다중 카메라를 이용한 이미지 객체의 이동 경로 추적1)

오종현*, 박호현*
*중앙대학교
e-mail:hyper_@naver.com

A tracking method for moving objects using multiple cameras

Jong-Hyun Oh*, Ho-Hyun Park*
*Chung-Ang University

요 약

본 논문에서는 인덱싱 카메라나, 감시 시스템의 네트워크로 부터 입력받은 영상들에서 사용자가 원하는 오브젝트를 추출해내고, 카메라의 위치와 시간으로 정렬되어진 데이터베이스상의 이미지들과 비교분석하여 원하는 객체와의 유사도를 측정하고, 이것을 바탕으로 동일한 오브젝트들을 가진 이미지들의 집합을 시간과 공간데이터를 이용하여 객체의 이동경로를 파악하는 시스템을 제시한다.

1. 서론

최근 컴퓨터를 통한 영상처리 기술의 발달로 다양한 분야에서 영상 시스템들이 활용되고 있다. 그 중에서도 우리주변에서 흔히 볼 수 있는 편의점이나 빌딩 입구의 출입감시, 무인감시 주차장, 엘리베이터 등에서 감시 시스템의 역할이 매우 중요하다. 초창기의 감시 시스템의 경우 관리자가 지속적으로 화면을 보며 지역이나 대상을 감시해야 했기 때문에 매우 수동적이며 비효율적이었다. 하지만 현재는 기술의 발달로 감시 영역의 움직임 검출, 특징 추출을 통한 인식, 이동 객체의 속도 검출 등 많은 부분들을 자동으로 처리 할 수 있게 되었고, 사용자는 이것을 바탕으로 컴퓨터가 처리할 수 없는 부분만을 판단하여 감시 시스템을 운영할 수 있게 되었다.

감시 시스템은 카메라로 부터 입력된 영상에서 많은 오브젝트들을 검출해 낼 수 있고, 이렇게 검출되어진 오브젝트들에 대해 특정한 분석을 하게 된다. 이때 관리자가 자신이 원하는 특정 객체를 선택하게 되고 감시 시스템이 해당 객체를 추적하는 경우를 생각해 볼 수 있다. 이때 자체적인 움직임이 가능한 카메라 한대가 표적 객체를 추적하는 경우도 있을 것이고, 혹은 다중 카메라 환경에서 각각의 영역을 여러 개의 카메라가 나누어 추적하는 경우도 생각해 볼 수 있다. 한대의 카메라가 객체를 추적하는 시스템은 감시하고자 하는 범위가 비교적 제한적이므로 넓은 범위로 활동하는 객체의 추적에 효율적이지 않다. 실제로 많은 감시 시스템의 주요 목표인 사람의 경우 그 활동영역을 한대의 카메라로 모두 포착하기에 힘들다. 반면에, 다중 카메라 환경의 경우는 이러한 범위적인 제한이 없고, 카메라의 위치에 따른 인덱싱을 통해 목표 객체의 이동경로를 쉽게 파악할 수 있는 장점이 있다. 하지만 이러한 다중 카메라 감시 시스템의 경우 각 카메라가 설치된 환경에 따라 영상의 각도나 객체로부터의 거리, 조도 등이 각각 다르게 영상에 영향을 줄 수가 있으므로 영상에 대한 추가적인 사전 보정작업이 필수가 된다. 또한, 카메라의 수가 늘어나는 만큼 많은 양의

1) 본 연구는 산학 공동기술개발지원사업으로 시행된 감시 시스템 네트워크를 이용한 객체의 이동 경로 추적의 연구지원에 의하여 수행되었음.

이미지 정보를 처리, 저장할 수 있는 인프라와 하드웨어가 바탕이 되어야 한다.

본 논문에서는 사용자가 원하는 객체를 추출하고, 추출된 객체를 템플릿으로 지정하여 다중 카메라 환경에서 촬영된 영상 데이터들을 시간과 공간(카메라의 위치)을 통해 비교, 검색함으로써 목표 객체의 이동경로를 추적하는 시스템을 구축하고자 한다.

소개되는 감시 시스템은 크게 세 부분으로 나눌 수 있다. 첫 번째는 입력 영상으로부터 사용자가 원하는 객체를 추출하는 기능으로, 사용자가 설정하는 영역과 배경을 비교해서 해당 영역 내에서 객체를 추출해내고, 추출된 객체의 크기에 맞게 이미지의 일부분을 잘라내어 템플릿을 만들어낸다. 두 번째는 추출되어진 템플릿 이미지를 다른 카메라들에서 촬영된 이미지와 비교하여 유사도를 측정하여 목표 객체를 찾아내는 과정이다. 마지막으로 목표객체가 포함된 것으로 판명된 이미지들을 모으고, 이 집합들의 시간과 공간 데이터를 비교 분석하여, 이미지를 한 번 더 걸러냄과 동시에 객체의 이동경로를 파악하여 이미지들에 대한 포인터를 리스트 형태로 저장하게 된다. 이러한 이동경로에 대한 데이터가 구축되면 사용자는 후에 특정객체가 몇 시에 어떤 경로를 통해 이동했는지를 쉽게 파악할 수 있게 된다.

2장에서는 본 연구에서 사용된 이미지 내의 객체 추출과, 템플릿 매칭, 시공간 데이터의 제어에 관한 관련 기술 및 아이디어에 대해 기술하고, 3장에서는 제안하는 다중 카메라 감시 시스템의 전체적인 흐름 및 성능 분석에 대해 기술하며, 마지막으로 4장에서는 향후 연구 방향에 대해 기술한다.

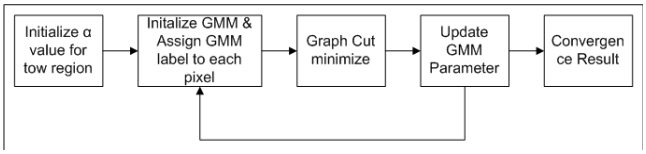
2. 알고리즘

본 장에서는 객체의 추출과 추출된 객체의 템플릿 설정을 통한 템플릿 매칭으로 유사이미지를 검출해내는 방법, 그리고 시공간 데이터에 대해 기술 한다.

2.1 GrabCut Algorithm

본 연구에서 객체 추출은 GrabCut알고리즘을 사용하였고, 그림 1과 같은 과정으로 수행된다.

첫 번째로 사용자가 설정한 사각형 영역을 Foreground 영역으로 설정하고, 나머지 부분을 Background 영역으로 성정한다. 두 번째로 두 영역에 대해 각각 GMM을 초기화한다.



[그림 1] GrabCut Algorithm Flow Diagram

세 번째로 Graph Cut 알고리즘을 수행한다. 네 번째로 GMM Parameter들을 갱신하게 되고, 마지막으로 수렴 여부를 판단하여 결과를 사용자에게 보여주거나, 다시 Graph Cut 알고리즘과 GMM Parameter의 갱신을 반복하여 수행하게 된다.

Graph Cut의 주요한 내부 동작을 살펴보면, 이미지 내의 각 픽셀들이 두 가지의 link를 가진다. 먼저 Foreground 노드와 Background 노드중 하나와 연결되어 픽셀 자신이 속한 노드를 나타내는 T-link가 있고, 자기 주변의 8개의 이웃들과 각각 가중치를 가진 채 연결되어 있는 N-link가 있다. Graph Cut은 이 N-link의 값이 매우 높은 곳은 Gradient가 낮은 값으로 판별하고, 반대로 이 값이 낮은 곳을 Gradient가 높은 영역 즉, 경계선으로 판별하게 된다. 그 후, T-link를 Foreground나 Background 노드로 연결하는데, 이것은 확률로 계산되어지고, 이 확률은 GMM에 포함되어 있다. GrabCut 알고리즘에 의해 GMM의 갱신과 Graph Cut이 반복 수행됨에 따라, T-link의 가중치가 변하게 되므로, 픽셀이 포함되는 영역이 Foreground나 Background로 바뀔 수 도 있게 되는 것이다.

2.2 Template Matching Algorithm

본 연구에서 두 이미지간의 비교단계 전에 입력 이미지로부터 사용자가 원하는 객체를 GrabCut을 통해 추출하기 때문에 이를 활용하여 추출된 객체를 템플릿으로 활용하면 보다 정확도가 높아지고 연산을 해야 하는 픽셀이 범위가 줄어든다. 또한 객체의 Scale에도 어느 정도 독립성이 있어서 우리가 가정한 감시영상 환경 하에서 이미지간의 상관관계를 추출하기에 좀 더 적합하다. 따라서 우리는 추출된 이미지와 검색되어지는 이미지간의 상관관계를 구하여 유사도 및 해당 객체의 유무를 판별하는 알고리즘으로 Template Matching을 선택하였다.

실험은 OpenCV의 cvMatchTemplate()함수를 사용하였다. 이 함수는 유클리디안 거리를 사용하여 소스 이미지 전체를 탐색하며 템플릿 이미지와 얼마나

유사한지를 측정하고 소스와 템플릿의 상관계수 맵(Coefficient Map)을 구한다, 여기서 최대의 유사도를 가지는 값이 매칭 되는 영역이 된다. Template Matching 알고리즘은 간단하며 위에 기술한 장점들이 있지만 소스이미지 전역을 탐색해야하므로 이미지의 해상도가 큰 경우 부적합하며 템플릿과 전혀 관계없는 영상의 경우도 유사도의 최대 점에 의해 매칭 되는 영역이 발생하는 단점이 있다. 따라서 본 연구에서는 이 부분을, 매칭 되는 영역의 크기 비교를 통해 성능을 보완하여 사용하였다.

2.3 시공간 데이터의 제어

본 연구에서 시간 데이터는 인덱싱 부분과, 추출과 매칭을 통해 객체의 전체 이동경로가 파악된 후에 경로의 실제적인 보정에 사용된다. 예를 들어 본 연구의 주된 감시 대상인 사람의 경우 같은 시간에 여러 지역에 등장할 수 없으며 불가능한 시간의 차이로 구역을 이동할 수도 없다. 따라서 우리는 이점을 이용하여 검색된 이미지들로부터 추출된 경로에서 불가능한 부분을 제거해낼 수 있게 된다. 또한 검색된 이미지들을 저장할 때 시간데이터에 의해 인덱싱을 하여 저장하게 되면, 사용자가 후에 객체의 경로를 검색할 때 원하는 시간의 객체들의 경로를 검색해 볼 수 있게 된다. 공간 데이터의 경우도 마찬가지로, 사용자가 선택한 공간을 지나간 객체들을 검색할 때 질의 자체를 해당 공간으로 할 수 있으므로 인덱싱을 통해 저장하면 매우 편리하게 객체의 이동경로들을 파악할 수 있게 된다.

3. 구현 결과

본 연구는 CxImage, OpenCV, Torch3, GNU Scientific Library를 사용하여 Visual C++ 6.0으로 구현된 소프트웨어와, SAMSUNG SDC-415S 카메라와 TRUEN의 TCS-200 VIDEO SEVER SYSTEM을 이용하여 카메라로 부터 영상을 입력받아 진행하였다. 이미지의 크기는 352 * 240이다.

그림 2(a)는 객체를 추출할 원본영상이며, 그림 2(b)는 비교할 후영상이다.

그림 3(b)은 원본 영상에서 GrabCut을 통해 사용자가 원본 이미지에서 원하는 객체를 추출한 것이다.

[그림 2] (a)원본영상, (b)비교할 후 영상

GrabCut은 Peng Wang의 "GrabCut" – Interactive Foreground Extraction 논문을 기반으로 오픈소스인 CxImage 라이브러리를 통해서 다시 구현하였다.

Foreground/BackGround Brush를 통해 추출된 객체를 좀 더 보완하였고 그림 3(c)에서 푸른색이 Background brush, 노란색이 Foreground brush로 수정한 부분을 나타내고 있다.

[그림 3] (a)원본영상, (b)추출된 객체, (c)추출영역 보정

그림 4(a)는 추출된 객체에 맞게 템플릿 사이즈를 축소시켜 템플릿을 생성한 것이고, 이를 두 후영상과 비교하여 템플릿 매칭한 결과를 표시한 것이다.

[그림 4] (a)추출된 템플릿, (b)후 영상과 매칭

이러한 매칭을 인덱싱된 각각의 카메라들의 데이터베이스에 저장된 다른 이미지들과 비교를 하여 유사도가 검출된 영상들을 바탕으로 인덱싱된 카메라의 고유 번호를 알아내어 객체가 해당 영역을 지났는지를 파악하게 된다.

[그림 5] (a) 타깃의 이동경로, (b)시간데이터를 통한 경로 보정

그림 5(a)는 계산되어진 유사도를 바탕으로 인덱싱된 카메라 영역에서 템플릿과 유사한 이미지가 검출된 지역의 경로를 표시한 화면이다. 그림 5(b)는 원본영상의 시간을 기준으로 각각의 이미지들의 시간데이터를 이용하여 불가능한 지역을 제거하여 영역이 수정된 부분이 표시된 화면이다.

4. 결론 및 향후 진행방향

본 연구에서는 객체를 추출하고, 추출된 객체를 템플릿으로 설정하여 타 이미지들과의 유사도를 기반으로 객체의 이동경로를 만들어내며, 시공간 데이터를 활용하여 그 경로의 정확도를 높이는 방법을 제안하였다. 결과적으로 우리는 제한적인 영역의 카메라들로 부터 간단한 데이터들을 이용해 경로를 파악해 낼 수 있었다. 그러나 향후에는 알고리즘의 속도 개선과, 특징 추출과 같은 알고리즘을 추가적으로 접목시켜 템플릿 매칭이 가지는 기본적인 성능적인 문제를 보완하여 비슷한 색상의 객체들 간의 구별이나, 사람 개개인의 파악과 같은, 보다 정확한 객체의 판별에 대해 연구할 것이다.

참고문헌

[1] P.Wang, “GrabCut - An interactive foreground extraction tool”, A project for class CS766 - Computer Vision, 2006

[2] C Rother, V Kolmogorov, A Blake, "GrabCut: interative foreground extraction using iteratied graph cuts", ACM Transactions on Graphics, 2004

[3] Y Boycov, M jolly, "Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images", IEEE Int. Conf. on computer vision, 2001

[4] Sudipta N Sinha, "Graph Cut Algorithms in Vision, Graphics and Machine Learning", Integrative Paper, November, 2004

[5] Yuri Boykov, Vladimir Kolmogorov, “An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision", In IEEE transactions on Pattern Analysis and Machine Intelligence, 2004

[6] Ross Kindermann, J. Laurie Snell, "Markov Random Fields and Their Applications", 2000

[7] V Kolmogorov, R Zabih, "What energy functions can be minimized via graph cuts?", ECCV, 2002