움직임 분석 및 배경 영상 갱신을 이용한 바둑 기보 저장

*김민성, **이윤구
광운대학교
*Minseong0124@kw.ac.kr, **yglee96@kw.ac.kr

Recognition of Go Game positions using Motion Analysis and Background Update

*Min Seong Kim, **Yun Gu Lee
Department of Computer Science, Kwangwoon University

요약

본 논문에서는 바둑 대국 동영상에서 배경 영상과의 차이를 이용하여 바둑판 내에서의 움직임을 분석하고, 분석 결과를 이용하여 바둑돌의 착수 위치 및 바둑돌의 종류를 인식하는 자동 바둑 기보 저장 알고리즘을 제안한다. 카메라의 내부 특성이 변하지 않고 렌즈 왜곡이 존재하지 않는다고 가정하였을 때, 바둑판 위에 움직임이 없는 배경 영상과 현재의 영상 간의 차이의 변화량을 블록 단위로 누적한 블록 단위 움직임 맵(Block Motion Map)을 기반으로 움직임의 존재 여부를 판단하고, 착수 후 물체의 움직임이 없어진 영상을 배경 영상으로 갱신하며, 해당 영상과 이전 배경 영상의 패치(Patch)를 이용하여 착수 위치 및 바둑돌의 종류를 인식한다.

1. 서론

바둑 대국에서 착수 순서, 위치 등을 기록해 놓은 것을 기보라고 한다. 바둑 프로기사 간의 대국이나 대회에서 진행되는 대국의 기보는 기록원 혹은 바둑 기사가 수기로 기록하게 되며, 온라인상에서 행해지는 대국은 바둑 프로그램이 자동으로 기보를 저장한다. 이렇게 기록된 기보는 일반인들이나 바둑 기사가 복기를 통하여 바둑을 공부하는데 사용된다. 하지만 오프라인 상에서 행해지는 일반인들의 대국은 기록원이 없어 대국 기보를 저장하기 위해서는 어려움이 있을 수 있다. 따라서 촬영한 대국 동영상이나 정지영상을 기반으로 기보를 자동으로 저장하는 알고리즘 및 기법에 대한 연구가 진행되어 왔다.

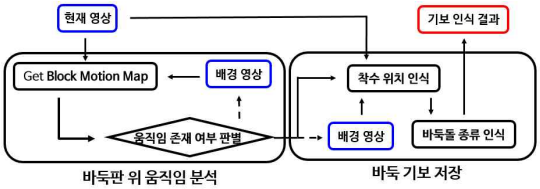
기존의 바둑 기보 자동 저장 관련 연구에서는 주로 바둑판 위의 선 정보를 기반으로 하거나[1] 바둑돌의 형태를 기반으로 원 형태를 검출하여 기보를 저장[2]하는 방식 등이 연구되어 왔다. 선 정보 기반의 기보 저장 관련 연구[1]에서는 영상에서 바둑판 위의 검은 가로 및 세로 선을 허프 변환[3]을 통하여 구하고, 선 정보를 기반으로 교차점의 위치를 구해낸 후 교차점 근처의 픽셀 영역에서 다양한 평가 값을 계산하여 해당 값을 기반으로 새로운 기보를 저장하였다. 원형 검출 기반의 바둑 기보 저장 관련 연구 [1]에서는 바둑돌의 형태가 원형이라는 점에 착안하여 이전 영상과 현재 영상의 차 영상에서 원형 허프 변환[4]을 이용하여 기보를 저장하였다. 하지만 기존의 연구들은 동영상을 기반으로 한 연구가 아닌 영상 시퀀스를 이용하였기 때문에, 영상 시퀀스 내에 손이나 물체의 움직임이 바둑판 위에 존재하는 경우 움직임으로 인해 가려진 부분의 인식이 불가하고, 원형 및 선 허프 변환은 연산량이 많아 실시간으로 대국의 기보를 저장하여야 하는 환경에 적용할 수 없다.

본 논문에서는 촬영하는 카메라의 내부적 특징 및 위치가 고정되어 있고 렌즈로 인한 왜곡이 발생하지 않는다고 가정하였을 때, 바둑판 위의 움직임 분석을 이용하여 기보를 자동으로 저장하는 방법을 제안한다. 대국 촬영 과정에서 손 및 물체의 움직임이 바둑판 위에 존재하는 지를 배경 영상과 현재 영상을 통해 분석하고 움직임이 발생하였다가 없어진 시점을 확인하여 해당 영상을 배경 영상으로 갱신한다. 또한 갱신하려는 영상과 이전 배경 영상의 패치[5]를 비교하여 기보를 저장하는 알고리즘을 제안한다.

본 논문의 구성은 다음과 같다. 2절에서는 제안하는 알고리즘의 전체적인 구성 및 과정에 대하여 설명한다. 3절에서 실제 대국 촬영 영상을 통하여 알고리즘을 구현 및 실험한 결과에 대하여 서술하고 4절에서 결론을 맺는다.

2. 알고리즘 구성

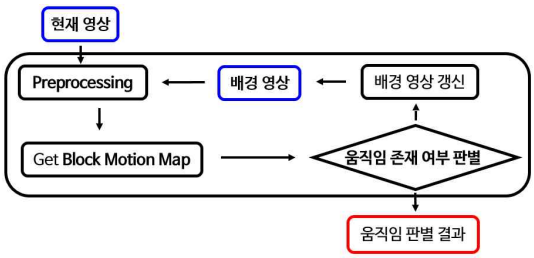
본 논문에서 제안하는 알고리즘은 크게 두 개의 과정으로 구성되어 있다. 먼저, 촬영 영상 내에서 배경 영상을 이용하여 바둑판의 움직임을 분석하여 움직임이 발생하였다가 없어진 시점의 영상을 추출해 내는 움직임 분석 과정과 새로 갱신하려는 영상을 배경 영상과 비교하여 바둑돌 착수 위치 및 바둑돌 종류를 인식하여 기보로 저장하는 기보 저장 과정으로 나뉘진다. 전체 알고리즘 구성은 <그림 2>와 같다.



<그림 2> 제안 알고리즘

2.1. 움직임 분석

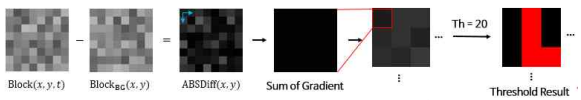
움직임 분석 및 배경 영상 갱신 과정의 알고리즘은 <그림 3>과 같다.



<그림 3> 움직임 판별 알고리즘

전처리 단계에서는 영상 내의 잡음을 제거하기 위하여 11x11 크기의 평균 필터를 적용하고 회색조 영상으로 변환하였다. 또한 바둑판 위에서의 움직임만을 분석하기 위하여 바둑판 위의 네 점을 이용하여 Perspective Transform을 수행하여 바둑판 전체를 정사각형 모양으로 변형한다. 바둑판 위의 네 점은 선 허프 변환[3]을 기반으로 네 점의 위치를 추출하는 알고리즘을 이용하여 구하였고, 바둑판의 실제 3D 좌표는 바둑판의 좌측 하단 꼭지점을 원점으로 하였고, 42.1cm x 39.4cm 크기를 바둑판의 크기로 정하였다.

바둑판 위의 움직임 분석을 위하여, 가로와 세로로 각 8개의 픽셀 집합인 Block으로 구성된 블록 단위 움직임 맵(Block Motion Map)을 설정하였다. Block Motion Map을 구하는 과정은 <그림 4>와 같다.

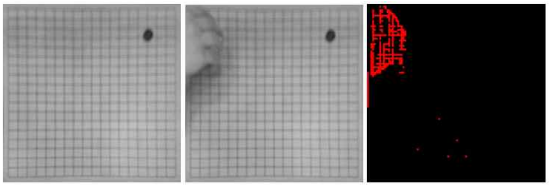


<그림 4> Block Motion Map 계산 과정

먼저 현재 입력 영상과 배경 영상을 각각 선처리한 후 두 영상의 차 영상을 계산한다. 그 후 차 영상에서 각 Block마다 x, y 방향의 변화량의 누적 값을 계산한다. 픽셀 값의 차이만을 이용하는 경우, 물체의 움직임이나 주변에서의 움직임으로 인해 발생하는 그림자에 강인한 특성을 가질 수 있다. 변화량 누적 값을 계산하는 수식은 다음과 같다.

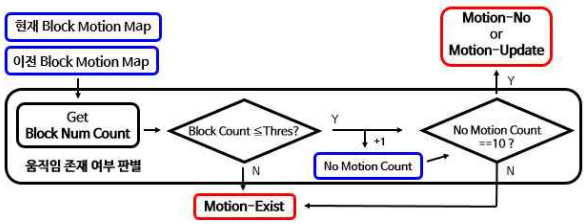
$$f_d(x,y) = \{f(x,y) - f(x,y+1)\} + \{f(x,y) - f(x+1,y)\} \cdots ①$$

변화량 누적 값을 각 픽셀별로 구한 후, Block 단위로 이 값의 합을 구하고, 문턱 값을 설정하여 Thresholding 연산을 실행한다. 문턱 값을 넘는 Block은 현재 움직임이 발생한 영역에 포함되어 있는 Block으로 판단한다. 문턱 값은 실험적으로 20으로 설정하였다. Block Motion Map을 계산한 결과는 <그림 5>와 같다.



<그림 5> Block Motion Map 계산 예시
(a) 배경 영상 (b) 현재 입력 영상
(c) Block Motion Map 계산 결과

계산된 Block Motion Map을 이용하여 바둑판 위에 물체의 움직임이 존재하는지를 판별한다. 움직임 판별 알고리즘은 <그림 6>과 같다.



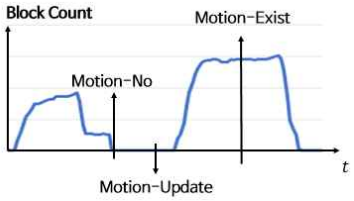
<그림 6> 움직임 상태 결정 알고리즘

영상 내의 움직임 상태는 총 3가지의 상태로 설정하였다. 먼저, 'Motion-Exist' 상태는 바둑판 내에 움직임이 존재하는 상태로, 영상 내에서 물체의 움직임이 존재하는 경우를 의미한다. 이 상태에서는 아무런 동작도 하지 않고 다음 영상으로 움직임 판별 과정을 진행한다. 'Motion-No' 상태는 바둑판 내에 움직임이 존재하지 않는 상태로, 영상 내에 착수를 위한 손의 움직임 혹은 물체의 움직임이 발생하였다가 없어진 이후를 의미한다. 이 상태에서는 배경 영상을 현재의 입력 영상으로 갱신하고 현재 프레임에서 착수 위치 및 돌의 종류를 인식하는 기보 저장 과정을 진행한다. 'Motion-Update' 상태는 바둑판 내에 움직임이 지속적으로 존재하지 않는 상태로, 움직임이 지속적으로 존재하지 않을 때에 기보 저장 과정을 진행하는 것을 방지하기 위한 상태이다.

영상 내의 움직임 상태를 판별하는 기준은 현재 영상과 이전 영상에서 계산한 Block Motion Map내부의 Block의 상태 변화를 이용한다. 만약 이전과 현재의 Block의 상태가 변화하는 경우, 움직임의 상태가 변화한 것이기 때문에 움직임이 발생하는 영역으로 설정하여야 한다. 본 논문에서는 움직임 존재 여부를 Block Motion Map 내에 존재하는 Block 중 현재와 이전의 Block의 상태가 다른 Block의 개수를 세어, 일정 개수 이상이거나 특정 조건인 경우 움직임이 있는 것으로 판별하였다.

Block 개수에 따른 상태 설정 예시는 <그림 7>과 같다. 만약 조건을 만족하는 Block의 개수가 설정한 문턱 값 이상인 경우, 현재 프레임에 움직임이 존재하는 것으로 판별하고 움직임 상태를 'Motion-Exist' 상태로 설정하고, Block의 개수가 문턱 값 이하이고, 이 상태가 일정

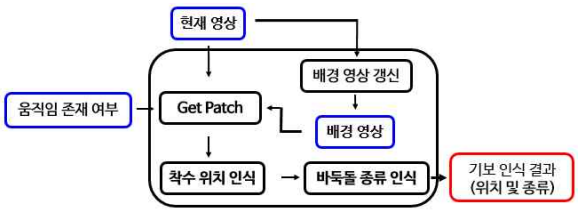
시간동안 유지되었을 경우 'Motion-No' 상태로 설정한다. 만약 'Motion-No' 상태 설정 후 일정 시간 이후에도 Block의 개수가 문턱 값 이하인 경우 'Motion-Update' 상태를 설정한다.



<그림 7> Block 개수의 변화에 따른 각 움직임 상태 설정

2.2. 바둑 기보 저장

움직임 판별 과정에서 'Motion-No' 상태로 설정되었을 경우 현재 영상과 배경 영상을 기반으로 착수 위치 및 바둑돌 종류를 인식하여 기보의 형태로 저장하는 기보 저장 과정을 진행한다. 기보 저장 과정의 알고리즘은 <그림 8> 과 같다.



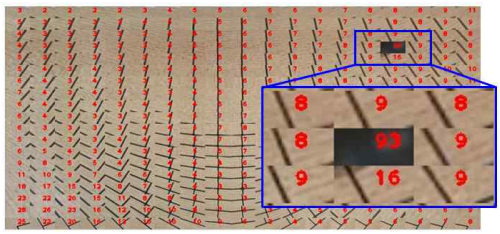
<그림 8> 바둑 기보 저장 알고리즘

기보 저장 과정은 각 착수 예상 위치에서 일정 직사각형 크기로 설정된 패치[5]를 이용하여 진행된다. 패치의 위치 및 크기는 카메라 캘리브레이션을 통하여 계산된 카메라 위치를 이용하여 카메라와 각 착수 예상 위치가 이루는 각도, 거리 및 착수 시 돌의 2D 영상 좌표 등을 종합하여 계산된다. 또한, 패치 위치 계산은 비교적 연산량이 많기 때문에,, 카메라 캘리브레이션 시 사용되는 내부 파라미터(Intrinsic Parameter) 및 외부 파라미터(Extrinsic Parameter)는 촬영 중 변하지 않는다고 가정하고 초기화 단계에서 그 위치를 미리 계산한다.. 패치 위치 계산 결과 예시는 <그림 9>와 같다.

<그림 9> 패치 위치 계산 결과 예시

착수 위치 인식 및 바둑돌 종류 판별은 위 과정에서 계산된 패치 내부 픽셀의 변화량을 기반으로 진행된다. 착수 위치 인식에서는 먼저 배경 영상과 현재 영상을 각각 평균 필터를 통과시켜 잡음을 일정 부분 제거한 후 회색조 영상으로 변환한다. 이후 각 영상의 착수 예상 위치에서의 패치의 MAD(Mean of Absolute Difference) 값을 통하여 MAD 가 설정한 문턱 값 이상인 경우 착수되었다고 판단한다. 패치의 가로 및 세로 길이가 각각 M, N 일 때 현재 영상의 패치 $f_p(x,y,t)$와 배경 영상에서의 패치 $f_p(x,y,t_{BG})$ 를 이용한 MAD 값 계산식은 다음과 같다.

$$\mathrm{MAD}(x,y) = \frac{\sum_{x,y}^{M,N} \left| f_p(x,y,t) - f_p(x,y,t_{BG}) \right|}{M \times N} \quad \cdots \ ②$$



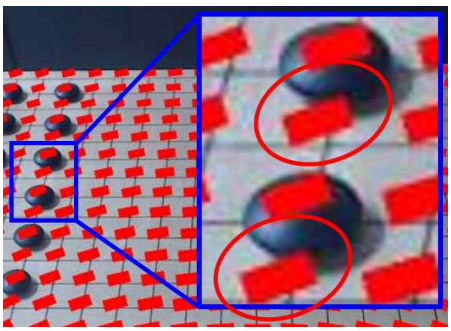
(b)

<그림 10> 각 패치별 MAD 계산 결과 예시

<그림 10>은 현재 영상과 배경 영상의 패치를 이용하여 각 패치별 MAD값을 계산한 결과이다. 바둑돌이 놓인 부분의 패치는 MAD 값이 주변에 비하여 큰 값을 가지기 때문에, 문턱 값을 이용하여 일정 값 이상인 패치의 위치를 착수 위치로 저장한다.

바둑돌 종류 인식 과정에서는 회색조로 변환된 현재 영상과 배경 영상의 착수 위치의 패치 패치의 차이의 누적을 계산하여 양수인 경우 흰 돌, 음수인 경우 검은 돌로 인식한다. 돌을 제거하는 경우도 감지하기 위하여, 기보를 저장할 때마다 각 패치의 현재 착수 상태를 저장해 두고, 착수 위치로 저장된 곳이 이미 돌이 착수된 곳이라면 돌이 제거된 것으로 인식하고 착수 상태를 돌이 없는 곳으로 변경한다.

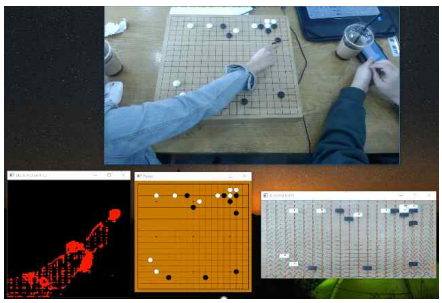
만약 바둑판을 촬영하는 각도가 비교적 낮다면, 착수된 돌의 그림자로 인하여 아래쪽의 패치 영역을 침범하는 문제가 발생한다. 아래 패치 영역을 침범하는 상황은 <그림 11>과 같다. 이와 같은 문제를 해결하기 위하여, 착수 위치로 인식된 부분을 검사하여 만약 아래의 착수 예상 위치도 착수 위치로 인식되었다면 해당 위치를 제거하는 작업을 수행한다. 또한, 물체의 강한 그림자로 인하여 발생하는 착수 위치 인식 오류를 방지하기 위하여 MAD 값이 가장 큰 2개의 착수 위치만을 기보로 저장한다. 이 과정에서 가장 최근에 저장된 바둑돌의 종류를 이용한다. 만약 저장하려는 기보의 바둑돌 종류가 최근에 저장된 것과 같다면, 기보 저장에서 제외시킨다.



<그림 11> 패치 영역 침범 예시

3. 실험 결과

제안하는 알고리즘을 실험하기 위하여, Visual C++ 및 영상처리 관련 오픈 소스 라이브러리인 OpenCV를 이용하여 바둑 기보 자동 저장 프로그램을 구현하였다. 구현 프로그램은 컴퓨터에 연결된 웹캠 혹은 스마트폰으로 촬영 된 대국 동영상을 입력으로 받아 해당 대국의 기보를 저장하는 프로그램이다. 프로그램 실행 예제는 <그림 12>와 같다. 프로그램 실행 시 현재 프레임, Block Motion Map, 모든 패치 및 기보 저장 결과가 영상으로 출력된다.



<그림 12> 구현 프로그램 실행 예제

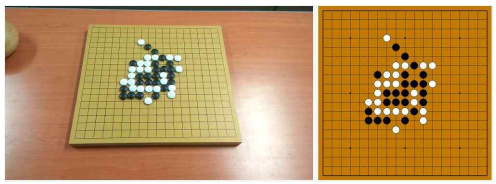
위의 프로그램을 이용하여, 실내에서 스마트폰 혹은 웹캠으로 촬영된 대국 동영상을 입력으로 하여 실제 기보와 자동으로 저장된 기보가 일치하는 비율을 퍼센트의 형태로 계산하였다. Video_00 ~ Video_03은 촬영 시 자동 초점 조절 및 조도 조절 기능을 사용하지 않았으며 실내에서 촬영하였다. 또한, 바둑판과 카메라가 이루는 각도가 40도 이상이 되도록 설정하였다. Video_04 ~ Video09는 일반적인 환경에서 촬영된 대국 동영상이다.

Test Set	총 착수 개수	정확히 인식한 개수	Percentage (%)
Video_00	31	31	100
Video_01	46	46	100
Video_02	33	33	100
Video_03	37	37	100
Video_04	33	32	96.97
Video_05	52	51	98.07
Video_06	25	25	100
Video_07	117	111	94.87
Video_08	166	158	95.18
Video_09	178	171	96.06

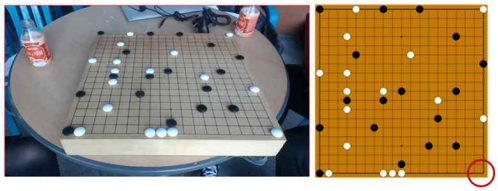
<표 1> 구현 프로그램 실행 예제

모든 테스트 영상의 실험 결과가 90% 이상의 정답률을 보였고, 환경을 제어한 상황에서 촬영한 4개의 테스트 동영상은 모든 기보를 정상적으로 인식하였다. Video_05~Video 09 실험 영상에서의 오답 상황은 다음과 같은 상황에서 발생하였다.

- 바둑돌의 착수 순서가 비정상적인 경우, 해당 착수 위치 인식 결과는 저장하지 않는다. 해당 문제의 예시는 <그림 13> - (b)와 같다.
- 착수를 위한 움직임이 연속적으로 존재하는 상황으로 인하여 3개 이상의 돌을 동시에 인식하여야 하는 경우 MAD 값이 큰 2개의 위치를 제외한 모든 위치는 기보로 저장하지 않는다.
- 집을 처리하는 알고리즘의 부재로 인하여 돌이 제거된 곳을 바둑판이 아닌 흰 돌 또는 검은 돌로 인식하는 문제가 발생한다.



(a)



(b)

<그림 13> 테스트 결과 예시
(a) Video_01의 테스트 결과 (b) Video_04의 테스트 결과

4. 결론

본 논문에서는 대국 촬영 과정에서 배경 영상과 현재 영상을 이용하여 움직임을 분석하고, 패치 영역을 이용하여 바둑돌 착수 위치 및 종류를 인식하는 알고리즘을 제안하였다. 향후에는 그림자로 인한 패치 영역 침범 문제를 해결하기 위하여 패치의 위치를 보정하거나 기보를 저장하기 위한 다른 방법 등을 적용하여 정확도를 향상시키는 연구를 진행해야 할 것이다.

Acknowledgement

본 연구는 미래창조과학부 및 정보통신기술진흥센터의 디지털콘텐츠 원천기술개발사업의 일환으로 수행하였음. [R0115-15-1012, 사용자 참여가 가능한 바둑 방송 및 기보 서비스 개발과 컨텐츠 생태계 조성을 위한 바둑 컨텐츠 마켓 플랫폼 개발]

5. 참조 문헌

[1] Corsolini, Mario, and Andrea Carta. "A New Approach to an Old Problem: The Reconstruction of a Go Game through a Series of Photographs." arXiv preprint arXiv:1508.03269 (2015).

[2] Change, Illumination, and Stone Dislocation. "조도변화와 바둑돌 미세위치변화를 고려한 CHT 기반의 자동 바둑 기보 시스템." 정보과학회논문지: 소프트웨어 및 응용 41.6 (2014).

[3] Duda, Richard O., and Peter E. Hart. "Use of the Hough transformation to detect lines and curves in pictures." Communications of the ACM 15.1 (1972): 11-15.

[4] Ballard, Dana H. "Generalizing the Hough transform to detect arbitrary shapes." Pattern recognition 13.2 (1981): 111-122.

[5] 이대규, and 이윤구. "자동 바둑 기보 저장을 위한 바둑돌 인식 알고리즘." 대한전자공학회 학술대회 (2016): 809-812.