특집논문 (Special Paper)
방송공학회논문지 제22권 제6호, 2017년 11월 (JBE Vol. 22, No. 6, November 2017)
https://doi.org/10.5909/JBE.2017.22.6.724
ISSN 2287-9137 (Online) ISSN 1226-7953 (Print)

방해물 분석 및 배경 영상 갱신을 이용한 바둑 기보 기록

김 민 성[a], 윤 여 경[a], 이 광 진[a], 이 윤 구[a]‡

Recognition of Go Game Positions using Obstacle Analysis and Background Update

Min-Seong Kim[a], Yeo-Kyung Yoon[a], Kwang-Jin Rhee[a], and Yun-Gu Lee[a]‡

요 약

바둑 기보를 자동으로 기록하는 기존의 방법들은 대국 중 발생하는 방해물(손 혹은 물체)의 바둑판 가림 현상을 제대로 고려하지 않았다. 방해물에 의해 바둑판이 가려지는 경우 바둑돌의 착수 위치를 인식하지 못하거나, 바둑돌의 착수 순서가 실제와 다르게 저장되는 문제가 발생할 수 있다. 제안된 알고리즘은 방해물이 없는 온전한 바둑판 영상만을 배경 영상으로 내부에 저장하고 배경 영상과 현재 입력 영상을 비교하여 방해물을 인식한다. 그림자가 방해물로 오인식되는 현상을 제거하기 위해 단순한 차 영상이 아닌 미분영상을 기반으로 한 방해물 검출 방법이 제안되었다. 추가로 노이즈에 강인하게 방해물을 인식하기 위한 노이즈 제거 방법도 제안되었다. 방해물이 없는 때는 배경 영상을 지속적으로 갱신한다. 최종적으로 각 순간마다 저장된 배경 영상들을 비교하여 바둑돌의 착수 위치와 바둑돌의 종류를 인식한다. 실험 결과에 따르면 일반적인 조명환경에서 제안된 알고리즘은 95%이상의 인식률을 보여준다.

Abstract

Conventional methods of automatically recording Go game positions do not properly consider obstacles (hand or object) on a Go board during the Go game. If the Go board is blocked by obstacles, the position of a Go stone may not be correctly recognized, or the sequences of moves may be stored differently from the actual one. In the proposed algorithm, only the complete Go board image without obstacles is stored as a background image and the obstacle is recognized by comparing the background image with the current input image. To eliminate the phenomenon that the shadow is mistaken as obstacles, this paper proposes the new obstacle detection method based on the gradient image instead of the simple differential image. When there is no obstacle on the Go board, the background image is updated. Finally, the successive background images are compared to recognize the position and type of the Go stone. Experimental results show that the proposed algorithm has more than 95% recognition rate in general illumination environment.

Keyword : Records of Go games, Background update, Obstacle analysis

Ⅰ. 서 론

바둑 대국에서 바둑 기사, 승리 정보, 착수 순서 및 위치 등 바둑 대국의 전반적인 정보 및 경기 결과를 기록해 놓은 것을 '기보'라고 한다. 저장된 기보는 바둑 기사 혹은 일반인들이 바둑을 학습하는 용도로 사용된다. 온라인에서 바둑 대국 프로그램을 이용하여 진행되는 대국의 경우 해당 프로그램에서 자동으로 기보를 저장하고 본인이 진행한 대국이나 프로기사 간의 대국 기보를 확인하고 공부할 수 있도록 지원하고 있다. 오프라인으로 진행되는 바둑 대회 혹은 프로 바둑기사 간의 대국의 경우 기록원이나 동료 바둑기사가 수기 혹은 컴퓨터 등으로 기보를 저장하고 있다. 하지만 일반인들의 대국의 경우 본인이 직접 기록하거나 다른 사람에게 부탁하는 식으로 기보를 저장해야하기 때문에 기보를 기록하는데 있어서 불편함이 존재한다. 따라서 촬영된 대국 영상을 이용하여 자동으로 기보를 저장하기 위한 연구가 진행되어 왔다.

기존의 바둑 기보 자동 저장 관련 연구에서는 주로 바둑판 위의 선 정보를 기반[1]으로 하거나 바둑돌의 형태인 원을 검출하여 기보를 저장[2]하는 방식이 연구되어 왔다. 선 검출 기반의 기존 연구[1]에서는 바둑판 위의 검은 선 및 교차점을 구하고, 교차점 근처 픽셀 집합의 특성을 이용하여 기보를 자동으로 저장하는 방식을 제안하였다. 원 검출 기반[2]의 기존 연구에서는 바둑돌의 형태가 원형이라는 점을 이용하여 이전 영상과 현재 영상간의 차 영상을 구하고 차 영상에서 허프 원 변환[3]을 이용하여 바둑돌의 착수 위치 및 바둑돌의 종류를 인식하는 방식을 제안하였다. 또한, 카메라 캘리브레이션을 이용하여 계산한 패치 내부의 픽셀 값을 비교하여 기보를 저장하는 방법에 관련된 연구[4]도 진행되었다. 하지만 기존의 연구에서 이용하는 허프 선 변환 혹은 원 변환은 연산량이 많아 실시간으로 대국의 기보를 저장하여야 하는 환경에 적용하기 힘든 단점이 존재한다. 또한, 동영상 기반이 아닌 영상 시퀀스를 이용하였기 때문에 영상 내에 손이나 물체의 움직임이 바둑판 위에 존재하는 경우 방해물로 인하여 가려진 부분의 인식이 불가능하다. 따라서 동영상이나 실시간 환경에서 기보를 자동으로 저장하기 위해서는 비교적 간단한 연산을 이용하여 착수 위치 및 종류를 인식하고, 영상 내의 방해물의 존재 여부 확인을 통하여 영상 내에 방해물이 없는 상태일 때 기보를 저장해야 할 필요성이 존재한다.

본 논문에서는 촬영하는 카메라의 내부적 특징이 변화하지 않고 렌즈로 인한 왜곡이 발생하지 않는다고 가정하였을 때, 바둑 대국 촬영 시 일반적인 환경에서는 카메라가 고정되어 있다는 점을 이용해 바둑판 촬영 영상 중 움직임이 없을 것으로 예상되는 첫 번째 프레임을 배경 영상으로 설정하고 배경 영상과의 차이를 이용하여 영상 내의 방해물의 존재 여부를 분석하였다. 또한 방해물이 없는 시점을 분석을 통하여 확인하고 해당 영상을 추출 후 새로운 배경 영상으로 갱신한다. 또한 갱신하려는 영상과 이전 배경 영상의 패치를 비교하여 바둑돌 착수 위치 및 종류를 인식하는 기법을 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 기존의 기보 자동 저장 관련 연구 및 배경 영상을 이용한 움직임 영역 추출에 대하여 설명하고, 3장에서는 본 논문에서 제안하는 방법에 대한 필요성 및 문제 정의에 관련하여 서술한다. 4장에서는 제안하는 알고리즘의 전체적인 구성 및 과정에 대하여 설명한다. 5장에서 실제 대국 촬영 영상을 통하여 알고리즘을 구현 및 실험한 결과에 대하여 서술하고 6장에서 결론을 맺는다.

Ⅱ. 관련 연구

바둑 기보 자동 저장 관련 연구로 허프 선 변환(Hough line Transform)[5]을 이용하여 바둑판 위의 가로, 세로 선

a) 광운대학교 컴퓨터과학과(Department of Computer Science, Kwangwoon University)
‡ Corresponding Author : 이윤구(Yun-Gu Lee)
E-mail: yglee96@kw.ac.kr
Tel: +82-2-940-8112
ORCID: http://orcid.org/0000-0001-8420-7196
※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]
※이 논문의 연구결과 중 일부는 "2017년 한국방송·미디어공학회 하계학술대회"에서 발표한 바 있음.
· Manuscript received September 4, 2017; Revised November 15, 2017; Accepted November 16, 2017.

및 교차점을 검출하고 교차점 근처 픽셀 그룹의 특성을 이용하여 기보를 자동으로 저장하는 연구가 있다[1]. 이 연구에서는 DOG(Difference of Gaussian) 혹은 3×3 크기의 변화량 검출 필터를 이용하여 경계선 검출 영상을 생성한 후, 허프 선 변환을 이용하여 가로 및 세로 각각 19개의 선을 검출해내었다. 선을 검출한 후 가로 선과 세로 선의 교차점의 위치를 계산하고 교차점 근처의 픽셀 그룹을 초승달 형태 혹은 작은 원의 형태로 추출한다. 그 후 추출한 픽셀 그룹의 밝기, 색상 정보 등을 이용하여 계산한 특정 값을 조건식을 이용하여 각 교차점 별 상태를 구분해내었다.

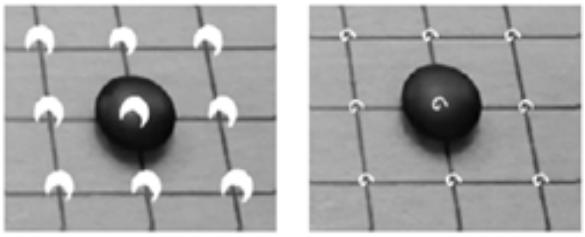

그림 1. 교차점 상태 추정을 위한 교차점 근처 픽셀 그룹 영역 추출
Fig. 1. An example of a pixel group region near the intersection for estimating states

또한, 바둑돌의 형태가 원형이라는 것에 착안하여 허프 원 변환[3]을 이용하여 바둑돌의 위치를 직접적으로 추출해내는 기법에 대한 연구[2]가 진행되었다. 해당 연구에서는 연산량을 감소시키기 위하여 바둑판 및 촬영 중인 카메라

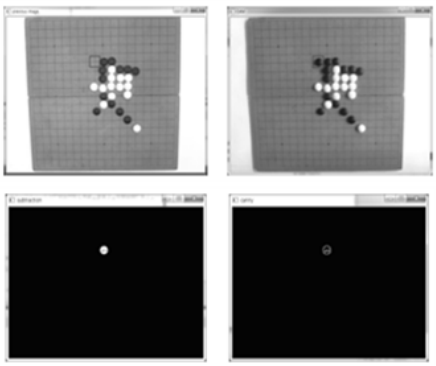

그림 2. 차 영상 및 허프 원 변환을 이용한 바둑 기보 저장
Fig. 2. Go game records using Hough circle transform and subtract Image

가 움직이지 않는다고 가정한 후, 현재 영상과 이전 영상의 차 영상을 구하고 차 영상에서 원을 검출하여 새로 놓인 돌 영역을 허프 원 변환을 이용하여 검출하였다.

바둑 기보를 자동으로 저장하기 위한 다른 연구로, 바둑돌의 투영(Projection) 위치 예측 결과를 통해 계산된 패치(Patch)를 이용하여 기보를 자동으로 저장하는 연구가 진행되었다[4]. 착수 예상 위치를 계산하기 위하여, 카메라의 내부 파라미터(Intrinsic Parameter)[6] 및 바둑판의 검은 선 최외각의 네 점[7]을 이용하여 카메라의 회전 및 평행이동 행렬인 외부 파라미터(Extrinsic Parameter)를 계산한다. 그 후 구 2개가 겹쳐진 형태를 이용하여 바둑돌을 모델링하고, 외부 파라미터와 모델링 결과를 통해 바둑돌의 영상 좌표를 계산한다. 그리고 계산된 영상좌표를 기반으로 해당 좌표 근처에서 직사각형 형태의 패치 영역을 추출하게 된다. 만약 촬영하는 카메라의 각도가 낮은 경우 앞쪽의 바둑돌에 의하여 뒤의 바둑돌이 일부분 가려지는 현상이 발생하므로 바둑돌의 위쪽 부분을 기준으로 패치 위치를 계산한다.

그림 3. 모델링된 바둑돌의 투영을 고려한 패치 추출
Fig. 3. Extraction of patch considering projection of modeled stones

Ⅲ. 문제 정의

앞서 언급한 기존 연구는 동영상이 아닌 착수 시점에서의 영상[2] 혹은 진행 중인 대국 중 임의의 시점의 영상을 추출[1]한 영상 시퀀스에서 동작하는 것을 기본 가정으로 한다. 하지만 바둑판 위의 방해물을 고려하지 않았기 때문에 손이나 물체가 바둑판 위에 존재하는 경우 바둑판이 가려진 부분이 인식되지 않는 문제가 발생한다. 따라서 동영상 및 실시간 촬영 환경에서의 정상적인 동작을 가능하게 하는 자동 기보 저장 알고리즘의 필요성이 존재한다.

일반적인 대국 진행 과정 예시는 그림 4와 같다. 일반적

인 바둑 대국의 경우 그림 4에서의 1, 2, 3번 영상과 같이 대국 중에는 바둑판 위에 바둑돌을 착수하기 위한 손과 같은 방해물이 바둑판 내부에 들어오며, 이로 인하여 바둑판이 가려져 해당 부분의 인식이 불가능하다. '방해물'은 카메라와 바둑판 사이에 존재하며 각 착점을 가리는 손 또는 임의의 물체를 의미한다. 만일 바둑 촬영 동영상에서 방해물이 바둑판 위에 존재 하지 않는 영상들을 알아낼 수 있다면, 해당 영상들을 이용하여 바둑 기보를 저장할 수 있다. 본 논문에서는 바둑 대국 촬영 동영상에서 바둑판 위에 방해물이 없는 영상만을 추출하는 알고리즘과 이를 이용하여 기보를 저장하는 알고리즘을 제안한다. 예를 들면, 그림 4에서의 4, 5, 6번째 또는 99, 100번째와 같은 영상을 방해물이 없는 영상 추출 과정에서 추출해내고, 해당 영상을 이용하여 기보 인식 알고리즘을 수행한다.

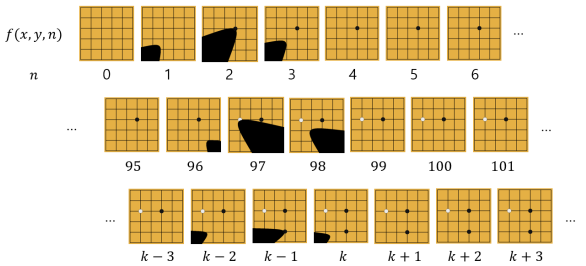

그림 4. 일반적인 바둑 대국 진행 과정
Fig. 4. General Go process

본 논문에서는 현재 영상과 배경 영상과의 차 영상을 이용하여 방해물 존재 여부를 확인하기 위해 몇 가지의 기본 환경을 설정한다. 먼저, 대국을 촬영하는 카메라 및 바둑판은 대국 중 움직이지 않아야 한다. 영상 내부의 바둑판이 움직이는 경우 기존에 제안된 물체 추적 알고리즘[8][9][10]을 이용하여 움직인 바둑판의 위치를 일정 부분 보완할 수 있으나, 본 논문에서는 해당 상황을 배제하였다. 두 번째로, 카메라의 내부 파라미터 변화로 인해 발생하는 복잡한 문제들을 배제하기 위하여 기보 저장 시작을 기준으로 초점거리를 고정시키고, 스마트폰 등에서 지원하는 AF(Auto focusing) 기능의 사용을 배제하였다. 세 번째로, 대부분의 스마트폰 및 웹캠 등에서 지원하는 자동 밝기 및 화이트 밸런스 조절 기능을 사용하지 않아야한다.

또한, 광원의 세기 및 위치는 일반적인 실내에서 사용되는 형광등의 위치 및 세기를 기준으로 한다. 광원이 매우 밝거나 어두운 경우, 혹은 광원이 바둑판 내부에서 균일하지 않은 경우 바둑판의 빛 반사로 인하여 방해물이 정상적으로 인식되지 않는 등의 문제가 발생할 수 있다. 광원의 위치 또한 성능에 영향을 미칠 수 있다. 광원과 바둑판이 이루는 각도가 작은 경우, 바둑돌의 그림자가 성능에 영향을 줄 수 있다. 그림자로 인한 오류는 기존에 연구되었던 그림자 제거 알고리즘[11][12]을 통하여 일정 부분 보완할 수 있으나, 각도가 낮거나 광원의 세기가 극단적인 경우는 일반적인 상황이 아니므로 본 논문에서는 일반적인 상황에서 촬영하는 것으로 환경을 제한한다. 해당 조건을 만족하지 않는 상황 예시는 그림 5의 (a)와 같다.

또한, 바둑돌 착수 간격이 일정 시간 이하인 경우 영상 내에 방해물이 연속적으로 존재하는 것으로 판단하기 때문에, 동시에 여러 개의 돌을 인식해야 하는 경우가 발생할 수 있다. 따라서 본 논문에서는 그림 4의 영상 $f(x,y,2)$ 또는 $f(x,y,97)$과 같이 한 번에 하나의 바둑돌만을 착수하는 것을 기준으로 하였다. 해당 조건을 만족하지 않는 상황 예시는 그림 5의 (b)와 같다. 마지막으로, 착수 순서는 일반적인 바둑 대국에서의 순서인 '검은 돌 - 흰 돌'을 기준으로 하였다.

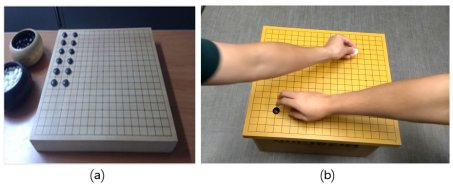

그림 5. 목표 환경에 맞지 않는 상황 예시
Fig. 5. Example situations that do not fit the target environment

Ⅳ. 제안 알고리즘

1. 개요

본 논문에서는 배경 영상 추출 알고리즘 및 해당 영상을 이용한 기보 인식 알고리즘을 제안한다. 바둑 대국에서의

'배경 영상'은 바둑판 위에 착수된 바둑돌을 제외한 어떠한 물체 혹은 물체의 움직임도 존재하지 않는 영상을 의미한다. m번째 배경 영상을 $f_{BG}(x,y,m)$이라 정의할 때, 방해물이 없는 영상 추출 과정에서는 m번째의 배경 영상을 찾기 위하여 기존의 배경 영상인 $f_{BG}(x,y,m-1)$과 입력 영상의 차 영상을 이용하여 바둑판 위에 방해물이 존재하는지를 확인하고, 이 과정을 방해물이 없는 영상을 찾을 때까지 반복한다. 해당 과정을 통하여 찾은 영상을 $m+1$번째의 방해물이 없는 영상인 $f_{BG}(x,y,m+1)$을 찾기 위해 새로운 배경 영상인 $f_{BG}(x,y,m)$으로 설정한다. 첫 번째 방해물이 없는 영상인 $f_{BG}(x,y,1)$을 찾기 위하여, 첫 배경 영상인 $f_{BG}(x,y,0)$은 동영상의 첫 프레임으로 설정한다. 해당 과정에 대한 예시는 그림 6과 같다.

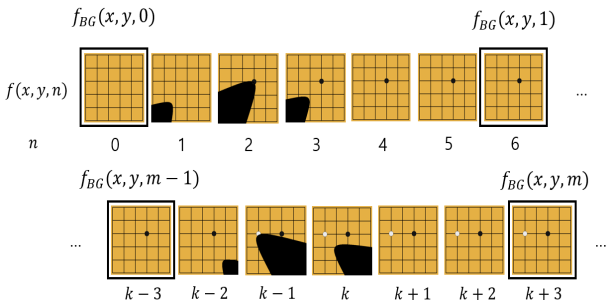

그림 6. 방해물이 없는 영상 추출 과정
Fig. 6. image without obstacle extraction process

배경 영상 추출 과정이 정상적으로 진행되었다면, 이전 배경 영상인 $f_{BG}(x,y,m-1)$과 추출한 영상인 $f_{BG}(x,y,m)$을 이용하여 기보 인식 과정을 진행한다. 기보 인식 과정은 관련 연구에서 언급하였던 패치[4]를 이용한다. 본 논문에서 제안하는 알고리즘의 전반적인 진행도는 그림 7과 같다.

2. 배경 영상 추출 과정

배경 영상 추출 과정은 크게 전처리 과정 및 특정 조건을 이용한 방해물 존재 여부 판별 과정으로 이루어진다. 전처리 단계에서는 영상 내의 잡음을 제거하기 위하여 11×11 크기의 평균 필터를 적용하고, 밝기 영역에서의 차이만을 이용하기 위하여 회색조 영상으로 변환하였다. 또한 바둑판 위의 상태만을 분석하기 위하여 최외각 네 점 추출[7] 및 초점거리 예측을 진행[6]하고, 이를 이용해 원근 변환을 적용하여 바둑판 전체를 정사각형 모양으로 변환한다.

다음 전처리 과정으로 이전 배경 영상인 $f_{BG}(x,y,m-1)$과 현재 영상인 $f(x,y,n)$을 이용하여 n번째 영상의 차 영상인 $f_{D_n}(x,y)$를 구한다. 이 때 밝기 기반으로 차 영상을 구하는 경우 바둑판 선 근처 픽셀들에 잡음이 존재하게 된다. 이를 해결하기 위해 평균 필터 등을 강하게 적용하여 일정 부분 제거 할 수 있으나, 방해물 영역의 일부도 같이 사라지게 된다. 본 논문에서는 안전하게 방해물 존재 여부를 판단하기 위해, 판단에 영향을 미치지 않는 정도의 평균

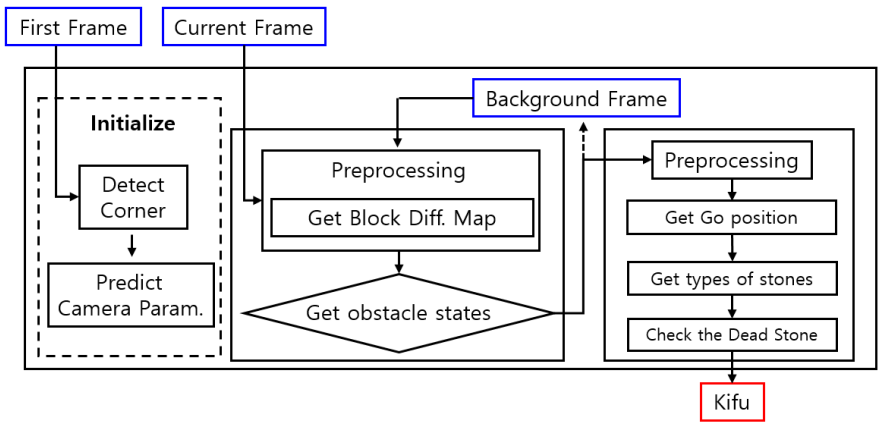

그림 7. 방해물 분석을 통한한 자동 기보 인식 알고리즘
Fig. 7. Algorithm of recognition of Go game positions using obstacle analysis

필터를 실험적으로 정한 필터 크기인 11로 적용한다. 또한 다음 과정을 위해서는 전처리 과정에서 방해물 존재 여부를 나타내는 이진화된 정보가 필요하므로, 본 논문에서는 8x8 픽셀 크기의 블록(Block)으로 이루어진 블록 단위 차이 맵(Block difference map)을 설정하고 이를 배경 영상 추출 기준으로 이용한다. 이를 이용함으로써 바둑판 위의 선 근처 픽셀의 잡음을 감소시킬 수 있고, 차 영상 내부의 방해물 영역과 아닌 영역을 이진화된 형태의 정보로 변환할 수 있다.

블록 내부의 값은 바둑판 내의 방해물 또는 바둑판 외부 물체로 인해 발생하는 그림자의 영향을 감소시키기 위하여 차 영상의 x, y 방향의 변화량을 구하고 해당 블록 위치의 변화량을 모두 더한 결과로 채운다. 현재 영상을 $f(x,y,n)$이라 할 때, 차 영상의 (x_1, y_1) 위치에서의 변화량 $f_{D_n}(x_1, y_1, n)$을 계산하는 수식은 다음과 같다.

$$f_{D_n}(x_1,y_1,n) = |f_{D_n}(x_1,y_1,n) - f_{D_n}(x_1,y_1+1,n)| + |f_{D_n}(x_1,y_1,n) - f_{D_n}(x_1+1,y_1,n)| \quad (1)$$

변화량 $f_{D_n}(x,y,n)$을 모든 위치에서 구한 후, 현재 영상의 크기가 $M \times M$인 l번째 블록 $f_B(l,n)$은 다음과 같은 수식을 이용하여 계산한다.

$$f_B(l,n) = \sum_{y=y_0}^{y_0+M} \sum_{x=x_0}^{x_0+M} f_{D_N}(x,y,n) \quad (2)$$

이 때, x_0, y_0는 블록의 좌측 상단의 픽셀 좌표이다. 영상 내의 모든 블록을 값을 채운 후, 임계연산을 통하여 블록 단위 차이 맵을 이진화한다. 임계값은 실험적으로 15를 적용하였다. 블록 단위 차이 맵 설정 예시는 그림 8과 같다.

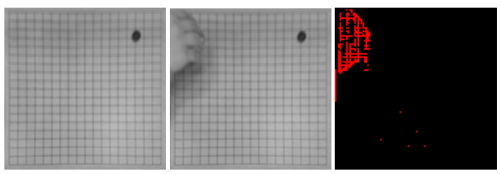

그림 8. 배경 영상을 이용한 블록 단위 차이 맵 설정
Fig. 8. 'Block Difference Map' using background frame

전처리 단계를 진행하여 블록 단위 차이 맵을 만든 이후, 다음 단계인 방해물 존재 여부 판별 과정으로 진행한다. 해당 과정에서는 현재 블록 단위 차이 맵인 $f_B(l,n)$과 이전 영상에서 설정하였던 $f_B(l,n-1)$을 비교하여 상태가 바뀐 블록의 개수를 기준으로 방해물 존재 여부를 판단한다. 이전과 현재의 동일한 위치에 있는 블록의 상태가 다른 경우, 해당 블록은 방해물이 바둑판 내로 진입 중이거나 빠져나가고 있는 상태이기 때문에 해당 블록은 방해물 근처 영역이거나 방해물 내부인 것으로 판단한다. 본 논문에서는 실험적으로 블록의 개수 기준을 15로 정하였다.

만약 착수를 진행하였을 경우 현재의 차 영상 f_{D_n} 내부에 그림 9와 같이 새로 착수된 바둑돌 영역이 남게 된다. 이러한 문제를 해결하기 위해 본 논문에서는 위의 조건을 만족하는 블록의 개수가 일정 프레임 동안 유지되는 경우 영상 내에 방해물이 없는 것으로 판단하고 해당 영상을 추출한다. 본 논문에서는 해당 기준을 10 프레임으로 설정하였다.

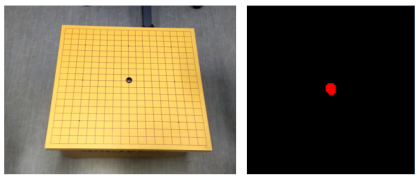

그림 9. 착수 후의 블록 단위 차이 맵 예시
Fig. 9. Block difference map after placement

블록의 개수를 구한 후, 이를 토대로 영상 내의 방해물 존재 상태를 설정한다. 본 논문에서는 영상 내의 방해물 존재 상태를 세 가지로 분류하였다. 'Obstacle-Exist' 상태는 바둑판 내에 방해물이 존재하는 상태로, 바둑 기보 인식 과정으로 진행하지 않고 다음 영상을 이용하여 방해물 존재 여부를 다시 판단한다. 'Obstacle-No' 상태는 바둑판 내에 방해물이 존재하지 않는 상태로, 배경 영상 갱신 및 기보 저장 과정을 진행한다. 'Obstacle-Update' 상태는 바둑판 내에 방해물이 지속적으로 존재하지 않는 상태로, 기보 인식 과정을 진행하는 것을 방지하여 연산량을 감소시키는 목적으로 사용된다. 실제 바둑 대국 촬영 동영상의 각 영상에서 계산된 블록 개수의 변화에 따른 방해물 존재 상태 설정 결과는 그림 10과 같다.

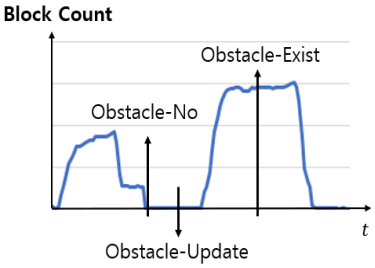

그림 10. 블록의 개수에 따른 방해물 존재 상태
Fig. 10. Current obstacle-exist states using the number of blocks

3. 갱신 영상을 이용한 바둑 기보 인식

움직임 판별 과정에서 'Obstacle-No' 상태로 설정되었을 경우 이전의 배경 영상과 이전 과정에서 방해물이 없는 것으로 판단된 영상을 기반으로 착수 위치 및 바둑돌 종류를 인식하고, 사석 존재 여부를 판단하여 저장하는 기보 인식 과정 및 사석 처리 과정을 진행한다.

기보 인식 과정은 2절에서 서술하였던 각 착수 예상 위치에서 일정 직사각형 크기로 설정된 패치[4]를 이용하여 진행된다. 패치의 위치 및 크기는 카메라 캘리브레이션을 통하여 계산된 카메라 위치를 이용하여 카메라와 각 착수 예상 위치간의 각도, 거리 및 착수 시 돌의 2D 영상 좌표 등을 종합하여 계산된다. 카메라의 촬영 각도에 따른 패치 계산 결과에 대한 예시는 그림 11과 같다.

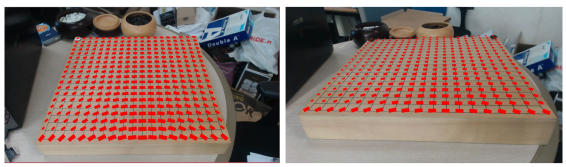

그림 11. 패치 위치 계산 결과 예시
Fig. 11. Example of patch position calculation results

착수 위치 인식 및 바둑돌 종류 판별은 위 과정에서 계산된 패치의 밝기의 변화량을 기반으로 한다. 착수 위치 인식 과정에서는 먼저 배경 영상과 현재 영상을 각각 11×11 크기의 평균 필터를 통과시켜 잡음을 감소시킨 후 회색조 영상으로 변환한다. 이후 각 영상의 착수 예상 위치에서 현재 영상과 배경 영상의 패치의 MAD(Mean of Absolute Difference)가 설정한 문턱값 이상인 경우 해당 위치에 착수되었다고 판단한다. 본 논문에서는 실험적으로 문턱값을 20으로 설정하였다. 바둑돌이 놓인 부분의 패치는 MAD 값이 주변에 비하여 큰 값을 가지기 때문에, 문턱값을 이용하여 일정 값 이상인 패치의 위치를 착수 위치로 저장한다.

바둑돌 종류 인식 과정에서는 회색조로 변환된 현재 영상과 배경 영상의 착수 위치의 패치의 차이의 누적을 계산하여 양수인 경우 흰 돌, 음수인 경우 검은 돌로 인식한다. 이 과정에서 MAD 값이 가장 큰 두 패치만을 착수된 위치로 설정하고, 가장 최근에 착수된 돌의 종류와 반대인 바둑돌의 착수 위치 및 바둑돌 종류를 저장한다.

착수된 바둑돌의 종류 및 위치를 인식한 후, 바둑판 위의 사석을 제거하는 과정을 진행한다. 바둑돌을 추가하는 과정은 바둑돌이 착수되지 않은 착수 예상 위치의 변화량을 이용하지만, 바둑돌을 제거하는 과정은 이미 바둑돌이 놓인 착수 위치의 상태 및 변화량을 이용한다.

해당 과정에서는 먼저 현재 착수된 바둑돌의 위치 및 바둑판 전체의 착수 상태 정보를 이용하여 제거해야 할 바둑돌의 위치를 찾는다. 만약 제거해야 할 바둑돌이 존재한다면, 해당 위치의 MAD 및 패치의 차이의 누적을 이용하여 제거 여부를 판단한다. 만약 검은 돌을 제거해야하는 상황이라면, 사석 위치의 MAD가 문턱 값 이상이고 차이의 누적이 양수인 경우에만 해당 위치의 상태를 돌이 없는 상태로 변경한다. 해당 조건을 만족하지 않는 경우 다음 배경 영상 갱신 시 해당 위치의 변화 여부를 다시 확인하고, 돌이 제거되는 시점까지 이 과정을 반복한다.

V. 실험 및 결과

제안하는 알고리즘을 실험하기 위하여, 웹캠 혹은 스마트폰으로 촬영된 바둑 대국 동영상 및 구현한 프로그램을 이용하여 자동으로 바둑 기보를 저장하고 실제 정답과 비교하여 정답률을 계산하였다. 각 테스트 영상의 특성 및 촬영 환경은 표 1과 같다.

표 1. 테스트 영상의 특성 및 환경
Table 1. Characteristics and environment of test video

Test Video	Environmentt
TestSet_00, TestSet_01	It is a video of a Go game taken with the camera's focus and brightness automatic control function turned on.
TestSet_02, TestSet 03	The internal parameters and the automatic brightness control are off, but there is a process where the shooting angle is low and black stones are not followed by white stones, or stones are removed.
TestSet_04	It is a very fast video of a Go game with the focus and brightness automatic control turned off, but the interval between placing a stone is within 0.2 to 0.3 seconds.
TestSet_05, TestSet_06	The internal parameters and automatic brightness control were turned off, and the angle between the Go-board and the camera was about 45 °, which was taken indoors. The interval between placing a stone was about 5 seconds to 10 seconds, and we shoot a game of a Omok, not a Go.
TestSet_07 ~ TestSet_09	The shooting environment is the same as the above item, and it is a video that placing a stone at the intersection above the board) has over 100 moves.

모든 테스트 영상 촬영 시 카메라 및 바둑판은 고정된 상태이다. 대국에서 총 착수 개수를 N_i, 프로그램이 정확히 인식한 기보의 개수를 C_i라 할 때, 정답률 P_i는 다음과 같은 식을 이용하여 계산하였다. 대국 촬영 동영상을 입력으로 한 자동 기보 저장 테스트 결과는 표 2 및 그림 12와 같다.

$$P_i = \frac{C_i}{N_i} \times 100 \ [\%] \tag{3}$$

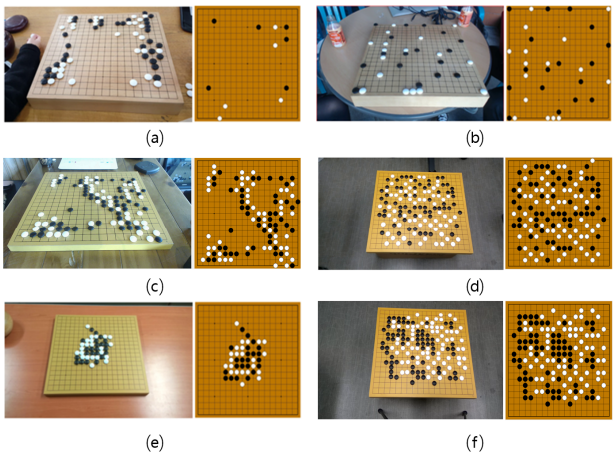

그림. 12. 알고리즘 실험 결과 영상
Fig. 12. Experimental result image of algorithm test

움직임 분석 과정과 기보 저장 과정은 PC를 기준으로 하여 각각 22~24msec, 65~75 msec의 시간을 소요하였다. 인식 결과가 정답과 일치하지 않는 오답 상황은 다음과 같은 상황에서 발생하였다.

먼저 카메라의 내부 파라미터 및 밝기 자동 조절 기능을 켜둔 상태에서 촬영한 테스트 영상인 TestSet_00 및 TestSet_01의 경우, 초점의 변화로 인하여 바둑판 전체가 움직이는 효과를 발생시키기 때문에 블록 단위 차이 맵을 설정하였을 때 바둑판 전역에 방해물이 존재하는 것으로 판별되었다. 이로 인하여 초점 변화가 발생한 이후의 모든 기보를

표 2. 테스트 영상을 이용한 저동 기보 저장 실험 결과
Table 2. Recognition of Go game positions test using test video

Test Video	Smartphone / WebCam Model	resolution	fps	N_i	C_i	P_i (%)
TestSet_00	Samsung Galaxy Note 5	1080p	30.0	151	10	6.62
TestSet_01	Samsung Galaxy Note 5	1080p	30.0	137	48	35.03
TestSet_02	Logitech c920 HD Pro	1080p	30.0	33	32	96.97
TestSet_03	Logitech c920 HD Pro	1080p	30.0	52	51	98.07
TestSet_04	Logitech c920 HD Pro	1080p	30.0	120	104	86.67
TestSet_05	iPhone 6s+	1080p	30.0	31	31	100
TestSet_06	iPhone 6s+	1080p	30.0	46	46	100
TestSet_07	Xiaomi Redmi Note 3	1080p	30.0	100	100	100
TestSet_08	Xiaomi Redmi Note 3	1080p	30.0	146	146	100
TestSet_09	Xiaomi Redmi Note 3	1080p	30.0	130	128	98.46

인식하지 못하는 문제가 발생하였다. 초점 변화로 인한 블록 단위 차이 맵 오류 영상은 그림 13의 (b), (c)와 같다. 이와 같은 상황이 발생한 결과 예시는 그림 12의 (a)와 같다.

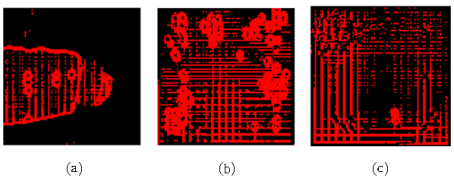

그림. 13. 테스트 영상의 블록 단위 차이 맵 예시
Fig. 13. 'Block Motion Map' of test video (a) TestSet_02 (b) TestSet_00 (c) TestSet_01

두 번째로, TestSet_02에서는 일반적인 바둑 대국에서의 착수 순서를 따르지 않아 해당 착수가 누락되는 오류가 발생하였다. 이와 같은 상황이 발생한 결과 영상은 그림 12의 (b)와 같다. 또한, TestSet_03에서는 사석이 아닌 바둑돌을 제거하였고 이를 감지하지 못하여 해당 착수가 누락되는 오류가 발생하였으나, 이는 일반적인 바둑 대국의 상황이 아니므로 고려하지 않는다.

TestSet_04에서는 착수 시점 간의 시간차가 10프레임 이하로 매우 짧은 상황이 존재하는데, 이 경우 본 알고리즘에서는 방해물이 연속해서 존재하는 것으로 판별하였다. 이 경우 착수 간격이 10프레임 이상인 시점에서 기보 저장을 진행하게 되는데, 만약 동시에 여러 개의 돌을 인식하여 저장해야 하는 경우 MAD 값이 큰 2개의 착수 위치만 저장되고 나머지 착수된 돌은 무시된다. 이와 같은 상황이 발생한 결과 영상은 그림 12의 (c)와 같다.

마지막으로, TestSet_09에서 발생한 오류의 경우 바둑판의 외각에 바둑돌을 착수할 때 손이 바둑판 위에 있음에도 블록의 개수가 기준 이하여서 배경 영상 갱신이 진행되어 발생하였다. 이와 같은 상황이 발생한 결과 영상은 그림 12의 (d)와 같다. 이러한 상황에 대한 보정 알고리즘을 추가해야 할 필요성이 존재한다.

Ⅵ. 결 론

본 논문에서는 일반적인 대국 촬영 상황에서는 바둑판과 카메라를 고정시키므로 영상 내에서 바둑판이 움직이지 않는다는 가정을 바탕으로 카메라의 내부 파라미터가 고정되어 있고 렌즈 왜곡이 없는 상황에서 배경 영상과 현재 영상의 차이의 변화량을 이용하여 방해물의 존재 여부를 판별하고 방해물이 없는 영상을 새로운 배경 영상을 갱신하여 해당 과정이 연속적으로 이루어질 수 있도록 하였다. 또한 카메라 캘리브레이션 및 바둑돌의 예측된 영상 좌표 등을 이용하여 계산된 패치 영역의 차이를 이용하여 바둑 기보를 자동으로 저장하는 기법을 제안하였다. 기존의 연구와 비교해 보았을 때, 동영상 내에 방해물이 존재하지 않는 영상을 추출할 수 있으므로 기존의 연구에 사용되었던 방법에 비해 비교적 간단한 연산을 이용하여 착수 위치 및 바둑돌의 종류를 인식해낼 수 있을 것으로 예상한다.

하지만 바둑판의 외각 영역에 착수하는 상황에서 움직임이 매우 미세하거나 바둑판 위에서 장시간 움직이지 않는 경우 물체의 움직임이 바둑판 위에 존재함에도 불구하고 움직임이 없는 것으로 판별하는 오류가 발생할 수 있다. 또한 광원이 매우 밝아 바둑판에 빛 반사가 강하게 일어나거나 강한 그림자가 발생하는 경우 흰색 혹은 검은 색 돌의 인식이 제대로 이루어지지 않을 수 있으므로 향후 이러한 문제점들을 보완할 수 있는 기법에 대한 연구가 진행되어야 할 것이다.

참 고 문 헌 (References)

[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] D. Park and K, Jun, "CHT-based Automatic Go Recording System under Illumination Change and Stone Dislocation", Journal of KIISE : Software and Application, Vol.41, No.6, pp.448-455, June 2014.
[3] BALLARD, Dana H. "Generalizing the Hough transform to detect arbitrary shapes". Pattern Recognition, Vol.13, No.2, pp.183-194, 1991.
[4] D. Lee and Y. Lee, "Algorithm of recognizing Go stones for the automatic records of Go games", IEIE Summer Conference, Jeju, Korea, pp.809-812, 2016.
[5] DUDA, Richard O. and HART, Peter E, "Use of the Hough transformation to detect lines and curves in pictures", Communications of the ACM, Vol.15, No.1, pp.11-15. 1972.
[6] K. Lee, and Y. Lee, "Prediction of Camera Intrinsic Parameter using Go Board Image", The Korean Institute of Broadcast and Media Engineers Summer Conference, Jeju, Korea, pp.237-240, 2017.6.

[7] Y. Yoon and Y. Lee, “Algorithm of detecting Go board corners”, The Korean Institute of Broadcast and Media Engineers Summer Conference, Jeju, Korea, pp.233-236, 2017.6.
[8] D. Comaniciu, V. Ramesh and P. Meer, "Kernel-based object tracking," IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol.25, No.5, pp.564-577, 2003.
[9] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," *Proceeding of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Fort Collins, Colorado, USA, pp. 246-252, 1999.
[10] Euncheol Choi, Suk-Ho Lee, Moon Gi Kang. “Object Tracking Algorithm Using Weighted Color Centroids Shifting”. *JOURNAL OF BROADCAST ENGINEERING*, Vol.15, No.2, pp.236-247, 2010
[11] Kim Cheol-Mun, Kwak Gae-Ho, Kim Whoi-Yul. “Moving Cast Shadow Detection based on Global Gaussian Modeling”. *The Korean Institute of Broadcast and Media Engineers Conference Proceedings*, Seoul, Korea, pp.259-262. 2009.
[12] Divya, K. A. Roshna, K. I. Mathai and Shelmy, "Shadow detection and removal by object-wise segmentation." Computational Intelligence and Computing Research, International Conference on IEEE, Madurai, India, pp. 1-4, 2015

저 자 소 개

김 민 성

- 2015년 8월 : 광운대학교 전자공학과 공학사
- 2017년 8월 : 광운대학교 컴퓨터과학과 석사
- ORCID : https://orcid.org/0000-0001-6430-6138
- 주관심분야 : 영상 처리, 비디오 코딩, 패턴 인식 등

윤 여 경

- 2017년 2월 : 광운대학교 컴퓨터소프트웨어학과 공학사
- 2017년 3월 ~ 현재 : 광운대학교 컴퓨터과학과 석사과정
- ORCID : http://orcid.org/0000-0061-6969-9110
- 주관심분야 : 영상 처리, 패턴 인식, Virtual Reality, 영상 정합 등

이 광 진

- 2017년 2월 : 광운대학교 컴퓨터소프트웨어학과 공학사
- 2017년 3월 ~ 현재 : 광운대학교 컴퓨터과학과 석사과정
- ORCID : http://orcid.org/0000-0001-8226-4998
- 주관심분야 : 영상 처리, 영상 정합, Machine Learning 등

이 윤 구

- 2000년 2월 : 한국과학기술원 전자전산학과 공학사
- 2002년 2월 : 한국과학기술원 전자전산학과 석사
- 2006년 8월 : 한국과학기술원 전자전산학과 박사
- 2006년 9월 ~ 2013년 2월 : 삼성전자 DMC연구소 책임/수석연구원
- 2013년 3월 ~ 2015년 2월 : 광운대학교 컴퓨터소프트웨어학과 조교수
- 2015년 3월 ~ 현재 : 광운대학교 컴퓨터소프트웨어학과 부교수
- ORCID : http://orcid.org/0000-0001-8420-7196
- 주관심분야 : 영상처리, 컴퓨터비젼, 비디오 코딩, 카메라 시스템 등