배경의 가중 평균 값을 갖습니다. 이 경우 대화형 분할을 사용하여 [0 1] 범위의 값 마스크인 매트를 준비할 수 있습니다. 매트는 전통적으로 α로 작성되며 i번째 픽셀 값의 모델은 αf + (1 α)b입니다. 여기서 f와 b는 전경 및 배경 값입니다(그림 9.13). 로토스코핑은 매트와 같은 프로세스이지만 비디오에 적용됩니다. 여기에서 움직이는 객체에 해당하는 프레임당 하나씩 세그먼트 세트를 복구합니다. 그런 다음 이러한 세그먼트를 새 배경에 합성하여 개체가 새 배경에서 움직이는 것처럼 보이게 할 수 있습니다. 매팅 및 로토스코핑 방법은 분할 방법과 밀접한 관련이 있지만 표현 방식이 약간 다릅니다. 메모에 몇 가지 지침을 제공합니다.

9.2.4 이미지 영역 형성

분할의 한 가지 응용 프로그램은 대략적으로 일관된 색상과 질감을 가진 영역으로 이미지를 분해하는 것입니다. 일반적으로 이러한 영역의 모양은 특별히 중요하지 않지만 일관성은 중요합니다. 이 프로세스는 상당히 광범위하게 연구되었으며(종종 세분화라는 용어의 배타적 의미라고 함) 일반적으로 인식의 첫 번째 단계로 간주됩니다. 영역은 여러 애플리케이션에서 중요한 이미지 표현입니다. 영역은 이미지를 압축하는 경로를 제공할 수 있습니다. 각 영역은 일관된 모양을 갖기 때문에 영역의 모양과 모양을 개별적으로 설명하여 이미지를 압축할 수 있습니다(각 픽셀을 독립적으로 설명하는 것과 반대로).

영역은 다른 많은 시각적 계산의 중추로 사용될 수 있습니다. 을 위한

그림 9.13: 매팅 방법은 일부 픽셀이 전경 및 배경 값의 평균으로 구성되는 경우 가려진 경계 등에서 머리카락의 효과를 보상하기 위해 전경 배경 마스크가 아닌 실제 값 마스크를 생성합니다. 매트는 전경 픽셀의 경우 밝고 배경 픽셀의 경우 어둡습니다. 머리카락의 일부 픽셀은 회색입니다. 즉, 전경이 새 이미지로 전송될 때 이러한 픽셀은 전경과 배경의 가중 합계가 되어야 합니다. 회색 값은 가중치를 나타냅니다. 이 그림은 원래 A. Levin, A. Rav-Acha 및 D. Lischinski, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2008 c IEEE, 2008에 의해 "Spectral Matting"의 그림 6으로 게시되었습니다.

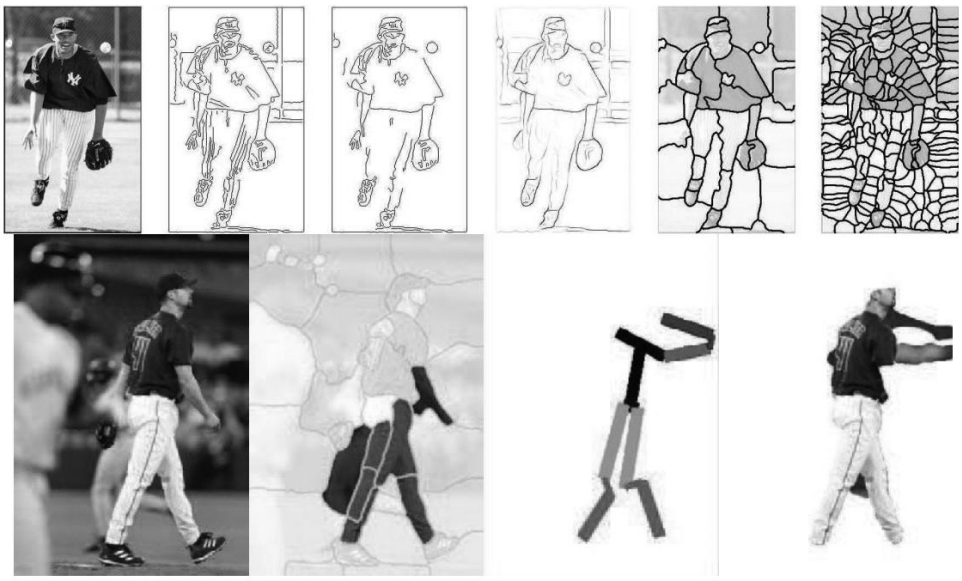


그림 9.14: 슈퍼픽셀은 종종 다른 표현이 숨기는 이미지의 구조를 노출할 수 있습니다. 인체 부분은 길고 얇은 부분으로 나타나는 경향이 있습니다. 맨 위 행에는 3개의 서로 다른 에지 맵(2개의 스무딩 스케일이 있는 5.2.1절의 에지 검출기 및 17.1.3절의 Pb)과 2개의 "스케일"(이 경우에는)에서 계산된 슈퍼 픽셀이 있는 이미지가 있습니다. , 슈퍼픽셀의 수가 제한됨). 더 거친 슈퍼픽셀은 사지 세그먼트를 직접적인 방식으로 노출시키는 경향이 있습니다. 맨 아래 행에는 다른 이미지, 해당 슈퍼픽셀 및 슈퍼픽셀 표현에서 유추된 두 가지 버전의 본문 레이아웃이 있습니다. 이 그림은 원래 G. Mori, X. Ren, A. Efros 및 J. Malik, Proc.에 의해 "Recovering human body configurations: Combining Segmentation and Recognition"의 그림 3 및 그림 10의 일부로 게시되었습니다. IEEE CVPR, 2004 c IEEE, 2004.

예를 들어, 광학 흐름을 계산하거나 이미지의 일부를 등록하기 위해 두 이미지 사이의 대응 관계를 식별하려는 경우 영역 간의 대응 관계에서 시작할 수 있습니다. 두 번째 예로, 존재하는 객체의 이름으로 이미지에 레이블을 지정하려는 경우 영역을 사용하여 특정 레이블에 해당하는 이미지 픽셀을 알 수 있기 때문에 레이블이 지정된 항목을 추적하는 데 도움이 됩니다. 또 다른 예로서, 영역을 이미지 내의 다른 영역과 일치시켜 예를 들어 건물 정면의 창문에서 보는 반복의 종류를 찾을 수 있습니다. 텍스처는 아니지만 여전히 반복적입니다.

일부 응용 프로그램에서는 영역이 상당히 커야 하며 모양이 복잡할 수 있습니다. 우리는 영역이 객체 경계를 크게 존중하기를 원할 수 있습니다(예: 이미지에서 객체에 레이블을 지정하는 경우). 대부분의 클러스터링 방법은 이와 같은 영역을 구성하는 분할기를 생성하도록 조정할 수 있습니다.

다른 응용 프로그램에서는 작고 조밀한 영역을 갖는 것이 더 유용합니다. 이를 일반적으로 슈퍼픽셀이라고 합니다. 슈퍼픽셀은 픽셀 그리드에 비해 작지만 여전히 매우 풍부한 표현이 필요할 때 특히 유용합니다(이러한 표현을 때때로 과잉 세분화라고 함). 응용 프로그램의 한 예는 컴퓨팅 밝기입니다(섹션 2.2.3). 음영 필드를 표현하려는 경우 픽셀 그리드에 표현하는 것은 느리게 변경되기 때문에 낭비입니다. 대신 슈퍼픽셀당 하나의 음영 값을 가질 수 있으며 결과를 부드럽게 할 수 있습니다.

다른 응용 프로그램이 인식 중입니다. 예를 들어, 인간의 팔과 다리는 길고 곧은 경향이 있습니다. 암시적인 그룹을 형성하기 위해 슈퍼픽셀을 조립하여 찾을 수 있습니다. 이것은 큰 영역을 자르는 것보다 더 쉬운 것 같습니다(그림 9.14).

9.3 클러스터링 픽셀에 의한 이미지 분할

클러스터링은 데이터 세트가 함께 속한 데이터 포인트의 모음인 클러스터로 대체되는 프로세스입니다. 이미지 분할을 클러스터링으로 생각하는 것은 당연합니다. 함께 속한 픽셀 클러스터로 이미지를 표현하고 싶습니다. 사용되는 특정 기준은 애플리케이션에 따라 다릅니다.

픽셀은 색상이 동일하고 텍스처가 동일하며 근처에 있기 때문에 함께 속할 수 있습니다.

클러스터링에 의한 이미지 분할의 일반적인 레시피는 다음과 같습니다. 특징 벡터로 각 이미지 픽셀을 나타냅니다. 이 특징 벡터에는 픽셀을 설명하는 데 관련될 수 있는 모든 측정값이 포함됩니다. 자연적 특징 벡터에는 다음이 포함됩니다. 픽셀의 강도; 픽셀의 강도와 위치; 적절한 색 공간으로 표현되는 픽셀의 색; 픽셀의 색상과 위치 픽셀의 색상, 위치, 로컬 텍스처 표현에서 필터 출력의 벡터(섹션 6.1과 비교). 이러한 특징 벡터를 클러스터링합니다. 모든 특징 벡터는 정확히 하나의 클러스터에 속하므로 각 클러스터는 이미지 세그먼트를 나타냅니다. 각 픽셀의 특징 벡터를 해당 특징 벡터의 클러스터 중심 번호로 대체하여 클러스터로 표현되는 이미지 세그먼트를 얻을 수 있습니다. 이 절차를 벡터 양자화(섹션 6.2.1)와 비교해야 합니다. 이 설명은 매우 일반적입니다. 서로 다른 특징 벡터는 서로 다른 클러스터러와 마찬가지로 서로 다른 종류의 이미지 세그먼트로 이어집니다.

특징 벡터와 클러스터러의 특정 조합이 좋은 결과를 내는지 여부

성능은 필요한 것에 따라 다릅니다. 그러나 몇 가지 일반적인 진술을 하는 것은 가능합니다. 일반 레시피는 세그먼트의 연결을 보장하지 않으며 이는 중요할 수도 있고 그렇지 않을 수도 있습니다. 이미지를 분할하여 압축하는 경우 미국 국기를 세 개의 세그먼트(빨간색, 흰색 및 파란색)로 인코딩하는 것이 좋습니다. 개체를 나타내기 위해 분할하는 경우 모든 흰색 별을 단일 세그먼트로 간주하기 때문에 이는 잘못된 표현일 수 있습니다. 특징 벡터에 픽셀 위치의 표현이 포함된 경우 세그먼트의 중심에서 매우 멀리 떨어진 픽셀이 다른 클러스터에 속하는 경향이 있기 때문에 결과 세그먼트는 "뭉툭한" 경향이 있습니다. 이것은 세그먼트가 연결되었는지 확인하는 한 가지 방법입니다. 색상 정보를 나타내는 것은 세그멘터를 더 좋게 만드는 경향이 있습니다. 이 경우 쉬운 이미지를 잘못 이해하기 어렵기 때문입니다(색상이 어려운 이미지를 더 쉽게 만드는 것 같지는 않음). 일부 응용 프로그램의 경우 쉬운 이미지를 잘 수행하는 것으로 충분합니다.

9.3.1 기본 클러스터링 방법

클러스터링에는 두 가지 자연 알고리즘이 있습니다. 분할 클러스터링에서는 전체 데이터 세트를 하나의 클러스터로 간주한 다음 클러스터를 재귀적으로 분할하여 좋은 클러스터링을 생성합니다(알고리즘 9.4). 응집 클러스터링에서는 각 데이터 항목을 클러스터로 간주하고 클러스터를 재귀적으로 병합하여 좋은 클러스터링을 생성합니다(알고리즘 9.3).

```
각 포인트를 별도의 클러스터로 만들기
클러스터링이 만족스러울 때까지
        클러스터 간 거리가 가장 작은 두 클러스터
            병합
끝
```

알고리즘 9.3: 응집 클러스터링 또는 병합에 의한 클러스터링.

```
모든 포인트를 포함하는 단일 클러스터 구성
클러스터링이 만족스러울 때까지
        클러스터 간 거리가 가장 큰 두 구성 요소를 생성
            하는 클러스터 분할
끝
```

알고리즘 9.4: 분할 클러스터링 또는 분할에 의한 클러스터링.

클러스터링에 대해 생각할 때 두 가지 주요 문제가 있습니다.

- 좋은 클러스터 간 거리는 얼마입니까? 집적 클러스터링은 클러스터 간 거리를 사용하여 인근 클러스터를 융합합니다. 분할 클러스터링은 이를 사용하여 충분히 일관된 클러스터로 분할합니다. 데이터 포인트 사이의 자연스러운 거리를 사용할 수 있더라도(시력 문제의 경우가 아닐 수 있음) 정식 클러스터 간 거리가 없습니다. 일반적으로 데이터 세트에 적합해 보이는 거리를 선택합니다. 예를 들어, 다음과 같은 거리를 선택할 수 있습니다.

확장된 클러스터를 생성하는 경향이 있는 클러스터 간 거리로 가장 가까운 요소 사이를 트위닝합니다(통계학자는 이 방법을 단일 링크 클러스터링이라고 함). 또 다른 자연스러운 선택은 첫 번째 클러스터의 요소와 두 번째 요소 사이의 최대 거리로, 둥근 클러스터를 생성하는 경향이 있습니다(통계학자는 이 방법을 완전 연결 클러스터링이라고 함). 마지막으로, "둥근" 클러스터를 생성하는 경향이 있는 클러스터의 요소 간 평균 거리를 사용할 수 있습니다(통계학자는 이 방법을 그룹 평균 클러스터링이라고 함).

- 얼마나 많은 클러스터가 있습니까? 클러스터를 생성한 프로세스에 대한 모델이 없는 경우 이는 본질적으로 어려운 작업입니다. 우리가 설명한 알고리즘은 클러스터의 계층 구조를 생성합니다. 일반적으로 이 계층 구조는 덴드로그램(클러스터 간 거리를 표시하는 클러스터 계층 구조의 표현)의 형태로 사용자에게 표시되며 덴드로그램에서 적절한 클러스터 선택이 이루어집니다(그림 9.15의 예 참조). .

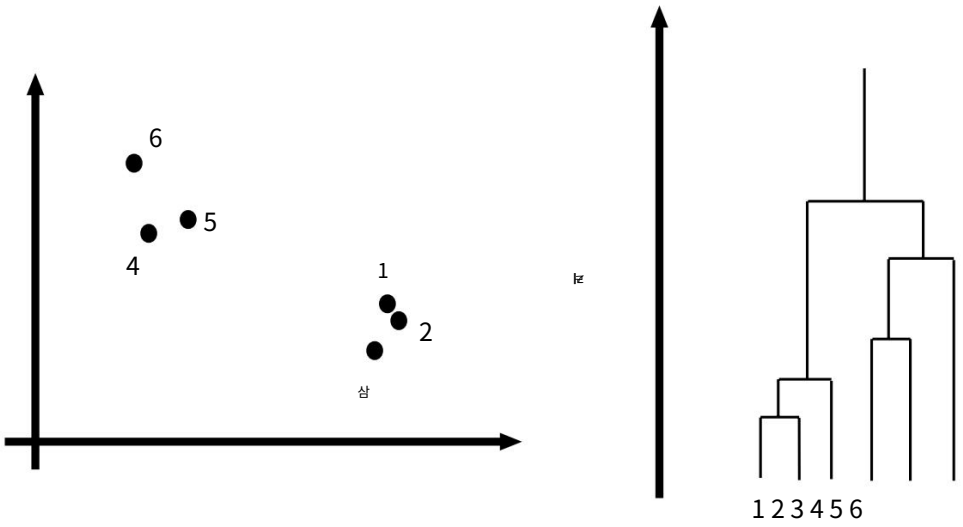


그림 9.15: 왼쪽, 데이터 세트; 오른쪽, 단일 링크 클러스터링을 사용한 응집 클러스터링으로 얻은 덴드로그램. 특정 거리 값을 선택하면 해당 거리의 수평선이 덴드로그램을 클러스터로 분할합니다. 이 표현을 통해 얼마나 많은 클러스터가 있는지 추측하고 클러스터가 얼마나 좋은지에 대한 통찰력을 얻을 수 있습니다.

응집 또는 분할 클러스터링 방법을 직접 사용하는 데 있어 가장 어려운 점은 이미지에 엄청나게 많은 픽셀이 있다는 것입니다. 데이터의 양이 너무 많다는 것을 의미하기 때문에 아덴드로그램을 검토할 합리적인 전망이 없습니다. 실제로 이는 세그먼터가 일련의 임계값 테스트를 사용하여 분할 또는 병합을 중지할 시기를 결정함을 의미합니다. 예를 들어 클러스터 사이의 거리가 충분히 낮거나 클러스터 수가 특정 값에 도달하면 집적 분할기가 병합을 중지할 수 있습니다. 분할 클러스터러는 결과 클러스터가 일부 유사성 테스트를 충족하면 분할을 중지할 수 있습니다.

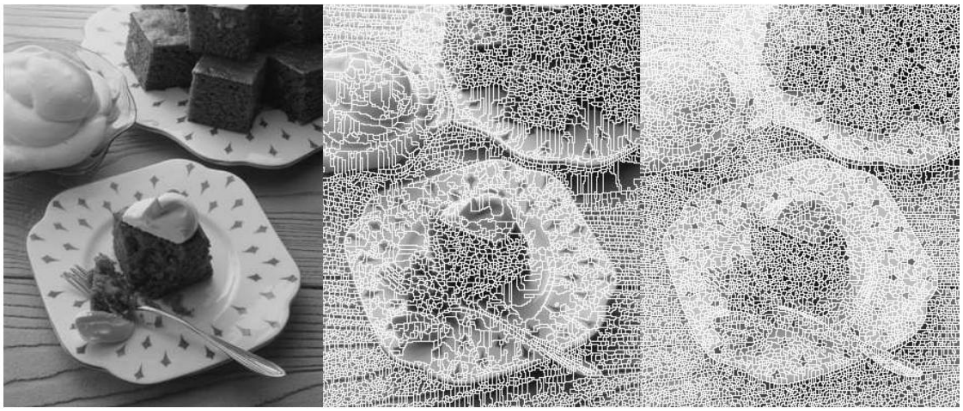


그림 9.16: Martin Brigdale이 이미지에 적용한 분수령 알고리즘의 분할 결과. 중앙: 이미지 강도에 적용되는 유역; 일부 긴 슈퍼픽셀을 확인하십시오. 오른쪽: 이미지 기울기 크기에 적용된 유역; 이것은 더 둥근 슈퍼픽셀을 생성하는 경향이 있습니다. Martin Brigdale c Dorling Kindersley, 허가 하에 사용.

분할 및 응집 클러스터러를 모두 수정하여 영역이 연결되도록 하는 것은 간단합니다. 집적 클러스터러는 공유 경계가 있는 클러스터만 병합해야 합니다. 모든 분할의 자식이 연결되도록 해야 하는 분할 클러스터러를 수정하는 것이 더 어렵습니다. 이를 수행하는 한 가지 방법은 분할되는 세그먼트의 공간 경계를 따라 분할하는 것입니다. 클러스터의 최상의 분할(분할 방법) 또는 최상의 병합(집합 방법)을 찾는 것은 일반적으로 비실용적입니다. 분할 방법은 일반적으로 좋은 분할을 제안하기 위해 클러스터의 요약 형식을 사용하여 수정됩니다(예: 픽셀 색상의 히스토그램). 집적 방법도 수정할 필요가 있는데, 픽셀 수가 많다는 것은 클러스터 간 거리(클러스터 무게 중심 간 거리가 자주 사용됨)에 주의해야 함을 의미하기 때문입니다.

마지막으로 가장 가까운 쌍을 검색하는 대신 단순히 이미지를 스캔하고 거리가 임계값 미만인 모든 쌍을 병합하여 영역을 병합하는 것이 유용할 수 있습니다.

9.3.2 유역 알고리즘

여전히 널리 사용되는 초기 분할 알고리즘은 워터셰드 알고리즘입니다.
이미지 I를 분할한다고 가정합니다. 이 알고리즘에서 이미지 기울기 크기 $||\nabla I||$의 맵을 계산합니다. 이 맵의 0은 국부적으로 극단적인 강도 값입니다. 우리는 각각을 세그먼트의 시드로 취하고 각 시드에 고유한 레이블을 지정합니다.
이제 높이 맵을 물(그래서 이름)로 채우는 것과 다소 대략적으로 유사한 절차를 통해 시드에 픽셀을 할당합니다. 픽셀(i, j)에서 시작한다고 상상해 보십시오. $||\nabla I||$의 기울기 아래로 뒤로 이동하면 고유한 시드에 도달하게 됩니다. 각 픽셀은 이 절차에 의해 적중된 시드의 레이블을 가져옵니다.

이 설명을 최단 경로 알고리즘의 한 형태로 인식해야 합니다. 그것은 또한 agglomerative clusterer의 한 형태로 볼 수 있습니다. 시드 클러스터로 시작한 다음 클러스터로의 경로가 "내리막"일 때 픽셀을 클러스터로 집결합니다.

픽셀에서. 이는 우리가 스케치한 것보다 훨씬 더 효율적인 알고리즘을 생성할 수 있으며 이러한 알고리즘에 대한 상당한 문헌이 있음을 의미합니다. 이 문헌에서 저자는 과도한 분할, 즉 "너무 많은" 세그먼트를 생성하는 워터셰드 알고리즘을 비판하는 경향이 있습니다. 보다 최근에는 유역 알고리즘이 허용 가능한 슈퍼 픽셀을 생성하고 효율적이기 때문에 상당히 널리 사용됩니다. 유역 알고리즘의 좋은 구현은 널리 사용 가능합니다. 그림 9.16을 생성하기 위해 Matlab의 이미지 처리 도구 상자에서 구현을 사용했습니다. 그라데이션 크기를 사용하여 유역 변환을 유도하는 것은 자연스러운 일입니다. 이는 이미지를 상대적으로 작은 그라데이션 영역으로 나누기 때문입니다. 그러나 이미지 강도를 사용할 수도 있으며, 이 경우 각 영역은 강도 최소 또는 최대의 매력 도메인입니다. Gradient watershed는 더 유용한 슈퍼픽셀을 생성하는 경향이 있습니다(그림 9.16).

9.3.3 K-평균을 사용한 분할

우리가 설명한 이미지 분할과 벡터 양자화 사이에는 강한 공명이 있습니다. 6장에서는 텍스처 표현의 맥락에서 벡터 양자화를 설명하고 k-평균 알고리즘을 소개했습니다. K는 일부 응용 프로그램에 대해 좋은 이미지 세그먼트를 생성한다는 것을 의미합니다. 일반 레시피에 따라 각 픽셀을 나타내는 특징 벡터를 계산하고 k-평균을 적용하며 각 픽셀은 특징 벡터를 요구하는 클러스터 중심으로 표시되는 세그먼트로 이동합니다. k-평균을 사용하는 주요 결과는 얼마나 많은 세그먼트가 있는지 알 수 있다는 것입니다. 일부 응용 프로그램의 경우 이것은 좋은 것입니다. 예를 들어, 그림 9.17의 분할은 5개의 세그먼트를 사용하고 기본적으로 이미지 회색 수준(또는 각각 색상)을 5개 수준으로 재양자화하는 것을 나타냅니다. 이는 일부 코딩 및 압축 응용 프로그램에 유용할 수 있습니다.

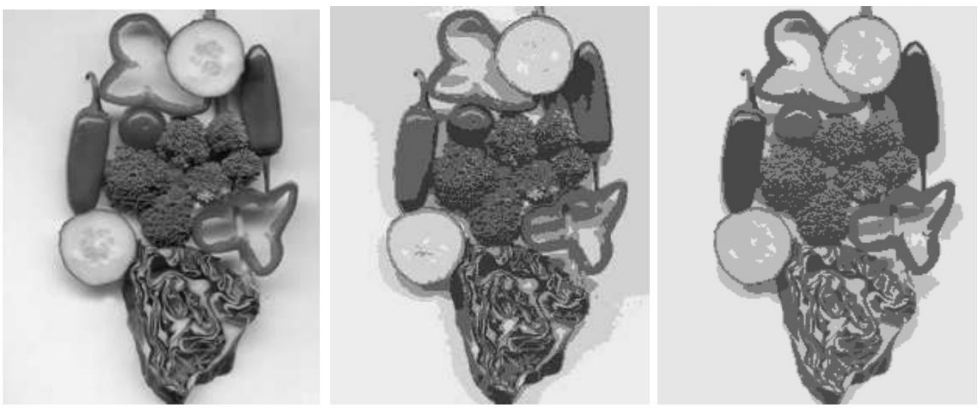


그림 9.17: 왼쪽에는 k-평균을 사용하여 분할된 혼합 야채 이미지가 있으며 중앙과 오른쪽에 이미지가 생성됩니다. 각 픽셀을 해당 클러스터의 평균값으로 대체했습니다. 그 결과는 예상대로 적응형 재양자화와 비슷합니다. 중앙에는 강도 정보만을 사용하여 얻은 분할이 있습니다.

오른쪽은 색상 정보를 사용하여 얻은 분할입니다. 각 분할은 5개의 클러스터를 가정합니다.

이미지 분할에 이 접근 방식을 사용할 때 한 가지 어려운 점은 세그먼트가

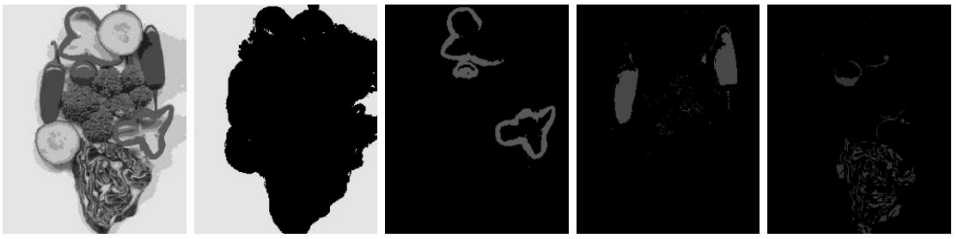


그림 9.18: 여기서는 11개의 구성 요소 집합을 가정하여 k-평균으로 분할된 야채 이미지를 보여줍니다. 왼쪽 그림은 원래 이미지 값 대신 평균값을 사용하여 함께 표시된 모든 세그먼트를 보여줍니다. 다른 그림은 4개의 세그먼트를 보여줍니다.

이 접근 방식은 반드시 연결되지 않은 일련의 세그먼트로 이어집니다.
이 이미지의 경우 일부 세그먼트는 실제로 개체와 매우 밀접하게 연결되어 있지만 하나의 세그먼트는 많은 개체(고추)를 나타낼 수 있습니다. 다른 것들은 대체로 의미가 없습니다. 적양배추 슬라이스로 인해 발생하는 다양한 세그먼트가 나타내는 것처럼 질감 측정이 없으면 심각한 어려움이 발생합니다.

연결되지 않고 매우 광범위하게 분산될 수 있습니다(그림 9.17 및 9.18). 이 효과는 픽셀 좌표를 기능으로 사용하여 줄일 수 있습니다. 이 접근 방식은 큰 영역이 분할되는 결과를 가져옵니다(그림 9.19).

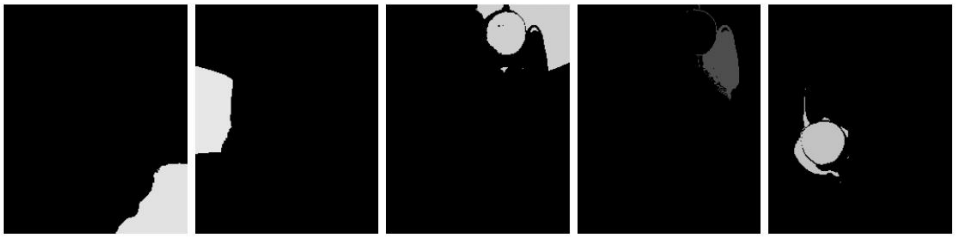


그림 9.19: 위치를 픽셀을 설명하는 특징 벡터의 일부로 사용하는 k-평균 분할기로 야채 이미지를 분할하여 얻은 세그먼트 중 5개는 이제 11개가 아닌 20개의 세그먼트를 사용합니다. 포인트가 중심에서 너무 멀리 떨어져 있기 때문에 일관성이 끊어졌습니다. 개별 고추는 이제 더 잘 분리되지만 붉은 양배추는 질감 측정이 없기 때문에 여전히 잘립니다.

9.3.4 평균 이동: 데이터에서 로컬 모드 찾기

클러스터링은 밀도 추정 문제로 추상화할 수 있습니다. 일부 기본 확률 밀도에서 나온 일부 기능 공간에 샘플 포인트 세트가 있습니다. Comaniciu와 Meer(2002)는 클러스터를 이 밀도에서 로컬 최대값(로컬 모드)으로 생각하는 평균 이동 알고리즘을 사용하여 매우 중요한 세그멘터를 만들었습니다. 이를 위해서는 대략적인 밀도 표현이 필요합니다. 근사치를 구축하는 한 가지 방법은 커널 스무딩을 사용하는 것입니다. 여기서 우리는 "블롭" 또는 "범프"처럼 보이는 함수 세트를 가져와 각 데이터 포인트 위에 하나씩 배치하여 많은 데이터 포인트가 서로 가까이 있을 때 크고 데이터 포인트가 넓을 때 작은 부드러운 함수를 생성합니다. 분리.

이것은 다양한 범프 기능에 적용되는 매우 일반적인 전략입니다.
우리는 특정 커널을 더 매끄럽게 사용하여 작성합니다.

$$\text{케이(엑스; h)} = \frac{(2\pi)\ (\quad d/2)}{h\,d}\ 1\ \text{경험치}\ -\ \frac{||x||2}{2}$$

범프 기능을 위해. 최상의 표현을 얻기 위해 조정할 수 있는 양의 척도 매개변수 h를 도입합니다. 그런 다음 밀도의 모델은 다음과 같습니다.

$$\text{에프(엑스)} = \frac{1}{N}\sum_{i=1}^{N} K(xi - x; h)$$

(이것이 밀도인지 확인해야 합니다. 즉, 음수가 아닌지, 적분이 1인지 확인해야 합니다). 보류된 데이터의 평균 우도를 최대화하여 h를 추정할 수 있습니다. 이 절차에서는 다음 실험을 여러 번 반복합니다. 하나의 데이터 포인트를 임의로 유지하고 밀도 모델을 나머지 데이터에 맞춘 다음 보류된 데이터 포인트의 우도를 h의 함수로 계산합니다(아마도 계산을 통해). h의 다른 샘플 포인트 집합에서의 값). 이런 식으로 계산된 우도를 평균화한 다음 해당 우도를 최대화하는 h를 사용합니다.

$k(u) = \exp$ $\frac{1}{2}$유(이것을

커널 프로파일) 및 $C = nhd$, $\frac{(2\pi)^{(\ d/2)}}{}$

$$\text{에프(엑스)} = C\sum_{i=1}^{N} \text{케이}\ ||\ \frac{x - xi}{\text{사전}}\ ||^{2} \qquad (9.1)$$

(로컬 모드)을 클러스 디 (유). 어떤 점 x0 에서 시작하여 이제 밀도 값을 최대화하는 근처 점 g를 씁니다. 이 로컬 최대값터 센터로 사용합니다. 평균 이동 절차는 방정식 9.1의 형태로 표현을 최대화합니다. 기울기 ∇f가 그 지점에서 사라지도록 y를 찾고 있습니다. 우리는 그것을 요구해야 합니다

$$\nabla f(x)\ |x=y = 0$$

$$= \text{씨}\quad k(||\ ||\ h\frac{xi\quad y}{}^{2})$$

$$= Ch\ \frac{2}{}^{\text{나}}\quad - y]\ g(||\ ||\ h\quad \frac{xi - y\ [xi}{}^{2})$$

$$= Ch\ \frac{2}{}^{\text{나}}\quad \frac{\text{나 다음}(||\ \frac{xi\quad y}{\text{사전}}\ ||2)}{\text{나}\ \frac{xi\text{-}y\ g(||\ h}{}\ ||2)}\ -\ \text{및}\ \times\quad ||\ h\ \frac{xi - yg(||}{}^{2})\ .$$

우리는 그것을 기대합니다 $_{\text{나}}\ g(||\ \frac{xi\text{-}y\ h}{}\ ||2)$는 0이 아니므로 최대값은 다음과 같은 경우에 발생합니다.

$$\frac{_{\text{나}}\ \text{다음}(\ ||\ \frac{xi\text{-}y\ h}{}\ ||2)}{_{\text{나}}\ g(||\ \frac{xi\text{-}y\ h}{}\ ||2)}\ -y = 0,$$

또는 동등하게, 때

$$\text{그리고} = \frac{\sum_{i} x_i g(\| h \frac{x_i - y}{} \|_2)}{\sum_{i} g(\| \frac{x_i - y}{h} \|_2)}.$$

평균 이동 절차에는 일련의 추정값 y 생성이 포함됩니다. (j) 어디서

$$y^{(d+1)} = \frac{\sum_{i} 다음(\| \frac{x_i - y^{(j)}}{h} \|^2)}{\sum_{i} g(\| \frac{x_i - y^{(j)}}{h} \|^2)}.$$

이 절차는 가중 평균 형태의 점으로 이동한다는 사실에서 이름을 얻었습니다(알고리즘 9.5 참조).

차원 d의 모드 y, 스케일링 상수 h, 커널 프로파일의 도함 (0) 및 n 데이터 벡터 세트 xi
수 g의 추정으로 시작합니다.

업데이트가 작을 때까지
새로운 견적을 형성

$$y^{(d+1)} = \frac{\sum_{i} 다음(\| \frac{x_i - y^{(j)}}{h} \|^2)}{\sum_{i} g(\| \frac{x_i - y^{(j)}}{h} \|^2)}$$

알고리즘 9.5: 평균 이동으로 모드 찾기

9.3.5 평균 이동을 사용한 클러스터링 및 분할

평균 이동을 사용한 클러스터링은 원칙적으로 간단합니다. 모든 데이터 포인트에서 평균 이동 절차를 시작하여 각 데이터 포인트에 대한 모드를 생성합니다. 우리는 연속 변수로 작업하고 있기 때문에 이러한 모드는 모두 다르지만 모드가 매우 밀접하게 클러스터링될 것으로 예상합니다. 실제 모드의 작은 집합이 있어야 하며 이러한 각각의 추정치는 그중 하나에 매우 가깝습니다. 이러한 추정치는 이미지 필터링의 한 형태를 나타내기 때문에 그 자체로 유용합니다.

각 픽셀을 모드 표현으로 바꿀 수 있습니다. 이것은 이미지 경계를 존중하는 방식으로 이미지를 상당히 매끄럽게 만듭니다(그림 9.20). 데이터를 클러스터링하기 위해 모드 추정치에 집적 클러스터러를 적용합니다.

모드가 매우 밀접하게 클러스터링될 것으로 예상하기 때문에 그룹 평균 거리는 좋은 거리 선택이며 이 거리가 작은 임계값을 초과하면 클러스터링을 중지할 수 있습니다. 이렇게 하면 광범위하게 분리된 작고 조밀한 클러스터 집합이 생성됩니다. 이제 각 데이터 포인트를 해당 모드에 해당하는 클러스터 중심에 매핑합니다(알고리즘 9.6).

이 방법은 이미지 분할에 거의 직접적으로 적용할 수 있습니다. 각 픽셀을 특징 벡터로 표현한 다음 특징 벡터를 클러스터링합니다. 각 클러스터 중심은 세그먼트를 나타내며 각 픽셀을 클러스터 중심의 번호로 바꿉니다. 보다 명시적으로 공간과 외관 특징의 균형을 맞추는 표현으로 향상된 성능을 얻을 수 있습니다. 특히 우리는 i번째 픽셀을 두 가지 요소를 가진 특징 벡터 xi 로 표현합니다 . x는 차원을 가지고 있습니다.

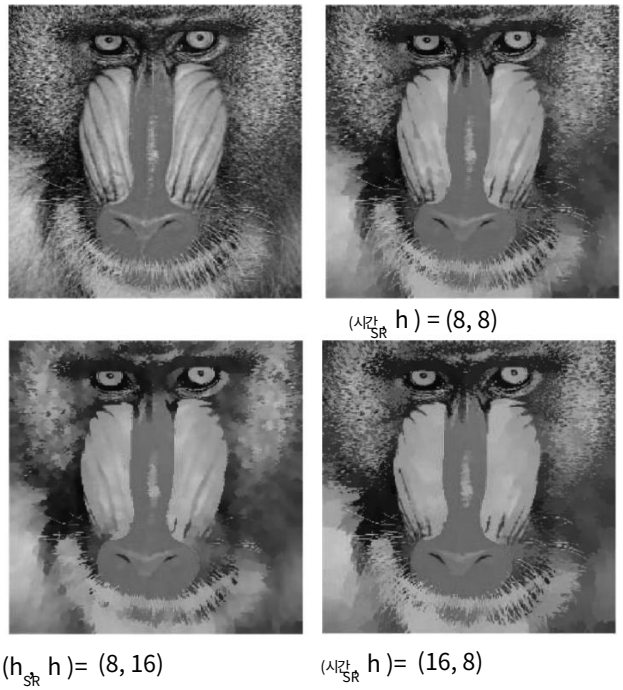


그림 9.20: 공간 hs 및 외관 시간 에 대해 서로 다른 클러스터링 스케일로 얻은 이미지(왼쪽 상단) 및 평균 이동 모드 . hs 가 작은 경우 커널 함수가 상대적으로 작은 반지름에서 매끄럽게 처리되어 많은 고유 모드를 허용하기 때문에 메서드는 공간적으로 상대적으로 작고 컴팩트한 클러스터를 생성해야 합니다. hr 이 작 으면 클러스터 모양이 콤팩트합니다. 즉, 작은 hs 와 큰 hr는 모양의 범위에 걸쳐 있을 수 있는 작고 얼룩진 군집을 생성하는 반면, 큰 h 와 작은 hr는 작은 범위의 모양을 가진 공간적으로 복잡하고 확장된 군집을 만드는 경향이 있습니다.

클러스터 경계는 수준의 강도 곡선을 따르기 위해 더 열심히 노력할 것입니다. 이 그림은 원래 D. Comaniciu 및 P. Meer의 "Mean Shift: A Robust Approach Toward Feature Space Analysis"의 그림 5, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002 c IEEE, 2002로 게시되었습니다.

각 데이터 포인트 xi(0) 에 대
해
y = xi 로 시작하는 평균이동 절차(알고리즘 9.5)를 적용합니다. 결과 모드를 yi 로 기록합니다.

작고 조밀한 군집을 형성해야 하는 yi 를 군집화합니다 .
좋은 선택은 그룹 평균 거리,
그룹 평균 거리가 작은 임계값을 초과하면 클러스터링 중지

데이터 포인트 xi는 모드 yi가 속한 클러스터에 속합니다 .

알고리즘 9.6: 평균 시프트 클러스터링.

ds 는 픽셀의 위치를 나타내고 x는 차원 dr을 가지며 다른 모든 것을 나타냅니다. 이제 밀도 추정 절차에서 두 개의 커널과 두 개의 스무딩 매개변수를 사용하여 다음과 같이 작성합니다.

$$K(x; hs, hr) = \frac{(2\pi)(ds/2)}{h_s^{ds}} \text{ 케이 } \frac{\text{엑스}^{\text{ds}}}{hs} \quad \frac{(2\pi)(dr/2)}{h_r^{dr}} \text{ 케이 } \frac{\text{엑스}^{\text{dr}}}{\text{시간}} .$$

즉, 공간 및 모양 군집화의 균형을 맞출 수 있으며 예를 들어 다양한 모양을 가진 공간적으로 긴밀한 군집 등이 필요합니다. 이 경우 평균 이동 업데이트 방정식이 약간 변경됩니다(예제).

각 픽셀 pi 에 대해 특징 벡터 xi = (x s 공간 및 모양 구성 요소를 각각 계산합니다 , 나는) 대표
다.

스무딩 커널의 공간적(각각 모양) 척도인 hs, hr 를 선택합니다 .

이 데이터와 평균 시프트 클러스터링을 사용하여 xi 를 클러스터링합니다 (알고리즘 9.6).

(선택 사항) tmin 픽셀 미만의 클러스터를 이웃과 병합합니다. 클러스터가 작기 때문에 이웃 선택은 중요하
지 않습니다.

i번째 픽셀은 클러스터 중심에 해당하는 세그먼트에 속합니다(예를 들어 클러스터 중심 1...r에 레이블을 지정
한 다음 픽셀에 해당하는 레이블의 맵을 계산하여 세그먼트를 식별할 수 있음).

알고리즘 9.7: 평균 시프트 세분화.

9.4 분할, 클러스터링 및 그래프

클러스터링 알고리즘은 데이터 항목 간의 유사성을 처리합니다. 일부 알고리즘은 데이터 항목을 요약할 수 있지만(예: k-평균 알고리즘의 클러스터 중심) 핵심 문제는 데이터 항목 간의 유사성입니다. 데이터 항목의 모든 쌍을 비교하는 것은 유용하지 않을 수 있습니다. 예를 들어, 매우 멀리 있는 이미지 픽셀을 직접 비교하는 것은 실질적인 이점이 거의 없을 수 있습니다. 이 모든 것은 오히려 자연스럽게 그래프를 제안합니다. 각 데이터 항목은 정점이 됩니다. 유용하게 비교할 수 있는 모든 데이터 항목 쌍 사이에 가중 에지가 있습니다. 그런 다음 데이터를 클러스터링하는 프로세스는 그래프를 연결된 구성 요소로 분할하는 프로세스 중 하나가 됩니다.

9.4.1 그래프에 대한 용어 및 정보

잊어버리기 쉽기 때문에 여기에서 용어를 매우 간략하게 검토합니다.

- 그래프는 다양한 정점 쌍을 연결하는 정점 V와 모서리 E의 집합입니다. 그래프는 G = {V, E}로 쓸 수 있습니다. 각 모서리는 한 쌍의 정점으로 표현될 수 있습니다. 즉, E ⊂ V × V 입니다. 그래프는 종종 점을 연결하는 곡선이 있는 점 집합으로 그려집니다.

- 꼭짓점의 차수는 해당 꼭짓점에 입사하는 가장자리의 수입니다.

그림 9.21: 평균 이동 알고리즘을 사용하여 얻은 이미지의 분할. 이 그림은 원래 D. Comaniciu 및 P. Meer의 "Mean Shift: A Robust Approach Toward Feature Space Analysis"의 그림 10, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002 c IEEE, 2002로 게시되었습니다.

- 유향 그래프는 에지 (a, b)와 (b, a)가 구별되는 그래프입니다. 이러한 그래프는 의도된 방향을 나타내는 화살촉으로 그려집니다.
- 무방향 그래프는 에지 사이에 구분이 없는 그래프입니다. (a, b) 및 (b, a).
- 가중 그래프는 각 에지에 가중치가 연결된 그래프입니다.
- 공통 정점이 있는 두 모서리는 연속적입니다.
- 경로는 연속된 에지의 시퀀스입니다.
- 회로는 시작하는 정점에서 끝나는 경로입니다.
- 자체 루프는 각 끝에 동일한 정점이 있는 에지입니다. 자가 루프는 우리의 응용 프로그램에서 발생합니다.
- 한 정점에서 시작하여 다른 정점에서 끝나는 일련의 가장자리가 있을 때 두 정점이 연결되었다고 합니다. 그래프가 방향이 있는 경우 이 시퀀스의 화살표는 올바른 방향을 가리켜야 합니다.
- 연결된 그래프는 모든 정점 쌍이 연결된 그래프입니다.
- 트리는 회로가 없는 연결된 그래프입니다.

- 연결된 그래프 G = {V, E}가 주어지면 스패닝 트리는 꼭지점이 V이고 가장자리가 E의 부분 집합인 트리입니다. 우리의 정의에 따르면 트리는 연결되어 있으므로 스패닝 트리가 연결됩니다.

- 모든 그래프는 연결된 구성요소의 서로소 집합으로 구성됩니다. 즉, G = {V1 ∪ V2 ...Vn, E1 ∪ E2 ... En}, 여기서 {Vi , Ei} 는 모두 연결된 그래프이고 간선이 없습니다. i = j에 대해 Vi 의 요소를 Vj 중 하나와 연결하는 E에서 .

- 숲은 연결된 구성 요소가 나무인 그래프입니다.

가중 그래프에는 최소 가중치 스패닝 트리를 계산하기 위한 효율적인 알고리즘이 있습니다(예: Jungnickel(1999) 또는 Cormen et al.(2009) 참조).
효율적으로 해결할 수 있는 또 다른 매우 중요한 문제는 유향 그래프에서 흐름을 최대화하는 것입니다. 특히 유향 그래프에서 한 정점을 소스 s로, 다른 정점을 목표 t로 식별합니다. 음이 아닌 숫자인 각 방향 에지 ea 용량 c(e)와 연결합니다. 흐름은 다음 속성을 가진 각 에지와 관련된 음수가 아닌 값 f(e)입니다. 첫째, 0 ≤ f(e) ≤ c(e). 둘째, 임의의 정점 v ∈ {V s t}에서,

f(e) f(e) = 0 v 에서 귀가 v에서 출
발

(즉, 정점에 도달하는 모든 흐름은 정점을 떠납니다. 이것이 Kirchoff의 법칙입니다). 흐름의 가치는

요
정). 도착하는 귀

예를 들어 Ahuja et al.에서 흐름을 최대화하는 효율적인 알고리즘이 있습니다. (1993) 또는 Cormen et al. (2009). 이중 문제도 흥미 롭습니다. 정점을 s ∈ S 및 t ∈ T 가 되도록 두 개의 서로소 집합 S 및 T 로 분해합니다. 이것은 컷을 나타냅니다. S 에서 T 까지의 방향성 에지 세트인 W ∈ E를 고려하십시오. 컷 값은

있다).
e∈W

절단 값을 다시 효율적으로 최소화할 수 있습니다. 알고리즘은 예를 들어 Ahuja et al. (1993), Jungnickel(1999) 또는 Schrijver(2003).

9.4.2 그래프를 사용한 응집 클러스터링

Felzenszwalb 및 Huttenlocher(2004)는 그래프 이론적 아이디어를 사용하여 응집체 클러스터러를 기반으로 하는 간단하지만 매우 효과적인 세그멘터를 구축하는 방법을 보여주었습니다. 이미지를 가중 그래프로 나타냅니다. 이웃 픽셀 쌍 사이에는 가장자리가 있습니다. 각 가장자리에는 비유사성을 측정하는 가중치가 있습니다. 즉, 픽셀이 매우 다르면 가중치가 크고 비슷하면 가중치가 작습니다. 가중치는 다양한 픽셀 표현에서 나올 수 있습니다. 예를 들어 강도의 제곱 차이를 사용할 수 있습니다. 벡터로 각 픽셀의 색상을 나타낼 수 있고 차이 벡터의 길이를 사용할 수 있습니다. 필터 출력 벡터로 각 픽셀의 텍스처를 나타낼 수 있습니다(

클러스터 집합 Ci 에서 픽셀당 하나의 클러스터로 시작합니다.
가장자리 가중치가 감소하지 않는 순서대로 가장자리를 정렬하여 w(e1) ≥ w(e2) ≥ ... ≥
w(er)가 되도록 합니다.

i = 1 ~ r의 경우
    에지 ei가 클러스터 내부에 있는 경우 아무 작업도
        수행하지 않음
    또 다른
        diff(Cl , Cm) ≤ M Int(Cl , Cm) 이면 한쪽 끝은 클러스터 Cl 에 있고 다른 쪽
        끝은 클러스터 Cm 에 있습니다.
            Cl 과 Cm을 병합하여 새로운 클러스터 세트를 생성합니다.

나머지 클러스터 집합을 보고합니다.

알고리즘 9.8: 그래프를 사용한 응집 클러스터링.

섹션 6.1의 로컬 텍스처 표현) 차이 벡터의 길이를 사용합니다. 또는 이 모든 거리의 가중 합계를 사용할 수 있습니다.

클러스터를 형성하는 모든 픽셀로 시작한 다음 계속할 필요가 없을 때까지 클러스터를 병합합니다. 이를 위해서는 두 군집 사이의 거리에 대한 개념이 필요합니다. 각 클러스터는 클러스터의 모든 정점(픽셀)과 클러스터 내에서 시작하고 끝나는 모든 가장자리로 구성된 그래프의 구성 요소입니다. 그러면 두 구성 요소 간의 차이는 두 구성 요소를 연결하는 최소 가중치 가장자리입니다. 두 구성 요소에 대해 C1, C2 , 가장자리에 대해 E, v1과 v2 를 연결하는 가장자리의 가중치에 대해 w(v1 , v2) 를 씁니다 . 그런 다음

$$diff(C1, C2) = w(v1, v2). \text{ 최소 } v1 \in C1, v2 \in C2, (v1,v2) \in E$$

특정 클러스터가 얼마나 일관성이 있는지 아는 것도 도움이 됩니다. 이렇게 하면 클러스터링을 중지하는 데 도움이 됩니다. 구성 요소의 내부 차이를 구성 요소의 최소 스패닝 트리에서 가장 큰 가중치로 정의합니다. C의 최소 스패닝 트리에 대해 M(C) = {VC, EM }이라고 씁니다. 그런 다음,

$$int(C) = \text{최대 } w(e). \ e \in M(C)$$

모든 픽셀로 구성된 클러스터(또는 세그먼트) 세트(픽셀당 하나의 클러스터)로 시작합니다. 그런 다음 클러스터를 반복적으로 병합합니다. 가장자리 가중치가 감소하지 않는 순서대로 모든 가장자리를 정렬하여 그렇게 합니다. 각 에지에 대해 가장 작은 것부터 시작하여 에지의 양쪽 끝에 있는 클러스터를 고려합니다. 에지의 양쪽 끝이 같은 클러스터에 있으면 할 일이 없습니다. 각 끝에 고유한 클러스터가 있는 경우 병합할 수 있습니다. 각 클러스터의 내부 차이에 비해 에지 가중치가 작을 때 그렇게 합니다(작은 클러스터에 대한 주의가 필요합니다. 자세한 내용은 아래 참조). 이제 필요에 따라 병합하면서 모든 가장자리를 진행합니다. 최종 분할은 마지막 에지를 방문한 클러스터 집합입니다(알고리즘 9.8).

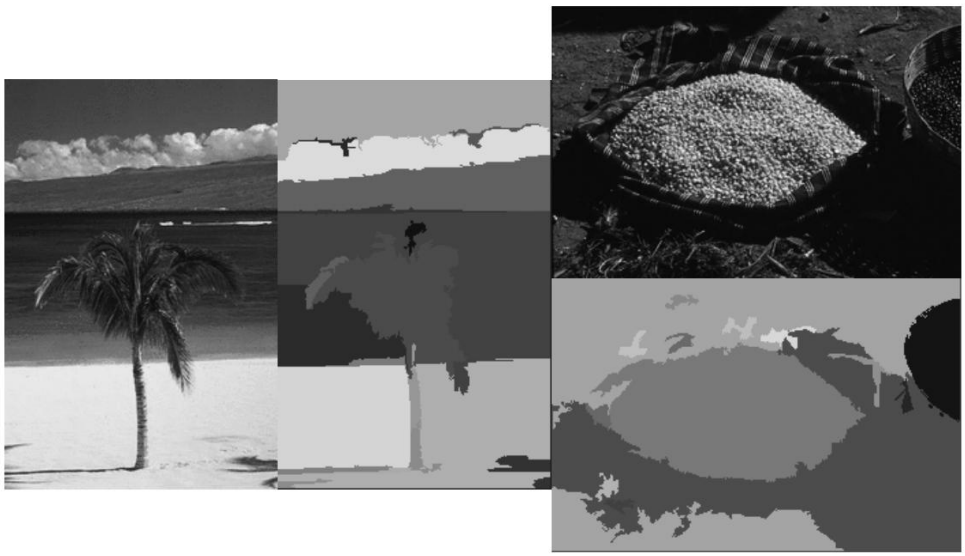


그림 9.22: 세그먼트 옆에 표시되는 알고리즘 9.8을 사용하여 분할된 이미지. http://people.cs.uchicago.edu/~pff/segment/에서 Pedro Felzenszwalb의 허락을 받아 얻은 수치.

에지 가중치를 클러스터의 내부 차이와 비교하려면 약간의 주의가 필요합니다. 작은 클러스터에서는 내부 거리가 0이거나(꼭짓점이 하나만 있는 경우) 터무니없이 작을 수 있기 때문입니다. 이를 처리하기 위해 Felzenszwalb 및 Huttenlocher(2004)는 MInt라는 두 클러스터의 함수를 다음과 같이 정의합니다.

$$\mathrm{MInt}(C1, C2) = \min(\mathrm{int}(C1) + \tau(C1), \mathrm{int}(C2) + \tau(C2))$$

여기서 τ(C)는 작은 군집에 대해 내부 차이를 위쪽으로 바이어스하는 항입니다. Felzenszwalb 및 Huttenlocher(2004)는 τ(C) = k/ | C |, k에 대해 일부 상수 매개변수. 이 알고리즘은 매우 빠르고 상대적으로 정확합니다(그림 9.22).

9.4.3 그래프를 사용한 분할 클러스터링

9.2.3 절에서 살펴본 것처럼 예제를 기반으로 이미지를 전경과 배경으로 분리하는 것은 매우 유용합니다. 이미지 픽셀당 하나씩 픽셀 맵이 있다고 가정합니다. 여기서 맵의 각 픽셀은 해당 이미지가 전경에 있는지 여부에 따라 전경, 배경 또는 알 수 없음(이러한 맵은 트리맵이라고도 함)의 세 가지 레이블 중 하나를 가지고 있습니다. , 배경 또는 알 수 없음. 우리는 전경 및 배경 픽셀을 가져와 이들로부터 모델을 구축한 다음 이러한 모델로 알 수 없는 픽셀에 레이블을 지정해야 합니다. 레이블에는 두 가지 중요한 제약 조건이 있습니다. 먼저 전경 예제처럼 보이는 픽셀은 전경 레이블을 가져와야 합니다(배경과 유사). 둘째, 픽셀은 이웃과 동일한 레이블을 갖는 경향이 있습니다.

Boykov와 Jolly(2001)는 이 문제를 에너지 최소화라고 표현했습니다. 전경 레이블이 있는 픽셀 집합에 대해 F를, 배경 레이블이 있는 집합에 대해 B를, 알 수 없는 픽셀에 대해 U를 씁니다. 이진 변수 δi를 i번째 미지수와 연결합니다.

픽셀. 우리는 i번째 픽셀이 배경이면 δi = -1이고 i번째 픽셀이 전경이면 δi = 1이라는 규칙을 채택합니다 . 에너지 함수를 최소화하는 이 이진 변수의 값을 찾을 것입니다. 에너지에는 두 가지 유형의 용어가 있습니다. 첫 번째 유형의 용어는 전경(각 배경) 모델과 유사한 픽셀이 전경(각 배경) 레이블을 갖도록 권장합니다. 두 번째는 픽셀이 이웃과 동일한 레이블을 갖도록 권장합니다.

i번째 픽셀을 나타내는 pi foravector를 씁니다 . 벡터는 강도를 포함할 수 있습니다. 강도 및 색상; 강도, 색상 및 질감; 또는 기타 정보.
픽셀 벡터 p를 전경 모델과 비교하는 함수에 대해 df (p)를 작성하십시오 . 이 기능은 픽셀이 전경과 같지 않을 때 크고, 전경과 같을 때 작습니다. 마찬가지로 픽셀 벡터를 배경과 비교하는 함수에 대해 db(p)를 작성합니다.
픽셀 i의 이웃에 대해 N(i)를 씁니다. 두 개의 픽셀을 비교하는 음이 아닌 대칭 함수에 대해 B(pi , pj)를 작성합니다. 이웃 픽셀을 다른 모델에 할당하기 위한 비용으로 사용할 것입니다. 이는 인접한 픽셀이 동일한 레이블을 갖도록 권장하는 상수만큼 간단할 수 있습니다. 더 복잡한 B는 유사한 두 픽셀의 경우 크고 다른 픽셀의 경우 작아야 합니다. 이 경우 레이블이 다르게 보이는 픽셀 사이에서 변경되도록 권장합니다.

그것을주의해라 ($\frac{1}{2}$)(1 δiδj)는 δi 와 δj가 다르면 1, 아니면 0의 값을 갖는다 . 모든 픽셀 집합에 대해 I, 알 수 없는 픽셀 집합에 대해 U, 알려진 전경 픽셀 집합에 대해 F, 알려진 배경 픽셀 집합에 대해 B를 씁니다.
이제 에너지 함수를 작성할 수 있습니다.

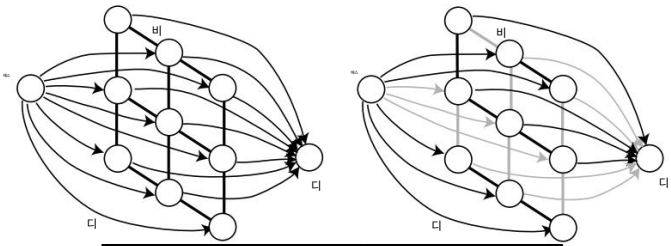
$$\text{그림 } (d) = \sum_{i \in I} db(\text{파이})\frac{11}{2}(1 \quad \delta i) + \frac{df(\text{파이})(1+\delta i) + }{2}$$

$$\sum_{i \in I\, j \in N(i)} B(\text{파이}, pj)(1\bar{}\,)(1 - \delta\imath\delta j)\, 2$$

k ∈ F의 경우 δk = 1, k ∈ B의 경우 δk = 0 으로 최소화해야 합니다. 전경 모델과 일치하는 픽셀을 δ = 1로 표시하여 이 에너지 함수를 작게 만들 수 있습니다. δ = 1인 배경 모델과 다르게 보이는 픽셀(즉, B가 작은 경우)에서 레이블이 변경되도록 합니다. 이 에너지를 최소화하는 것은 조합 문제이기 때문에 어려울 수 있습니다 (δj는 두 값만 가질 수 있음).

E를 최소화하는 것은 그래프의 컷을 최소화하는 것으로 바꿔 말할 수 있습니다. 이것을 보는 가장 쉬운 방법은 그림을 사용하는 것입니다. 그림 9.23의 그래프에서 컷을 상상해 보십시오. 이 그래프에서 각 픽셀은 정점으로 표시되고 소스 정점은 전경 레이블에 해당하고 대상 정점은 배경 레이블에 해당합니다. 각 픽셀을 소스에 연결하는 하나의 에지와 타겟에 연결하는 하나의 에지가 있습니다. 이 두 가장자리 중 하나만 잘라서 그래프를자를 수 있으며 둘 다 자르면 잘림이 최소화되지 않습니다. 소스(각 대상)에 대한 가장자리가 잘리지 않은 상태로 남아 있는지 여부에 따라 이러한 가장자리 중 하나만 자르는 컷을 픽셀에서 전경(각 배경)으로의 맵으로 해석할 수 있습니다. 더 나아가, 각 픽셀에 대해 이 두 가장자리 중 하나만 절단하는 절단 값은 해당 라벨링에 대한 에너지 함수 E의 값과 동일합니다.

결과적으로 최소 컷을 계산하여 에너지 함수를 최소화할 수 있습니다.
이것은 다항식인 것으로 알려져 있지만(섹션 9.4.1의 참조에서) 실제로는



에지	무게	사례
(i, j)	B(파이 , pj)	i, j, 이웃 p ∈ F p
(S → i)	케이 / 0 / DF (i)	∈ B 그 / 렇지 않 / 으면 p ∈ B
(i → 디)	케이 / 0 / 데시벨(i)	p ∈ F / 그렇지 / 않으면

그림 9.23: 왼쪽에는 전경/배경 분할을 그래프 잘라내기 문제로 설정하기 위해 이미지에서 파생된 그래프가 있습니다. 소스(S)에 연결된 픽셀을 전경 픽셀로 해석하고 드레인(D)에 연결된 픽셀을 배경 픽셀로 해석합니다. 레이블이 알려진 일부 픽셀은 둘 중 하나와 이웃에만 연결됩니다. 링크 가중치는 표에 나와 있습니다. 이웃 간의 링크는 각 방향에서 동일한 용량을 가지므로 방향 없이 그려집니다. 오른쪽에는 해당 그래프의 컷(컷된 가장자리는 회색으로 표시됨). 각 픽셀은 전경이나 배경 중 하나에 연결되어 있지만 둘 다에 연결되어 있지 않거나(그렇지 않으면 S와 D의 연결을 끊지 않았을 것이기 때문에) 둘 다에 연결되어 있지 않습니다(두 가장자리 중 하나를 복원하고 더 나은 가치). 또한 절단 모서리의 가중치 합은 에너지 비용 함수와 같습니다. 결과적으로 최소 비용 절감을 해결하여 이미지를 전경과 배경으로 분할할 수 있습니다. 표에 표시된 가중치를 사용하면 그래프의 컷 값은 에너지 함수 값과 동일합니다.

cut은 (S → i)와 (i → D)를 모두 자르지 않으며, K = 1 + maxp∈I q:{p,q}∈N B(p, q)입니다.
더 나은 절단은 하나만 절단하기 때문에 최소 절단은 둘 다 절단하지 않습니다. 이는 그래프를 잘라서 텍스트의 에너지 함수를 최소화할 수 있음을 의미합니다.

특수 알고리즘은 이제 이미지에서 그래프를 자르는 데 매우 빠릅니다.

이 절차는 섹션 6.3.2의 문제를 처리하는 한 가지 방법을 제공합니다.
여기 이미지에 구멍이 있고 구멍과 일치하는 패치가 있습니다. 그러나 패치는 일반적으로 정사각형이고 구멍은 일반적으로 그렇지 않습니다. 구멍 위에 패치를 놓습니다. 일부 픽셀의 경우 하나의 값(구멍 내부의 값과 패치 외부의 값)만 알고 있지만 다른 픽셀의 경우 두 값을 알고 있습니다. 이를 위해 최종 이미지에 나타나는 픽셀을 선택하고 싶습니다. 다시 말하지만 조합 문제가 있습니다. 최종 이미지의 i번째 픽셀이 패치에서 가져와야 하는 경우 값 -1을 취하고 그렇지 않으면 1을 취하는 변수에 대해 δi 를 작성합니다 . 레이블 중 하나를 취할 수 있는 픽셀에 대해 U를, 패치에서만 값을 취할 수 있는 픽셀에 대해 P를, 이미지에서만 값을 취할 수 있는 픽셀에 대해 I를 씁니다. 전경 또는 배경 모델이 없습니다. 일반적으로 우리는 픽셀이 이웃과 일치하는 δ를 갖기를 원합니다.

인접한 두 픽셀의 δ 값이 다른 경우(즉, 잘라낸 지점에서

재산	친화도 함수 exp (x	노트
거리	y) 티(x-y)/2σ 2 디	
강도 exp	(I(x) I(y))t (I(x) I(y))/2σ 2 나	I(x)는 x에 있는 픽셀의 강도입니다. c(x)는
색상	exp dist(c(x), c(y))2/2σ 2씨	x에 있는 픽셀의 색상입니다. f(x)는 섹
텍스처 지수 -	(f(x) - f(y))t (f(x) - f(y))/2σ 2 나	션 6.1에서와 같이 계산된 x의 픽셀을 설명하는 필터 출력의 벡터입니다.

표 9.1: 그래프 기반 분할기에 대한 픽셀을 비교하는 다양한 선호도 함수.
친화도를 결합할 수 있습니다. 지수 형식의 매력적인 특징 중 하나는 위치, 강도 및 질감 유사성을 곱하여 결합할 수 있다는 것입니다.

패치에서 이미지까지) 픽셀의 실제 값이 가능한 한 유사하기를 원합니다. 이는 이미지가 패치와 일치하는 위치에서 혼합되도록 하기 위한 것입니다. 이러한 기준은 그래프 컷으로 최소화할 수 있는 에너지 함수에 기록할 수 있습니다.

9.4.4 표준화된 컷

min-cut으로 이미지를 분할하는 것은 좋은 전경 및 배경 모델 없이는 일반적으로 제대로 작동하지 않습니다. 이는 작은 픽셀 그룹을 잘라냄으로써 아주 좋은 컷 값을 얻을 수 있기 때문입니다. 컷은 세그먼트 내 일관성과 세그먼트 간 차이의 균형을 맞추지 않습니다. Shi와 Malik(2000)은 정규화된 절단을 제안합니다. 즉, 절단 비용이 각 그룹 내 총 친화도의 작은 부분이 되도록 그래프를 두 개의 연결된 구성 요소로 절단합니다.

이를 위해서는 픽셀 간의 선호도 측정이 필요합니다. 이미지를 각 픽셀에 하나의 정점이 있고 각 픽셀에서 모든 이웃까지의 가장자리가 있는 그래프로 모델링합니다. 각 가장자리에 가중치를 두어야 하며 이를 픽셀 간 친화성이라고 합니다. 선호도 측정의 자세한 형식은 당면한 문제에 따라 다릅니다. 유사한 노드를 연결하는 호의 가중치는 커야 하고, 서로 다른 노드를 연결하는 호의 가중치는 작아야 합니다. 다를 때). 표 9.1은 현재 사용 중인 몇 가지 선호도 함수를 제공합니다.

정규화된 절단은 절단 비용이 각 그룹 내 총 친화도의 작은 부분이 되도록 그래프를 두 개의 연결된 구성 요소로 절단해야 한다는 점을 기억하십시오. 가중 그래프 V를 두 개의 구성 요소 A와 B로 분해하고 다음과 같이 분해 점수를 매기는 것으로 공식화할 수 있습니다.

$$\frac{\text{절단}(A, B)}{\text{결합}(A, V)} + \frac{\text{절단}(A, B)}{\text{결합}(B, V)}$$

(여기서 cut(A, B)는 한쪽 끝이 A이고 다른 쪽 끝이 B인 V의 모든 간선의 가중치 합이고, assoc(A, V)는 한쪽 끝이 다음인 모든 간선의 가중치 합입니다. ㅏ). 이 점수는 절단이 두 구성 요소 사이에 가중치가 낮은 모서리가 거의 없고 가중치가 높은 내부 모서리가 많은 두 구성 요소를 분리하는 경우 작습니다. 정규화된 컷이라고 하는 이 기준의 최소값을 가진 컷을 찾고 싶습니다.

이 기준은 실제로 성공적입니다(그림 9.24).

이 문제는 모든 그래프 컷을 살펴봐야 하기 때문에 이 형식으로는 해결하기가 너무 어렵습니다. 이것은 조합 최적화 문제이므로 인접한 컷이 특정 컷의 값을 얼마나 잘 받았는지 추론하기 위해 연속성 인수를 사용할 수 없습니다. 더 나쁜 것은 그리드 그래프의 경우에도 NP-완전 문제입니다.

그러나 Shi와 Malik(2000)은 좋은 컷을 생성하는 근사 알고리즘을 제공합니다.

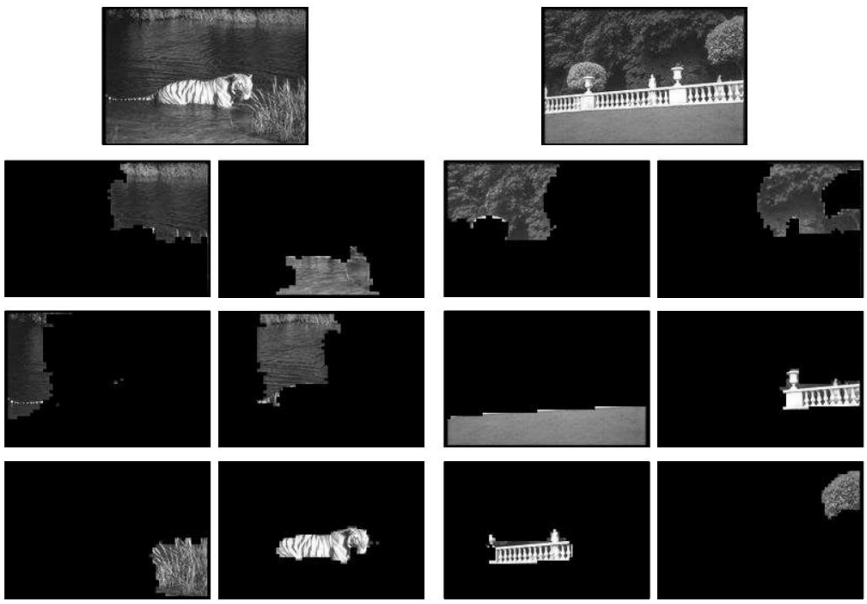


그림 9.24: 상단의 이미지는 텍스트에 설명된 정규화된 컷 프레임워크를 사용하여 표시된 구성 요소로 분할됩니다. 사용된 친화도 측정은 표 9.1에서와 같이 강도와 질감을 포함합니다. 헤엄치는 호랑이의 이미지는 본질적으로 호랑이인 한 부분, 풀인 부분, 그리고 호수에 해당하는 네 가지 구성 요소를 생성합니다. 마찬가지로 난간은 합리적으로 일관된 세 개의 세그먼트로 표시됩니다. 텍스처 측정을 통해 얻은 k-평균 세분화에 대한 개선에 주목하십시오. 이 그림은 원래 J. Shi, S. Belongie, T. Leung 및 J. Malik, Proc에 의해 "Image and video segmentation: the normalized cut framework"의 그림 2로 게시되었습니다. IEEE Int. 회의

이미지 처리, 1998 c IEEE, 1998.

9.5 실제 이미지 분할

이제 많은 중요한 이미지 분할기에서 코드를 사용할 수 있습니다. EDISON 코드(Rutgers의 컴퓨터 비전 그룹, http://coewww.rutgers.edu/에서 사용 가능)

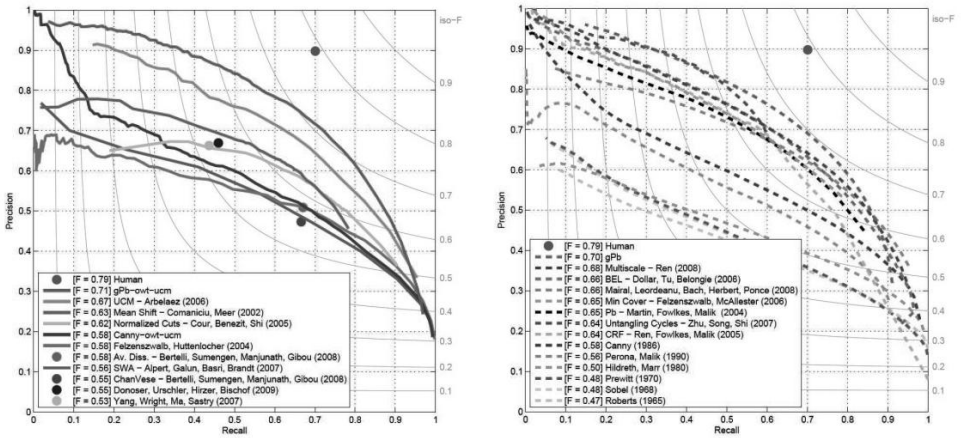


그림 9.25: 세그먼트와 에지 검출기는 예측된 경계를 사람들이 이미지에 표시하는 객체 경계와 비교하여 평가할 수 있습니다. 자연스러운 비교에는 정밀도(표시된 경계 지점 중 실제 경계 지점의 백분율) 및 재현율(표시된 실제 경계 지점의 백분율)이 포함됩니다. F 측정은 정밀도와 재현율을 단일 숫자 F = 2P R/(P + R)로 요약합니다. 왼쪽에는 다양한 세그먼터에 대한 이러한 측정값이 있습니다. 오른쪽은 다양한 에지 검출기용입니다. 이 그림은 원래 P. Arbelaez, M. Maire, C. Fowlkes 및 J. Malik, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2011, c IEEE의 "Contour Detection and Hierarchical Image Segmentation"의 그림 1 및 2로 게시되었습니다. , 2011.

riul/research/robust.html) 평균 이동 이미지 세분화를 구현합니다(섹션 9.3.5). 동일한 웹 페이지는 다양한 다른 평균 시프트 코드를 배포합니다.
Pedro Felzenszwalb는 http: //people.cs.uchicago.edu/~pff/segment/에서 그의 세그먼트(9.4.2절)용 코드를 배포합니다. Jianbo Shi는 http://www.cis.upenn.edu/~jshi/software/에서 정상화 컷용 코드를 배포합니다. Greg Mori는 http://www.cs.sfu.ca/~mori/research/superpixels/에서 normalized-cut 알고리즘을 사용하여 슈퍼픽셀을 계산하기 위한 코드를 배포합니다. Yuri Boykov는 http://vision.csd.uwo.ca/code/에서 min-cut 문제에 대한 코드를 배포합니다. 여기에는 매우 큰 그리드에 대한 코드가 포함됩니다. Vladimir Kolmogorov는 http://www.cs.ucl.ac.uk/staff/V.Kolmogorov/software.html에서 min-cut 코드를 배포합니다.

9.5.1 세그먼트 평가

세그먼터의 정량적 평가는 방법마다 목표가 다르기 때문에 다소 골치 아픈 문제입니다. 합리적인 목표 중 하나는 사람들이 이미지에 표시하는 객체 경계를 예측하는 것입니다. 이 보기는 사람들이 테스트 세트에 표시한 경계 지점에 대한 재현율 및 정밀도 측면에서 정량적 평가를 제공합니다. 자연스러운 비교는 정밀도 P(실제 경계 지점, 즉 사람이 표시한 표시된 경계 지점의 백분율) 및 리콜 R(표시된 실제 경계 지점의 백분율)을 포함합니다. F 측정은 정밀도와 재현율을 단일 숫자 F = 2P R/(P + R)로 요약합니다. 이 프레임워크에서 인간의 성과는 테스트 사람을 들고 테스트 사람의 마크업을 나머지 모든 사람과 비교한 다음 성능 통계를 평균화하여 평가할 수 있습니다.

테스트 사람들을 개최했습니다. 최신 세그먼터는 이 테스트에서 꽤 잘할 수 있지만 사람만큼은 아닙니다(그림 9.25).

Berkeley 분할 데이터 세트는 300개의 수동 분할 이미지로 구성되며 http://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/에서 배포됩니다. 이 페이지는 또한 해당 데이터 세트에 대한 최신 벤치마크를 유지합니다.
최신 버전(BSDS-500)에는 500개의 수동 분할 이미지가 있습니다. http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/resources를 참조하십시오. html. 다시 말하지만, 사용 가능한 해당 데이터 세트에 대한 일련의 벤치마크가 있습니다. Lotus Hill Institute는 http://www에서 학술용으로 무료로 사용할 수 있는 대용량 데이터 세트를 제공합니다. imageparsing.com/. 주석은 단순한 지역 구조보다 훨씬 풍부하며 지역 관계의 상세한 의미론적 계층으로 확장됩니다.

9.6 참고 사항

분할은 어려운 주제이며 매우 다양한 방법이 있습니다. 주로 역사적인 관심을 끄는 조사로는 Riseman과 Arbib(1977), Fu와 Mui(1981), Haralick과 Shapiro(1985), Nevatia(1986), Pal과 Pal(1993)이 있습니다.

한 가지 이유는 일반적으로 몇 가지 예를 보여주는 것보다 더 유용한 수준에서 세그멘터의 성능을 평가하기가 매우 어렵기 때문입니다. 원래 클러스터링 세그먼터는 Ohlander et al. (1978). 클러스터링 방법은 다소 자의적인 경향이 있습니다. 기억하세요. 이것이 유용하지 않다는 의미는 아닙니다. 클러스터링 대상과 방법을 예측하는 데 사용할 수 있는 이론이 실제로 많지 않기 때문입니다. 우리가 해야 할 일은 특정 애플리케이션에 도움이 되는 클러스터를 형성하는 것이지만 이 기준은 유용한 방식으로 공식화되지 않았습니다. 이 장에서 우리는 세부 사항을 무시하고 큰 그림을 제시하려고 시도했습니다. 수행된 작업에 대한 자세한 기록은 계몽적이지 않기 때문입니다. 모두가 응집 클러스터링, 분할 클러스터링, k-평균, 평균 이동 및 하나 이상의 그래프 기반 클러스터링 알고리즘(선택!)에 대해 알고 있어야 합니다. 이러한 아이디어는 많은 응용 프로그램에 매우 유용하기 때문입니다. 세분화는 클러스터링의 한 가지 응용 프로그램일 뿐입니다.

인간의 시각적 인식에서 그룹화의 역할에 대한 많은 문헌이 있습니다.
표준 게슈탈트 핸드북에는 Kanizsa(1979) 및 Koffka(1935)가 포함됩니다. 주관적 윤곽선은 Kanisza에 의해 처음 기술되었습니다. Kanizsa(1976)에 광범위한 요약 논의가 있습니다. Palmer(1999)의 권위 있는 책은 우리가 여기서 제공할 수 있는 것보다 훨씬 더 넓은 그림을 제공합니다. Gordon(1997)에는 다양한 시각 이론의 발전과 Gestalt 사고의 기원에 대한 많은 정보가 있습니다. 일부 그룹은 팝아웃(pop out)으로 알려진 현상인 시각적 프로세스 초기에 현저하게 형성되는 것으로 보입니다(Triesman 1982).

우리는 유역이 원래 Digabel과 Lantu´ejoul(1978)에 기인한다고 믿습니다. Vincent와 Soille(1991)을 참조하십시오. Fukunaga와 Hostetler(1975)가 처음으로 평균 이동을 설명했지만 Cheng(1995)이 작업할 때까지 대체로 무시되었습니다. 이제 컴퓨터 비전 연구의 주요 체류지입니다. 다음 장에서 볼 수 있듯이, 그것은 수많은 응용 프로그램을 가지고 있습니다.

다양한 그래프 이론 클러스터링 방법이 비전에 사용되었습니다(Sarkar and Boyer(1998), Wu and Leahy(1993) 참조; Weiss(1999)에 요약이 있음).

초고속 min-cut 덕분에 인터랙티브 분할이 가능해졌습니다.

관련 2개 레이블 Markov 랜덤 필드를 해결하는 알고리즘(Vogler et al.
(2000); 보이코프와 졸리(2001); 또는 Boykov 및 Funka Lea(2006)). 이제 많은 중요한 변형이 있습니다. Grabcut은 Rother et al. (2004); Objcut은 개체 모양에 대한 사전 정보를 사용하여 절단을 개선합니다(Kumar et al. 2010). 및 Duchenne et al. (2008). Wang과 Cohen(2007)이 자세히 조사한 수많은 매팅 방법이 있습니다.

정규화된 절단 형식주의는 Shi and Malik(1997) 및 (2000)에 기인합니다.
변형에는 모션 분할 Shi 및 Malik(1998a)에 대한 응용 프로그램과 출력 Shi 및 Malik(1998b)에서 유사성 메트릭을 추론하는 방법이 포함됩니다. 수많은 대체 기준이 있습니다(예: Cox et al.(1996), Perona 및 Freeman(1998)).

세분화의 평가에 관한 상당한 초기 문헌이 있습니다.
유용한 참고 문헌은 다음과 같습니다. Zhang(1996a); 장(1997); Beauchemin과 Thomson(1997); 장과 게르브란트(1994); 코레이아와 페레이라(2003); 레이와 우두파(2003); Warfieldet al. (2004); 팔리에로니(2004); 카르도소와 코르테 레알(2005); 카르도소와 코르테 레알(2006); Cardosoet al. (2009); Carleer et al. (2005); 및 Crum et al. (2006). 특정 작업의 맥락에서 평가가 더 쉽습니다. 다양한 작업을 다루는 문서로는 Yasnoff et al. (1977), Hartley et al. (1982), Zhang(1996b), Ranade와 Prewitt(1980). Martinet al. (2001)은 현재 평가의 표준인 Berkeley 세분화 데이터 세트를 도입했지만 사용할 수 있는 다양한 기준이 있습니다. Unnikrishnan et al. (2007) 랜드 지수 사용; Polaket al. (2009) 다중 객체 경계 사용; Polaket al. (2009)은 4가지 분할 알고리즘에 대한 자세한 평가를 제공합니다. Hanbury와 Stottinger(2008)는 메트릭을 비교합니다. and Zhang et al. (2008) 평가 방법에 대한 최근 조사를 제공합니다. 좋은 이미지 세그먼트는 내부적으로 일관성이 있을 가능성이 높지만 그 아이디어를 유용하게 만드는 것은 어렵습니다(Bagon et al. 2008).

세그멘테이션을 제대로 하기 어렵기 때문에 Russell et al. (2006)은 여러 세분화로 작업한 다음 좋은 조각을 선택하도록 제안합니다. 이 아이디어는 이제 매우 영향력이 있습니다. 지원 추정치를 개선하고(Malisiewicz 및 Efros 2007) 인식을 유도하기 위해(Pantofaru et al. 2008) 또는 (Malisiewicz 및 Efros 2008) 여러 세분화가 사용되었습니다. 여러 세그먼트를 포함 계층 구조로 구성할 수 있습니다(Tacc and Ahuja 1997). 계층 구조는 개체 모델을 생성하고(Todorovic and Ahuja 2008b) 일치시킬 수 있습니다(Todorovic and Ahuja 2008a).

우리는 주로 역사적 정확성보다는 설명에 중점을 두기 때문에 지각 조직의 일부 측면을 자세히 논의하지 않았으며 이러한 방법은 통합된 관점에서 따릅니다. 예를 들어, 우연히 발생했을 가능성이 없는 구성으로 이미지 가장자리 점 또는 선 세그먼트를 클러스터링하는 방법에 대한 긴 문헌이 있습니다. 우리는 다음 장에서 이러한 아이디어 중 일부를 다루지만 Amir와 Lindenbaum(1996), Huttenlocher와 Wayner(1992), Lowe(1985), Mohan과 Nevatia(1992), Sarkar와 Boyer(1993)에도 독자의 관심을 끕니다. , 그리고 Sarkar와 Boyer(1994)에게. 사용자 인터페이스를 구축할 때 지각적으로 두드러지는 것이 무엇인지 아는 것이 도움이 될 수 있습니다(예: Saund and Moran(1995)).

문제

9.1. 함수의 최빈값을 찾기 위한 평균 이동 절차

$$\text{에프(엑스)} = C \sum_{i=1}^{N} \left\| \text{시간} \frac{x \quad xi\ k}{} \right\|^2$$

치를 생성하는 경우 y (j) 일련의 추정

$$y^{(d+1)} = \frac{\text{나 h} \left(\frac{xi\text{-}y\ \text{xig}(\| \quad \|}{} \right)^{(j)\ 2}}{\text{나 g}(\| \frac{xi \quad y(j)}{\text{새은}} \|)^2}.$$

이제 함수가 있다고 가정합니다.

$$\text{에프(엑스)} = C \sum_{i=1}^{N} \frac{(2\pi)(\quad ds/2)}{h_{\text{ds}}^{\text{제이}}} \frac{\text{스엑스}^{\text{에}} \text{스 - 엑스나}^{\text{에}}}{hs} \frac{(2\pi)(\quad dr/2)}{\text{터}\,\text{h}^{\text{닥}}} \frac{rx \quad \text{- x 나}^{r}}{\text{시간}}.$$

이 함수에 대한 평균 이동 추정의 형식은 무엇입니까?

프로그래밍 실습

9.2. Mean Shift Segmenter를 구현합니다.

9.3. 섹션 9.4.2 이후에 그래프 기반 분할기를 구현합니다.

9.4. 섹션 9.4.3 이후에 그래프 기반 분할기를 구현합니다. 이를 위해 웹에서 사용할 수 있는 빠른 그래프 컷 패키지 중 하나를 사용해야 합니다.

9.5. 그래프 기반 분할기를 사용하여 대화형 분할 시스템을 구축하십시오.