

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
DE SÃO PAULO - CAMPUS SÃO CARLOS

ESPECIALIZAÇÃO *LATO SENSU* EM
DESENVOLVIMENTO DE SISTEMAS PARA DISPOSITIVOS MÓVEIS

Leonardo Sameshima Taba

**Registro automático de partidas de Go
por meio de processamento de imagens
embarcado em dispositivos móveis**

São Carlos – SP
2016

Leonardo Sameshima Taba

**Registro automático de partidas de Go
por meio de processamento de imagens embarcado em
dispositivos móveis**

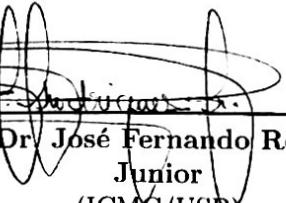
Monografia apresentada ao Instituto Federal de Educação, Ciência e Tecnologia de São Paulo – campus São Carlos, como parte dos requisitos exigidos para a obtenção do título de especialista em Desenvolvimento de Sistemas para Dispositivos Móveis. Orientador: Rodrigo Elias Bianchi

Data da aprovação: 22 / 06 / 2016

MEMBROS DA BANCA EXAMINADORA:


Prof. Dr. Rodrigo Elias Bianchi
(Orientador)
(IFSP – campus São Carlos)


Prof. Dr. Fernando Vernal Salina
(IFSP – campus São Carlos)


**Prof. Dr. José Fernando Rodrigues
Junior**
(ICMC/USP)

Brasil
Junho de 2016

| | |
|------|--|
| T11r | <p>Taba, Leonardo Sameshima Registro automático de partidas de Go por meio de processamento de imagens embarcado em dispositivos móveis / Leonardo Sameshima Taba; orientador, Prof. Dr. Rodrigo Elias Bianchi – São Carlos, SP, 2016. 37 p. : il. color.</p> <p>Monografia (Especialização) – Instituto Federal de Educação, Ciência e Tecnologia de São Paulo, Campus São Carlos, Especialização Lato Sensu em Desenvolvimento de Sistemas para Dispositivos Móveis.</p> <p>Inclui referências bibliográficas</p> <p>1. Computação. 2. Desenvolvimento de Dispositivos Móveis. 3. Go. 4. Processamento de Imagens. I. Bianchi, Rodrigo Elias, orient. II. Instituto Federal de São Paulo. III. Título.</p> |
| | CDD 005.282 |

*A todos os jogadores e jogadoras de Go
e todos que buscam melhorar a cada dia.*

Resumo

Go é um jogo de estratégia criado na China há cerca de 3000 anos. É um dos mais antigos jogos de tabuleiro jogados até hoje e tem milhões de jogadores em todo o mundo. Apesar do grande avanço tecnológico ocorrido nesses três milênios, o registro de partidas de Go ainda é feito quase que exclusivamente à mão, um método pouco prático e suscetível a erros. Nos dias atuais, dispositivos móveis como smartphones e tablets são cada vez mais comuns e têm poder de processamento cada vez maior, além de recursos avançados tais como câmeras de alta resolução. Considerando esse cenário e a pouca praticidade para registrar partidas de Go manualmente, este trabalho apresenta um método que utiliza a câmera de dispositivos móveis para realizar o registro automático de partidas desse jogo. O método proposto faz uso de técnicas de processamento de imagens e realiza o processamento exclusivamente no dispositivo, dispensando a necessidade de conexão à internet ou de outros acessórios. Um aplicativo para dispositivos Android que implementa esse método de registro de partidas foi desenvolvido e foram realizados experimentos para testar sua precisão e utilidade no registro de jogos reais. Os resultados encontrados foram muito positivos, mostrando que é possível realizar o registro de partidas de Go de forma automática e confiável utilizando dispositivos móveis.

Palavras-chaves: Go, processamento de imagens, dispositivos móveis.

Abstract

Go is a strategy game created in China around 3,000 years ago. It's one of the oldest board games still played up to this day and has millions of players around the world. Despite the great technological advances that occurred in these three millennia, recording Go games is still done almost exclusively by hand, which is impractical and error prone. Nowadays, mobile devices such as smartphones and tablets are becoming more and more common and have increasing processing power, in addition to advanced resources such as high resolution cameras. Considering this scenario and the impracticality of registering Go games manually, this work presents a method that uses mobile devices's cameras to register Go games automatically. The proposed method uses image processing techniques and does the processing entirely in the device, eliminating the necessity of an internet connection or other accessories. An application for Android devices that implements the game recording method was developed and experiments were performed to test its precision and usefulness in registering real games. The results found were very positive, showing that it's possible to register Go games in an automatic and reliable way using mobile devices.

Key-words: Go, image processing, mobile devices.

Lista de ilustrações

| | |
|---|----|
| Figura 1 – Partida de Go em andamento | 15 |
| Figura 2 – Registro de partida (kifu) tradicional feito à mão | 16 |
| Figura 3 – Imagem de partida transmitida pela televisão sul-coreana | 19 |
| Figura 4 – Visão de alto nível das etapas do processamento | 26 |
| Figura 5 – Foto utilizada nos exemplos de processamento | 27 |
| Figura 6 – Foto de exemplo com filtro Camy e dilatação aplicados | 28 |
| Figura 7 – Imagem anterior com contornos identificados | 28 |
| Figura 8 – Imagem anterior com os quadriláteros encontrados | 29 |
| Figura 9 – Imagem anterior com a hierarquia de quadriláteros identificada | 29 |
| Figura 10 – Tabuleiro transformado para visão ortogonal | 30 |
| Figura 11 – Região de interesse ao redor de uma interseção | 31 |
| Figura 12 – Tabuleiro identificado | 33 |
| Figura 13 – Exemplo de foto utilizada no primeiro experimento | 35 |
| Figura 14 – Outro exemplo de foto utilizada no primeiro experimento | 36 |
| Figura 15 – Primeira partida registrada. | 38 |
| Figura 16 – Segunda partida registrada. | 38 |
| Figura 17 – Terceira partida registrada. | 38 |
| Figura 18 – Quarta partida registrada. | 38 |
| Figura 19 – Quinta partida registrada. | 39 |
| Figura 20 – Sexta partida registrada. | 39 |
| Figura 21 – Sétima partida registrada. | 39 |

Lista de tabelas

Sumário

| | | |
|----------|---|-----------|
| 1 | Introdução | 15 |
| 1.1 | Organização do trabalho | 17 |
| 2 | Revisão Bibliográfica | 19 |
| 2.1 | Aplicativos para dispositivos móveis | 22 |
| 3 | Desenvolvimento | 25 |
| 3.1 | Método | 25 |
| 3.1.1 | Detecção do tabuleiro | 26 |
| 3.1.2 | Detecção das pedras | 30 |
| 3.1.3 | Detecção de jogadas | 33 |
| 3.2 | Implementação | 34 |
| 4 | Experimentos e Resultados | 35 |
| 4.1 | Testes com fotos | 35 |
| 4.2 | Experimento com partidas reais | 36 |
| 4.3 | Discussão | 38 |
| 4.3.1 | Comparação com trabalhos relacionados | 40 |
| 5 | Conclusão | 43 |
| 5.1 | Trabalhos futuros | 43 |
| | Referências | 45 |

1 Introdução

Go (também conhecido como *WeiQi* em chinês e *Baduk* em coreano) é um jogo de tabuleiro criado na China há cerca de 3000 anos. Suas regras são simples: basicamente, dois jogadores, um com pedras pretas e outro com brancas, alternam jogadas nas interseções de um tabuleiro de 19 linhas por 19 colunas (tamanhos menores para iniciantes também são comuns). As pedras, depois de colocadas, nunca se movem, mas podem ser capturadas. O vencedor é aquele que conseguir dominar mais território com suas pedras. A Figura 1 ilustra um jogo de Go em andamento.

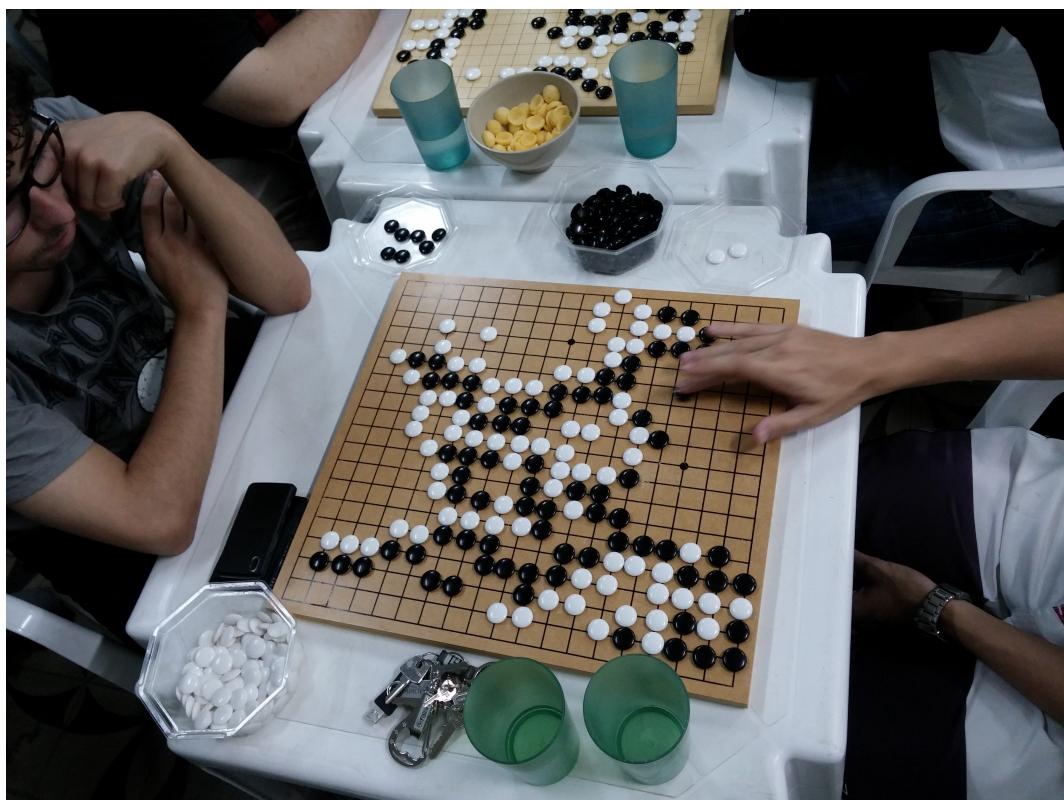


Figura 1 – Partida de Go em andamento (acervo pessoal do autor)

Apesar da simplicidade das regras¹, sua estratégia é extremamente complexa. O Go tem fascinando milhões de pessoas ao longo dos anos e é muito popular até os dias de hoje, sobretudo no oriente.

Tal qual outros jogos de estratégia como o xadrez, um dos principais métodos para o estudo e aprimoramento no Go é a reprodução e revisão de partidas. Tradicionalmente, o registro de partidas é feito utilizando diagramas em papel com anotações manuais de cada jogada (chamado de *kifu* em japonês), como o apresentado na Figura 2. Em jogos profis-

¹ As regras do jogo podem ser encontradas em diversos sites, como <http://go.alamino.net/playgo/>

sionais existem pessoas incumbidas apenas com essa função, de modo a não atrapalhar os jogadores. Entretanto, quando amadores desejam registrar jogos dificilmente pode-se contar com um terceiro para realizar esse trabalho, ficando a cargo de um dos jogadores fazê-lo. Isso pode atrapalhar o andamento do jogo e desconcentrar os jogadores, já que não é um processo prático e é suscetível a erros (SRISUPHAB et al., 2012).

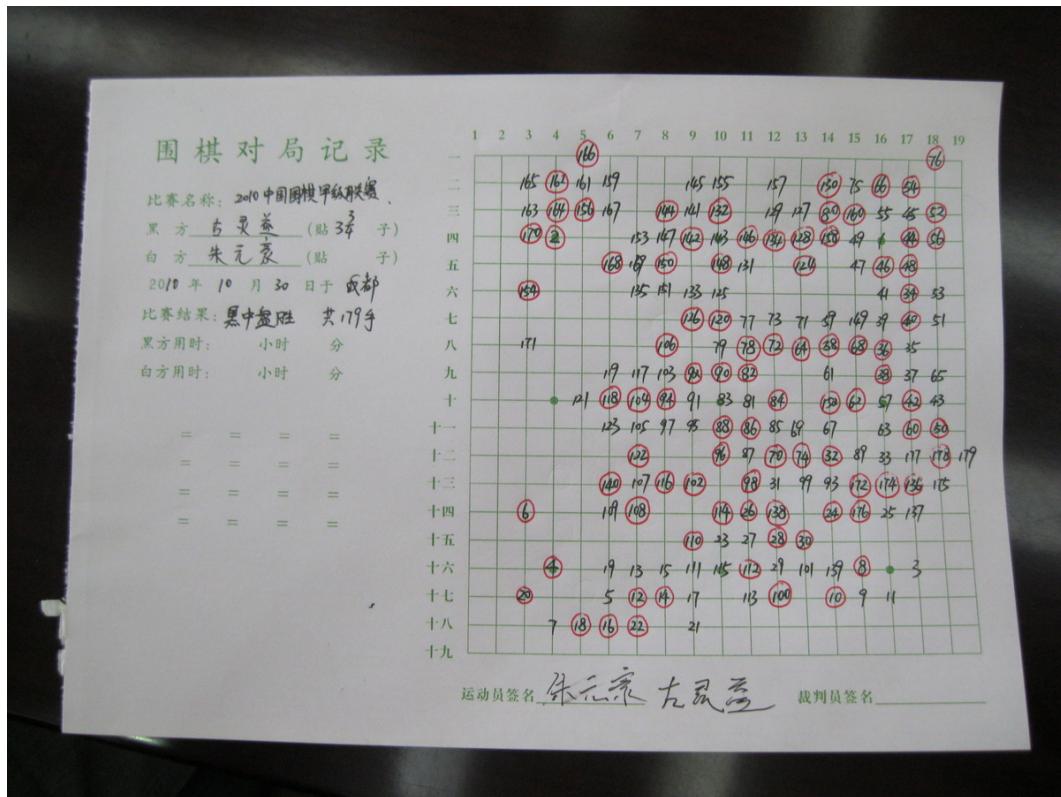


Figura 2 – Registro de partida (kifu) tradicional feito à mão (<http://blog.gokifu.com/2010/10/hello-world/>)

Considerando essa falta de praticidade no registro de partidas e levando em conta que nos dias atuais *smartphones* e *tablets* com câmera são cada vez mais comuns e têm poder de processamento cada vez maior, este trabalho apresenta um método automático para o registro de partidas de Go utilizando a câmera de dispositivos móveis.

Um aplicativo para dispositivos Android que implementa esse método foi desenvolvido e testado tanto com fotos de tabuleiros quanto em partidas reais e obteve boa precisão. Os resultados mostram que o método é robusto e útil para o registro de partidas de Go reais, podendo ser usado para realizar a tarefa custosa e suscetível a erros de anotação de partidas.

Até a conclusão desta pesquisa não foram encontrados aplicativos para dispositivos móveis que realizem essa tarefa de forma simples, satisfatória e com boa precisão. Isso mostra que este trabalho preenche uma grande lacuna na área de técnicas de apoio ao jogo de Go.

1.1 Organização do trabalho

O Capítulo 2 traz uma revisão bibliográfica sobre o tema de registro automático de partidas de Go, citando também alguns softwares relacionados. O Capítulo 3 apresenta o método proposto para a realização da tarefa em questão. O Capítulo 4 mostra os experimentos realizados para testar o método e seus resultados. Por fim, o Capítulo 5 traz as conclusões deste trabalho e possibilidades de trabalhos futuros.

2 Revisão Bibliográfica

Um dos primeiros trabalhos sobre o tema é de Shiba e Mori (2004), que, apesar de não objetivar o registro de partidas em si, buscou a detecção dos contornos de tabuleiros de Go em fotos. Segundo os autores, esse seria o primeiro passo para o registro de uma partida. São utilizados algoritmos genéticos para processar imagens de baixa resolução em preto e branco. Como este artigo tem um escopo limitado, ele não será discutido em mais detalhes.

Já o trabalho de Yanai e Hayashiyama (2006) apresenta um método para o registro de jogos a partir de vídeos transmitidos pela televisão. Foram utilizadas algumas técnicas de processamento de imagens, como aplicação de filtros para identificar fronteiras e a transformada de Hough (DUDA; HART, 1972), para detectar as linhas do tabuleiro. O método foi aplicado sobre 8 programas de Go da emissora japonesa NHK, alcançando uma precisão e cobertura médias de 95,7% (a precisão foi calculada como o número de jogadas detectadas corretamente dividido pelo número de jogadas detectadas, e a cobertura como o número de jogadas detectadas corretamente dividido pelo número total de jogadas).



Figura 3 – Imagem de partida transmitida pela televisão sul-coreana (<https://gogameguru.com/watch-baduk-tv-free-limited-time/>)

Esses são bons resultados, no entanto deve-se observar que em transmissões televisivas de partidas de Go existem algumas peculiaridades que tornam o problema mais simples. Uma das principais é que a câmera é colocada perfeitamente centralizada e acima do tabuleiro, conforme mostrado na Figura 3. Nessa posição a detecção das jogadas é mais fácil do que se a câmera estivesse em outro ângulo, já que não é necessário realizar a correção de perspectiva. Além disso, essas imagens contam com condições ideais e constantes de iluminação.

Hirsimäki (2005) analisa fotos de jogos de Go para tentar identificar o estado da partida. São aplicados filtros e a transformada de Hough para encontrar o tabuleiro e suas linhas, e a partir das intersecções encontradas é verificado se cada uma delas está vazia ou contém uma pedra branca ou preta, dependendo da média da intensidade de cores encontrada ao redor. Foram feitos testes com algumas fotos de partidas, onde a maioria teve suas posições identificadas corretamente. Algumas das dificuldades encontradas são discutidas (detecção das linhas quando há muitas pedras, detecção das pedras em condições adversas de iluminação). No entanto, não é feita uma discussão aprofundada dos resultados. O autor apenas diz que o método é “bastante robusto”, sendo assim não é possível comparar os resultados obtidos com outros trabalhos.

Scher, Crabb e Davis (2008) analisam sequências de imagens feitas por câmeras digitais comuns e utilizam um classificador do tipo Hidden Markov Model (HMM) para identificar as sequências de jogadas mais prováveis em partidas. Inicialmente as linhas e colunas do tabuleiro são detectadas usando transformadas de Hough e o algoritmo RANSAC (FISCHLER; BOLLES, 1981), e então as imagens do tabuleiro têm suas perspectivas normalizadas para uma visão ortogonal (como se o tabuleiro estivesse sendo visto de cima). A identificação das jogadas é feita aplicando-se o modelo HMM sobre a sequência de fotos, cada foto sendo fornecida como evidência para o classificador, e selecionando a sequência hipotética de jogadas que tem maior probabilidade de ter ocorrido dadas as evidências.

O método foi testado sobre 4 jogos, onde foi encontrada uma taxa de erro média de 8,65% (cada jogada identificada erroneamente é um erro). No entanto, o método foi testado sobre jogos com poucas jogadas (66 em média), e a maior parte dos erros aconteceu nos jogos mais longos. Partidas de Go usuais têm por volta de 300 jogadas, dessa forma não se sabe se este método é robusto o suficiente para registrar jogos completos.

Seewald (2010) utiliza aprendizado de máquina para registrar jogos a partir de fotos obtidas por câmeras de celulares. Para identificar o estado do tabuleiro foram utilizados métodos de identificação de pontos de interesse sobre as imagens e um classificador Support Vector Machine (SVM) para classificar cada uma das intersecções do tabuleiro como estando vazia ou contendo uma pedra preta ou branca. 32 (72,7%) de 44 imagens de teste tiveram seus jogos identificados sem nenhum erro, e as restantes obtiveram uma média de 1,83 erros (cada pedra identificada incorretamente é um erro).

Esse trabalho mostra que é possível realizar a identificação de jogos a partir de imagens de baixa resolução tiradas por câmeras de dispositivos móveis. Entretanto, ele foi testado com um tabuleiro pequeno (8 linhas por 8 colunas), enquanto partidas de Go normalmente são jogadas em um tabuleiro maior (19 linhas por 19 colunas). Não se sabe se o desempenho do método proposto seria o mesmo ao ser aplicado sobre o tabuleiro maior. Outra limitação é que o método depende de um conjunto de fotos de treinamento. Assim, para aplicá-lo a diferentes tabuleiros e condições de iluminação, provavelmente um

retreinamento seria necessário.

Srisuphab et al. (2012) utilizam sistemas de visão computacional sobre vídeo captado por uma webcam para registrar partidas e as armazena em um banco de dados. Foi utilizada a biblioteca OpenCV para o processamento das imagens e para a detecção dos componentes do jogo. Este é um dos únicos trabalhos, juntamente com Yanai e Hayashiyama (2006), que fazem a análise a partir de vídeos. O método descrito é bastante simples e parece ser possível de ser implementado em um dispositivo móvel. Contudo, sua performance não é discutida no artigo, impossibilitando sua comparação direta com outros trabalhos.

Musil (2014) apresenta um sistema robusto para a identificação automática de jogos de Go a partir de fotos. Entre as técnicas utilizadas estão a aplicação de filtros para detecção de fronteiras nas imagens e transformações de Hough e o algoritmo RANSAC para a identificação das linhas e colunas do tabuleiro. A identificação da cor das pedras é feita calculando-se a média das cores ao redor de cada interseção do tabuleiro e aplicando o algoritmo K-means para clusterizá-las em 3 grupos (interseção vazia, pedra preta ou branca).

O conjunto de dados de teste consistiu de 55 imagens, as quais o método classificou 42 (76,4%) corretamente, 3 (5,5%) com erro na identificação de 1 pedra, 5 (9,1%) com 2 pedras de erro e 5 (9,1%) com 3 pedras de erro. Não houve mais que 3 pedras de erro em nenhuma imagem. A análise de vídeos é sugerida como trabalho futuro, mas especula-se que não haveriam problemas já que um vídeo pode ser tratado como uma sequência de imagens.

Finalmente, Carta e Corsolini (2015) trazem o trabalho mais recente e completo encontrado sobre o tema, apresentando o estado da arte para a identificação automática de partidas de Go a partir de sequências de fotos. Nessa pesquisa é apresentado um método sofisticado que utiliza uma combinação de algoritmos para detectar o tabuleiro e determinar as pedras sobre ele.

Inicialmente, o tabuleiro é encontrado com a detecção de seu reticulado por meio da transformada de Hough linear. À medida que a partida se desenvolve e as pedras passam a esconder as linhas, passa-se a utilizar a transformada de Hough elíptica, que encontra elipses. Utilizando o resultado dessa transformada e as informações de estados anteriores da partida, é possível encontrar o tabuleiro mesmo quando ele se encontra preenchido por pedras. Para determinar a cor de uma nova pedra no tabuleiro são utilizadas diversas *features* visuais como a média de luminância e cor das pedras já jogadas e das intersecções vazias do tabuleiro.

Um dos diferenciais desse trabalho é a detecção de pequenos movimentos na posição do tabuleiro, que podem ocorrer por movimentação da câmera, vibração da mesa,

movimentação dos jogadores, entre outros motivos. Isso é feito através da aplicação de transformadas de Hough lineares ou elípticas sobre a vizinhança de onde se localizavam os cantos do tabuleiro na última imagem analisada. Dessa forma, os cantos podem ser reposicionados corretamente de acordo com a evolução temporal da imagem, mesmo que tenham se deslocado um pouco. Diversas otimizações são feitas para que esses cálculos sejam feitos o mais rápido possível, permitindo o registro de uma partida em tempo real.

Os autores chegaram à conclusão que, quanto maior o tamanho do tabuleiro na imagem (em pixels) e quanto mais o ângulo de visão da câmera se aproxima de 90 graus sobre o tabuleiro (visão ortogonal), melhores são as condições visuais. Nos testes realizados com boas condições de visibilidade do tabuleiro a precisão do sistema (número de jogadas detectadas corretamente dividido pelo número de jogadas totais) fica próxima de 100%. Em condições visuais não ideais, a precisão na detecção de cada jogada individual fica entre 95% e 100%, quando a pedra está completamente visível na imagem, e entre 50% e 60% quando ela está parcialmente visível. Ou seja, em condições visuais ideais, o método apresentado funciona praticamente sem falhas.

2.1 Aplicativos para dispositivos móveis

Existem vários softwares que objetivam realizar a detecção de estados de partidas de Go automaticamente. Porém, nesta revisão serão enfocados apenas os aplicativos para dispositivos móveis que realizam essa tarefa, o que limita bastante a gama de softwares disponíveis.

Um dos aplicativos encontrados para dispositivos Android é o KifuSnap¹, disponível na Google Play. Esse aplicativo permite fazer a análise de pontuação de um tabuleiro de Go sobre fotos tiradas no smartphone, sendo necessário apenas que o usuário faça a marcação manual das posições dos cantos do tabuleiro na imagem. Todo o processamento é realizado no próprio dispositivo.

O Go Scoring Camera², também para Android e disponível na Google Play por cerca de 4,50 dólares, realiza a mesma função que o KifuSnap, mas delega o processamento das imagens para um servidor remoto, necessitando de conexão à internet para ser utilizado.

Já o Baduk Cap³, para iOS, disponível na Apple Store por cerca de 3 dólares, também faz o reconhecimento de uma posição estática de tabuleiro de Go. A detecção do tabuleiro é feita automaticamente, mas é necessário um ajuste manual para o reconhecimento das cores das pedras.

¹ <http://www.remi-coulom.fr/kifu-snap/>

² <https://play.google.com/store/apps/details?id=com.jimrandomh.goscorincamera>

³ <https://itunes.apple.com/us/app/baduk-cap/id896353586>

Outro aplicativo para iOS é o PhotoKifu⁴, disponível na Apple Store por cerca de 5 dólares. Seu funcionamento é similar ao Baduk Cap, com reconhecimento automático do tabuleiro (com possibilidade de ajuste manual das bordas detectadas) e ajuste de cores manual para reconhecimento das pedras.

O último aplicativo pesquisado é o GoEye⁵, também para iOS, um visualizador e editor digital de partidas de Go que também tem a funcionalidade de reconhecer um tabuleiro a partir de fotos. É necessário apenas que o usuário marque os 4 cantos do tabuleiro na imagem. Aparentemente, as pedras são detectadas de forma automática, sem necessidade de intervenção do usuário.

Nenhum desses aplicativos pesquisados dispõe de documentação online, impossibilitando a análise do seu funcionamento em profundidade e comparação com os demais trabalhos relacionados. Ademais, nenhum deles realiza o registro de uma partida de Go completa, limitando-se a reconhecer posições de tabuleiro instantâneas a partir de fotos.

Embora não tenha sido desenvolvido para dispositivos móveis, é importante mencionar o software PhotoKifu⁶ para o sistema operacional Windows (diferente do aplicativo homônimo citado acima para iOS), que é a implementação do método descrito em (CARTA; CORSOLINI, 2015). Esse software é o mais atual e completo que se tem atualmente para registro automático de partidas de Go, tendo sido testado em vários jogos reais, conforme descrito no artigo. É o único software encontrado que realiza o registro de partidas de Go completas.

⁴ <https://itunes.apple.com/us/app/photo-kifu/id894386101?mt=8>

⁵ <http://e-intuit.hk/goeye/>

⁶ <http://www.oipaz.net/PhotoKifu.html>

3 Desenvolvimento

Com base nos trabalhos relacionados no capítulo anterior, foi desenvolvido um método para o registro automático de partidas de Go utilizando dispositivos móveis.

Considerando as principais dificuldades relatadas na revisão bibliográfica, decidiu-se impôr algumas restrições sobre as possibilidades do problema para torná-lo mais simples e tratável no escopo deste trabalho:

- O tabuleiro deve ter linhas bem definidas e com bom contraste;
- O tabuleiro deve começar vazio;
- A câmera e o tabuleiro devem permanecer imóveis durante todo o registro da partida;
- As condições de luz devem ser homogêneas, evitando sombras e reflexos de luz sobre o tabuleiro;
- As jogadas não podem ser feitas com muita rapidez (cerca de uma a cada dois segundos);
- As pedras devem ser jogadas o mais centralizadas possível sobre as interseções.

Sobre a restrição de imobilidade da câmera e do tabuleiro, embora (CARTA; COR-SOLINI, 2015) permita pequenas movimentações em ambos, decidiu-se por não implementar esse mecanismo por razões de simplicidade. Feitas essas considerações, apresenta-se o método de detecção de partidas a seguir.

3.1 Método

O método proposto é dividido em três partes: a **detecção do tabuleiro**, a **detecção das pedras** e a **detecção de jogadas**. Cada etapa é distinta e utiliza como entrada a saída da anterior. Essa divisão foi feita com o objetivo de simplificar o *pipeline* de processamento das imagens e aumentar a precisão final do sistema. Uma visão de alto nível de todas as etapas do método é apresentada na Figura 4.

Conforme descrito por Hirsimäki (2005), uma das dificuldades na detecção do tabuleiro é que se torna cada vez mais difícil fazê-lo quanto mais pedras estiverem sobre ele. Ao separar a detecção do tabuleiro das etapas restantes e fixar sua posição quando este é encontrado, esse problema é eliminado. Outro motivo para a separação das etapas é que o processamento para encontrar o tabuleiro é mais complexo e custoso em termos

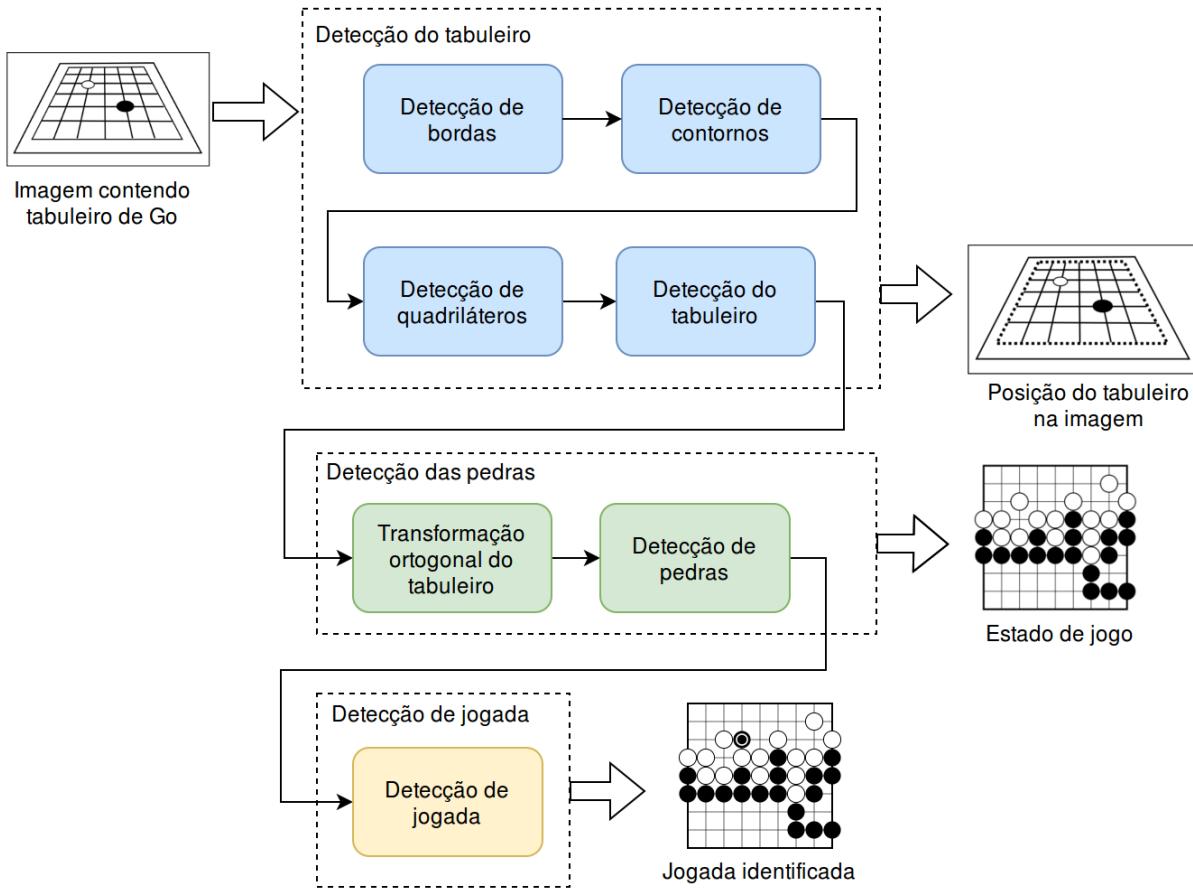


Figura 4 – Visão de alto nível das etapas do processamento

computacionais que o processamento para a detecção das pedras, tornando o método mais leve.

Após a detecção do tabuleiro, realizada uma única vez, as etapas de detecção das pedras e detecção de jogadas são executadas constantemente, a fim de verificar quando uma jogada é feita. Esse processamento é feito até que seja interrompido.

A seguir são detalhadas cada uma das etapas do *pipeline*, usando a foto da Figura 5 para exemplificar o resultado de cada processamento.

3.1.1 Detecção do tabuleiro

A detecção do tabuleiro consiste em determinar se existe um tabuleiro de Go em uma imagem e, caso exista, onde ele está localizado.

Ao contrário da maioria dos trabalhos relacionados, não é utilizada a transformada de Hough (DUDA; HART, 1972) para a detecção de linhas na imagem. Neste trabalho utiliza-se apenas o detector de bordas proposto por Canny (1986) e a detecção de contornos na imagem para encontrar quadriláteros. Essa decisão foi tomada após a verificação de que apenas a detecção de contornos já era suficiente para encontrar o tabuleiro.

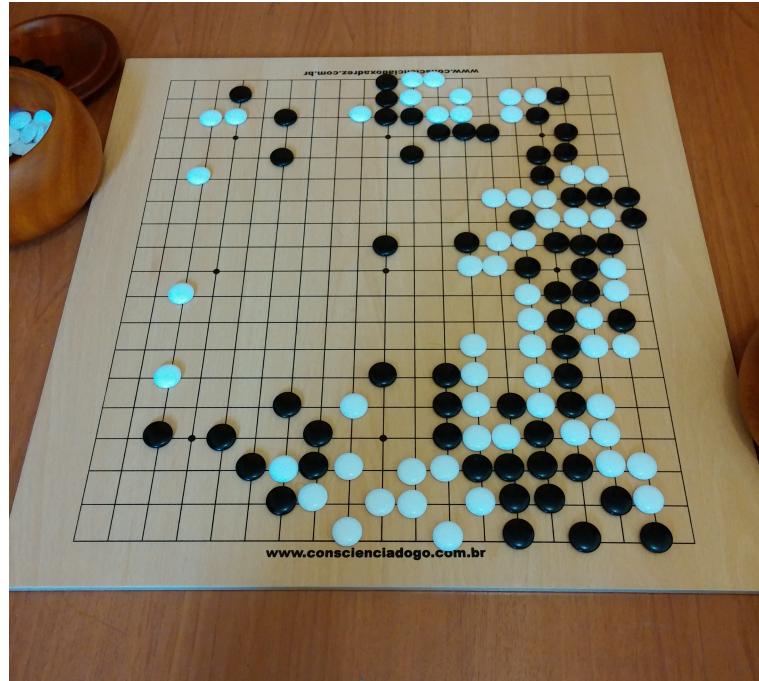


Figura 5 – Foto utilizada nos exemplos de processamento

Esta etapa se divide nos seguintes passos:

1. O primeiro passo é a detecção de bordas na imagem capturada, feita por meio da aplicação do filtro Canny. A detecção de bordas consiste em encontrar e delimitar as fronteiras entre diferentes elementos de uma figura, sendo um dos passos mais comuns em algoritmos de processamento de imagem. A saída do filtro Canny é uma imagem com fundo preto e as fronteiras detectadas em branco. Em seguida, é aplicado o operador de dilatação para expandir as bordas encontradas. Os resultados deste passo são apresentados na Figura 6.
2. Em seguida, a partir das bordas identificadas, é realizada a identificação dos contornos na imagem usando o algoritmo de Suzuki e Abe (1985). Um contorno, neste caso, é definido como qualquer curva fechada. Os contornos encontrados são mostrados na Figura 7.
3. Feito isso, os contornos encontrados são filtrados, mantendo apenas aqueles que são quadriláteros convexos e que tenham uma área de pelo menos 400 pixels. Dessa forma, quadriláteros muito pequenos, que provavelmente não pertencem ao tabuleiro, são descartados. Os quadriláteros encontrados podem ser vistos na Figura 8.
4. Finalmente, os quadriláteros resultantes são organizados em uma hierarquia espacial para determinar quais formas estão dentro de quais. O quadrilátero considerado o contorno do tabuleiro é o menor quadrilátero que contenha pelo menos dez quadriláteros folha dentro de si, ou seja, quadriláteros que não contenham outros. A figura 9

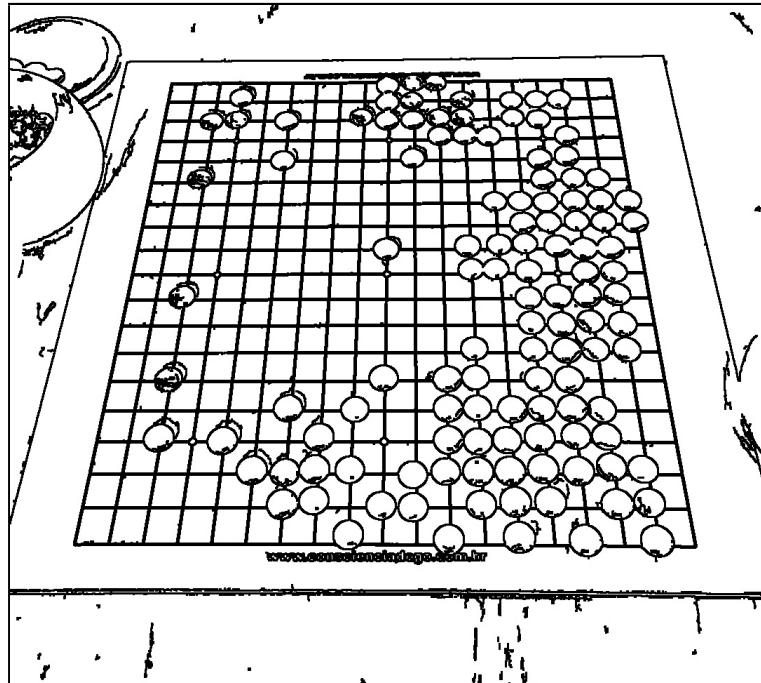


Figura 6 – Foto de exemplo com filtro Camy e dilatação aplicados

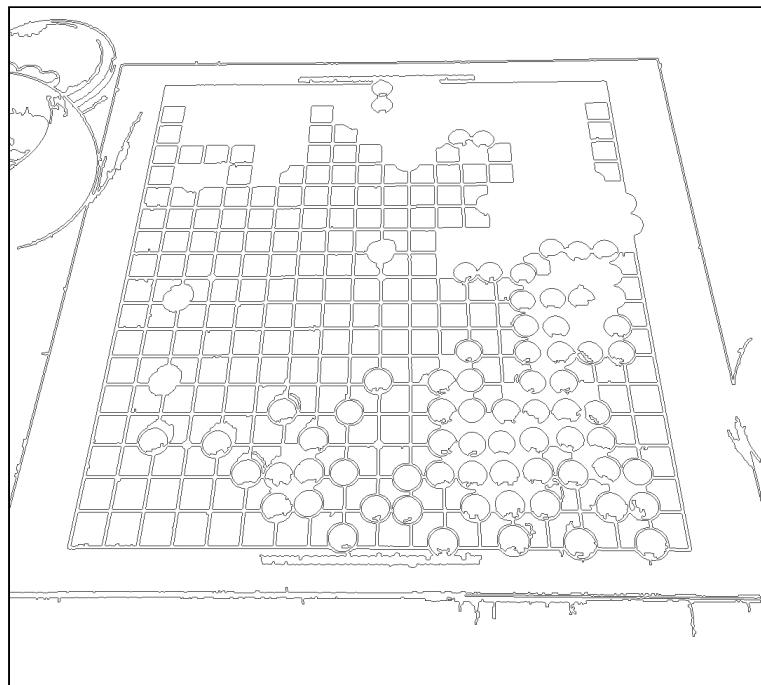


Figura 7 – Imagem anterior com contornos identificados

destaca em azul o contorno do tabuleiro e os quadriláteros restantes em vermelho. A posição do tabuleiro é então passada para a etapa de detecção das pedras.

5. A detecção da dimensão do tabuleiro (9x9, 13x13 ou 19x19) também é feita de forma automática por meio da análise da razão entre o tamanho dos quadriláteros internos e o tamanho do contorno do tabuleiro. Quanto menor for a razão, maior o tabuleiro,

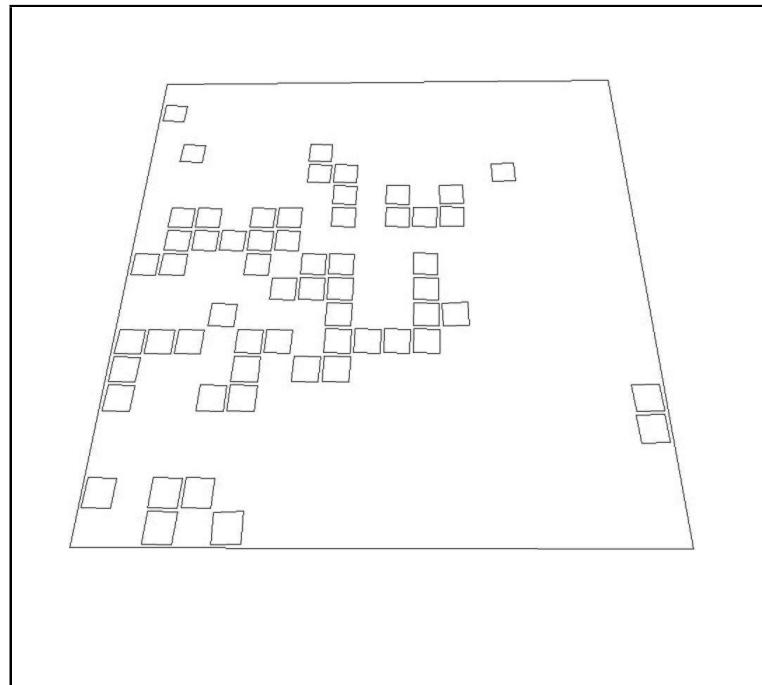


Figura 8 – Imagem anterior com os quadriláteros encontrados

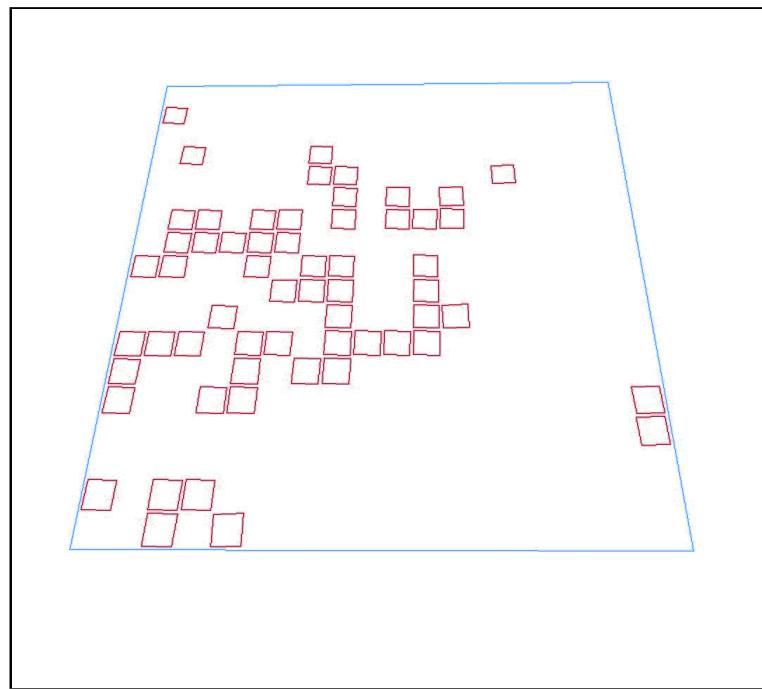


Figura 9 – Imagem anterior com a hierarquia de quadriláteros identificada

já que os quadriláteros internos são menores em relação ao externo. A informação de dimensão do tabuleiro também é passada para a etapa de detecção das pedras.

3.1.2 Detecção das pedras

Depois de o tabuleiro ser devidamente localizado, sua posição não pode mais ser modificada, conforme as restrições apresentadas no início do capítulo. Inicia-se então a etapa de detecção das pedras do tabuleiro, que segue os seguintes passos:

1. A partir da localização do tabuleiro na imagem, é realizada uma transformação que torna a imagem do tabuleiro ortogonal, como se estivesse sendo visto perfeitamente de cima e quadrado, como pode ser visto na Figura 10. Essa transformação também é feita por Scher, Crabb e Davis (2008).

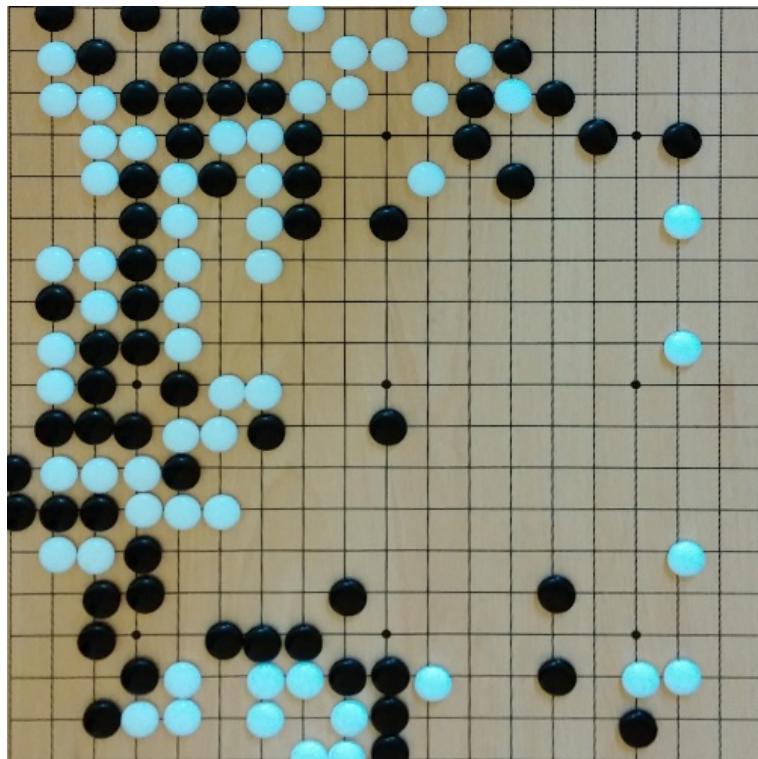


Figura 10 – Tabuleiro transformado para visão ortogonal

2. Tendo a imagem ortogonal do tabuleiro e a informação de seu tamanho, que vêm da etapa anterior, é simples determinar onde se localizam as interseções sobre as quais as pedras são jogadas, já que elas se dispõem de forma regular sobre a superfície do tabuleiro.

A detecção das pedras propriamente dita se baseia nas informações de cor presentes na imagem do tabuleiro, em formato RGB. Esse processo foi feito de duas formas distintas, uma durante a fase de testes com imagens estáticas e outra durante o registro de partidas. Isso acontece pois no registro de partidas a informação visual das pedras que já foram jogadas é utilizada na detecção das novas jogadas. Já nos testes com imagens estáticas não era possível saber de antemão quais posições continham pedras pretas ou brancas, sendo necessária a verificação de todas as

posições individualmente. Além disso, o objetivo da fase de testes foi verificar a precisão e ajustar os parâmetros dos algoritmos de visão computacional.

- a) Durante a fase de testes, quando a detecção foi feita sobre imagens estáticas, a presença de pedras pretas ou brancas sobre cada interseção foi determinada de acordo com limiares. Primeiramente foi calculada a média das cores de uma área circular ao redor de cada interseção, de tamanho aproximado de um terço de uma pedra, conforme indicado na Figura 11.

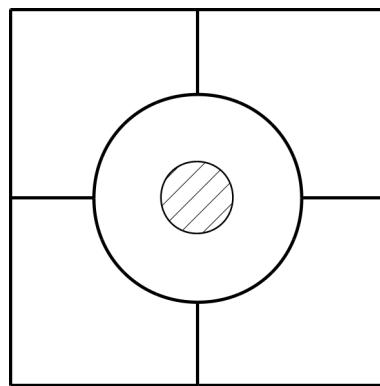


Figura 11 – Região de interesse ao redor de uma interseção

Se a distância (Definição 3.1) dessa cor média para a cor preta¹ for menor que um determinado limiar, considera-se que existe uma pedra preta nessa interseção.

$$\begin{aligned} Distancia(Cor1(R1, G1, B1), Cor2(R2, G2, B2)) = \\ |R1 - R2| + |G1 - G2| + |B1 - B2| \end{aligned} \quad (3.1)$$

A detecção de pedras brancas foi um pouco diferente já que percebeu-se que a componente azul era a maior responsável pelo contraste existente entre a cor de uma pedra branca e a cor de fundo do tabuleiro. Dessa forma, uma pedra branca foi detectada quando a componente azul da média de cores ao redor de uma interseção era maior do que um certo limiar. Se não foi detectada uma pedra preta nem uma pedra branca, foi considerado que essa interseção está vazia.

- b) Já no processamento normal, quando a detecção é feita durante o registro de uma partida, a verificação de novas pedras é feita comparando a cor média de uma interseção com a luminância das interseções vizinhas livres e com a cor das pedras que já foram jogadas.

¹ A cor preta corresponde ao valor RGB (0, 0, 0)

No início do jogo, quando o tabuleiro está vazio, a detecção de pedras é feita verificando-se somente a diferença de luminância (Definição 3.2) de uma determinada interseção com suas posições vizinhas que estão vazias.

$$\text{Luminancia}(\text{Cor}(R, G, B)) = 0.299 * R + 0.587 * G + 0.114 * B \quad (3.2)$$

Se a diferença de luminância é positiva, isso significa que a posição atual é mais clara que as demais e pode conter uma pedra branca. Ao contrário, se a diferença é negativa, isso indica que a interseção pode conter uma pedra preta. Quanto maiores essas diferenças de luminância, é mais provável que realmente exista uma pedra na posição. Ao ultrapassar um certo limiar de confiança, uma pedra é considerada detectada. Se o limiar não for alcançado, não é detectada nenhuma pedra.

A partir do momento em que há pelo menos uma pedra preta e uma pedra branca no tabuleiro, a detecção leva em conta também as cores das pedras já jogadas. São calculados três valores: as cores médias das pedras pretas e brancas que estão no tabuleiro e a cor média ao redor das interseções vazias.

A detecção de pedras procede então da seguinte forma: a diferença de luminância da posição de interesse continua a ser verificada. Se essa diferença não atinge o limiar de confiança, é computada a distância' da cor média dessa região para as três cores médias calculadas. O cálculo dessa distância' (Definição 3.3) envolve, além da distância das cores (Definição 3.1), as diferenças de luminância (Definição 3.2) e variância (entre as três componentes de cor).

$$\begin{aligned} \text{Distancia}'(\text{Cor1}, \text{Cor2}) &= \text{Distancia}(\text{Cor1}, \text{Cor2}) + \\ &| \text{Luminancia}(\text{Cor1}) - \text{Luminancia}(\text{Cor2}) | + \\ &| \text{Variancia}(\text{Cor1}) - \text{Variancia}(\text{Cor2}) | \end{aligned} \quad (3.3)$$

Assim, de acordo com esse cálculo de distância', se a cor da interseção de interesse for mais próxima da cor das pedras pretas, considera-se que há uma peça preta; se for mais próxima da cor das pedras brancas, uma peça branca; e se for mais próxima da cor das interseções vazias, considera-se que não há pedra sobre essa interseção.

No caso de uma pedra preta ou branca ter sido considerada, é feita uma última verificação: se o sistema não tem tanta “certeza” que a interseção contém realmente uma pedra – ou seja, se a distância' dessa interseção para a cor das pedras é próxima da distância' da cor das interseções livres – o aplicativo considera que é uma interseção vazia. Essa decisão foi tomada para diminuir o número de falsos positivos encontrados e aumentar a robustez do aplicativo.

Após a execução desses passos obtém-se um estado de jogo, que corresponde a um tabuleiro com uma determinada configuração de pedras, conforme ilustrado na Figura 12 a seguir.

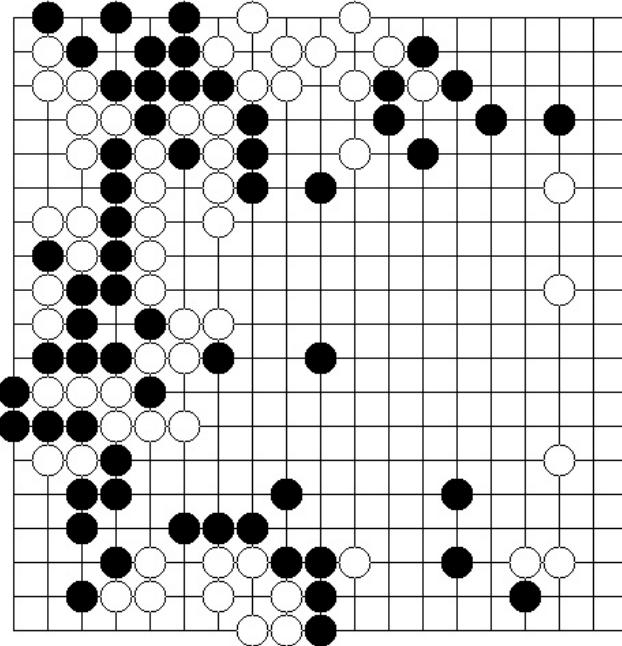


Figura 12 – Tabuleiro identificado

3.1.3 Detecção de jogadas

Finalmente, a partir de uma sequência de estados de tabuleiro é possível realizar a detecção de jogadas.

Essa detecção é feita verificando-se a diferença entre o último estado válido registrado do tabuleiro e o estado atual detectado. Se o tabuleiro atual tem uma pedra preta ou branca a mais que o último tabuleiro registrado, isso pode significar que uma nova jogada foi feita. Contudo, é possível que isso seja resultado de algum erro na detecção das pedras, o que implicaria em um registro de jogada errado.

A detecção errônea das pedras pode ocorrer devido a fatores como sombras, a mão dos jogadores, reflexos de luz ou qualquer outro fator que altere a imagem do tabuleiro. De forma a amenizar esse problema e evitar o registro de jogadas incorretas, foram determinadas duas condições para uma jogada ser considerada: (1) ela deve ser válida segundo as regras do Go, e (2) a disposição de pedras detectada não deve ter sofrido alterações por pelo menos dois segundos. Em outras palavras, para registrar uma jogada o sistema não pode ter visto mudanças na configuração do tabuleiro por pelo menos dois segundos. Além disso, jogadas inválidas, como pedras de mesma cor sendo jogadas em sequência ou grupos deixados sem liberdades, são descartadas.

Essas condições procuram trazer maior robustez e tolerância a erros para o sistema, já que situações errôneas, como uma sombra sobre o tabuleiro sendo detectada como pedras pretas ou a mão de um jogador sendo interpretada como pedras brancas, são descartadas, seja por não resultarem em jogadas válidas ou por configurarem mudanças muito bruscas no tabuleiro. Por outro lado, a restrição de tempo mínimo para uma jogada ser detectada implica em uma maior lentidão no registro do jogo. Por vezes partidas de Go tem jogadas feitas em rápida sucessão, situação que não seria registrada pelo sistema.

Quando uma jogada é considerada válida segundo as duas condições impostas, ela é adicionada ao registro da partida e passa a integrar o último estado de jogo registrado.

Ainda assim, é possível que jogadas espúrias sejam detectadas. Nesses casos, o sistema provê a funcionalidade de voltar a última jogada detectada para efetuar correções manuais.

Conforme explicado anteriormente, a detecção de pedras e de jogadas é feita continuamente até que se decida interromper o processo. Assim, o fim de uma partida não é detectado automaticamente, devendo o registro ser interrompido de forma manual.

3.2 Implementação

O método proposto foi implementado na forma de um aplicativo para o sistema operacional Android, que atualmente detém a maior fatia do mercado mundial de smartphones. Foi focada especificamente a sua versão 4.4.2 (Kitkat). O aplicativo recebeu o nome de **Kifu Recorder**.

Para a implementação dos algoritmos de visão computacional foi utilizada a biblioteca OpenCV²(BRADSKI, 2000) versão 2.4.11. A OpenCV foi escolhida por ser uma das mais utilizadas e robustas bibliotecas de visão computacional de código aberto, trazendo inúmeros algoritmos para processamento de imagens digitais, e por ter recursos prontos para desenvolvimento em Android.

O dispositivo móvel utilizado no desenvolvimento e testes do aplicativo foi um smartphone LG G2, que tem como sistema operacional o Android versão 4.4.2, além de processador quad core de 2.3GHz, 2 gigabytes de memória RAM e câmera de alta definição.

Também é pertinente mencionar que as partidas registradas pelo aplicativo são registradas no formato Smart Game Format (SGF)³, um dos mais utilizados para o registro digital de partidas de Go.

O aplicativo é open-source (licença MIT) e seu código fonte está disponível no endereço <https://github.com/leonardost/kifu-recorder/>.

² <http://opencv.org/>

³ <http://www.red-bean.com/sgf/>

4 Experimentos e Resultados

O método desenvolvido foi avaliado de duas formas: a primeira com experimentos sobre fotos de tabuleiros de Go com diferentes configurações de pedras e a segunda com partidas reais.

4.1 Testes com fotos

A primeira avaliação do sistema foi feita aplicando-se as duas primeiras etapas do método (detecção de tabuleiro e detecção de pedras) sobre fotos de tabuleiros de Go com diferentes configurações de pedras. Os objetivos deste experimento foram medir a precisão da parte de visão computacional do sistema e fazer o ajuste dos parâmetros dos algoritmos utilizados.

As fotos foram tomadas com diferentes características de fonte de luz, quantidade de pedras no tabuleiro e presença de sombras na imagem, de forma a simular as diferentes condições visuais que podem ocorrer durante uma partida real.

Dois exemplos de fotos utilizadas no experimento são apresentadas nas Figuras 13 e 14, juntamente com suas características.

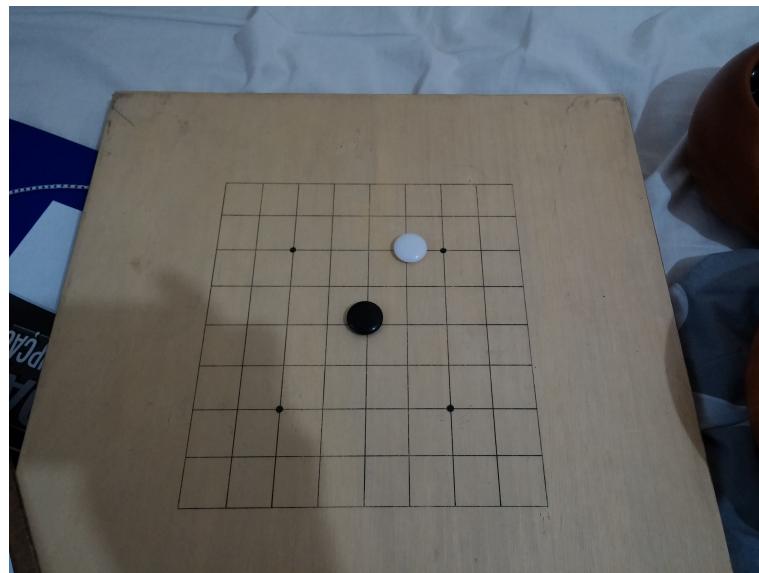


Figura 13 – Exemplo de foto utilizada no primeiro experimento. Características: Fonte de luz branca, duas pedras, há sombras.

Neste teste foi utilizado um conjunto de 66 fotos de tabuleiros. Desse total, 58 fotos tiveram suas pedras detectadas corretamente, resultando em uma precisão de 87,9%. Pode-se observar pelos resultados que um dos principais obstáculos para a correta iden-

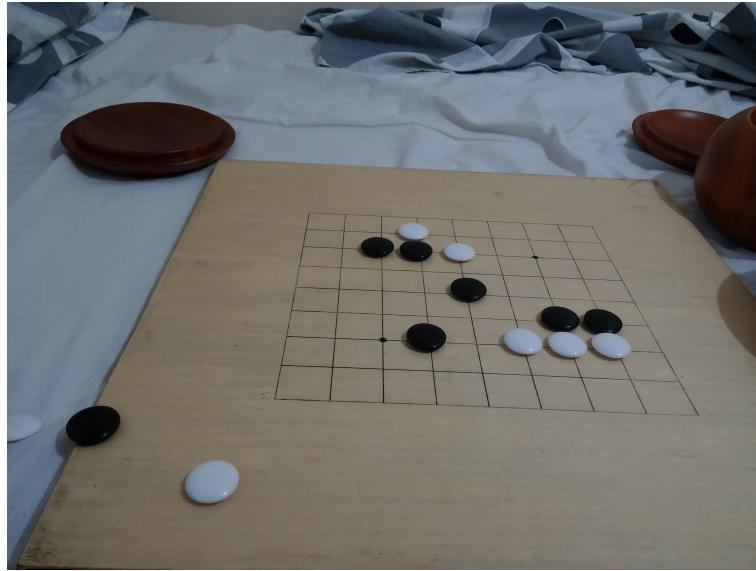


Figura 14 – Outro exemplo de foto utilizada no primeiro experimento. Características: Fonte de luz branca, onze pedras, sem sombras.

tificação das imagens é a presença de luz em excesso ou sombras sobre o tabuleiro, que gera resultados incorretos. O problema das sombras é que, quando são muito escuras, se confundem com possíveis pedras pretas. Já quando há excesso de luz ocorrem pontos de maior luminosidade no tabuleiro, levando o detector a encontrar pedras brancas onde não existem.

Verificou-se também que, quando uma foto com muitas pedras é apresentada ao sistema, especialmente se estiverem sobre as bordas, o tabuleiro não é encontrado pois as peças encobrem boa parte das suas linhas - um dos problemas mencionados por Hirsimäki (2005). Entretanto, como já mencionado no capítulo anterior, o método proposto neste trabalho separa as etapas de detecção do tabuleiro e de detecção das pedras e, além disso, impõe as restrições de que o tabuleiro deve começar vazio e permanecer imóvel. Dessa forma, esse problema ocorre apenas na detecção de pedras em imagens estáticas, já que no registro de uma partida o tabuleiro é detectado antes de as pedras começarem a ser colocadas.

4.2 Experimento com partidas reais

Considerando que nas fotos o resultado da etapa de visão computacional do método foi bom, indicando um desempenho satisfatório do sistema, partiu-se para os testes com registro de partidas reais. Sete partidas entre jogadores do Clube de Go de São Carlos¹ foram registradas. Um tripé com suporte para smartphone foi utilizado para manter a câmera fixa durante o registro das partidas.

¹ <http://www.vamosjogargo.com/>

Todas as sete partidas foram registradas corretamente. Contudo, durante o processo de registro ocorreram alguns problemas: o primeiro foi que quando pedras de material reflexivo (plástico, por exemplo) eram utilizadas e havia uma fonte de luz incidindo diretamente sobre elas, as pedras pretas eram confundidas com pedras brancas pelo sistema devido ao reflexo. A solução para esse problema foi utilizar pedras foscas, que produzem pouca reflexão.

As distorções causadas pela visualização do tabuleiro em perspectiva foram a origem de outro problema. A borda do tabuleiro mais distante da câmera, por sofrer maior distorção, ocasionalmente sofria interferência dos objetos adjacentes ao tabuleiro. Neste caso, a solução foi tentar manter a câmera com um ângulo de visão menos agudo em relação ao tabuleiro, o que diminui bastante esse problema. Uma angulação de ao menos 45 graus (considerando uma visão ortogonal em 90 graus) já foi suficiente.

Movimentos da câmera ou do tabuleiro, mesmo que pequenos, atrapalham a detecção, devendo ser evitados. Um último problema encontrado foi a propensão do detector a encontrar pedras brancas nas posições de borda do tabuleiro. Isso ocorreu principalmente no registro de partidas durante o entardecer, com parte da iluminação proveniente da luz solar. Esse tipo de luz tem maior intensidade que a luz branca artificial, o que leva o detector por vezes a confundir algumas interseções vazias com pedras brancas, conforme havia sido discutido anteriormente.

Os falsos positivos encontrados pelo sistema foram tratados de forma manual, já que o aplicativo provê a funcionalidade de voltar à última jogada registrada para os casos de registro incorreto. Dessa forma, quando uma jogada incorreta era detectada, ela era desfeita e tentava-se registrar a jogada correta novamente. Não houveram falsos negativos já que todas as jogadas corretas das partidas foram detectadas. A Tabela 4.2 a seguir mostra o número de falsos positivos e a precisão² encontrados durante o registro de cada partida.

| Partida | Número de jogadas | Falsos positivos | Precisão |
|--------------|-------------------|------------------|---------------|
| 1 | 37 | 0 | 100% |
| 2 | 48 | 1 | 97,96% |
| 3 | 54 | 0 | 100% |
| 4 | 244 | 11 | 95,69% |
| 5 | 254 | 10 | 96,21% |
| 6 | 267 | 0 | 100% |
| 7 | 237 | 2 | 99,16% |
| Média | 163 | 3,43 | 98,43% |

Tabela 1 – Número de falsos positivos e precisão encontrados em cada registro

Apesar da inconveniência de corrigir manualmente os falsos positivos, o registro das

² A precisão é calculada como o número de jogadas detectadas corretamente dividido pelo número total de jogadas detectadas.

partidas não foi prejudicado. Os registros dessas partidas são apresentados nas Figuras 15 a 20 a seguir.

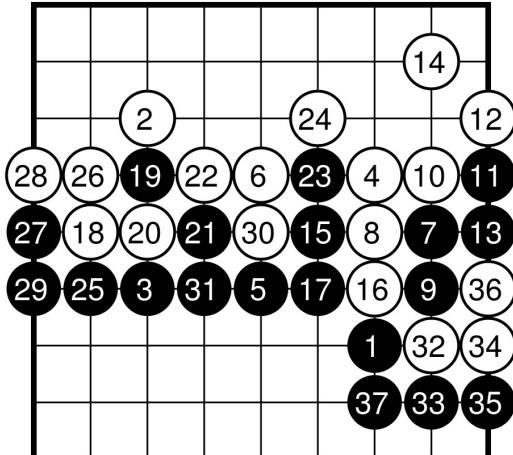


Figura 15 – Primeira partida registrada.

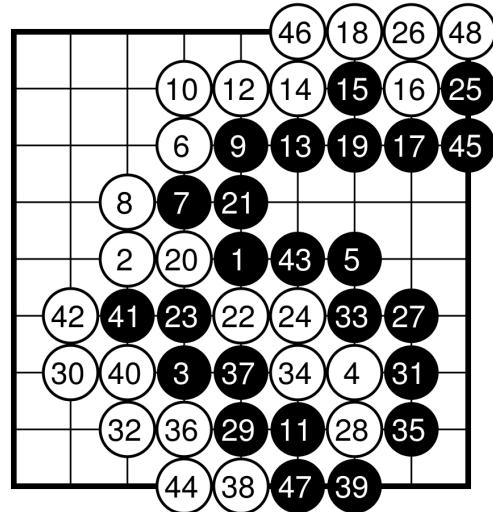


Figura 16 – Segunda partida registrada.

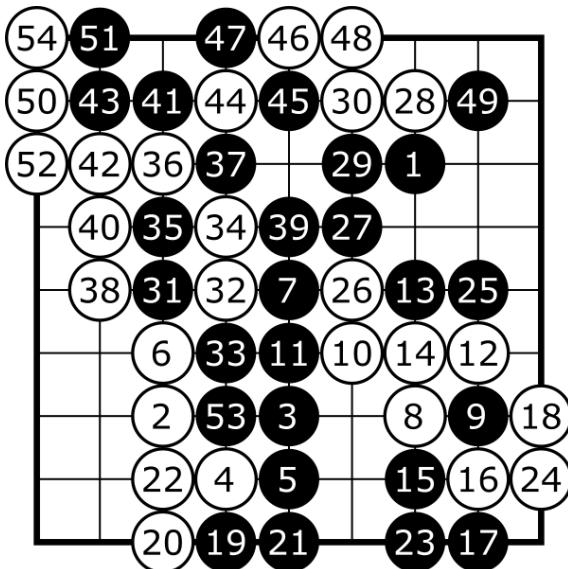


Figura 17 – Terceira partida registrada.

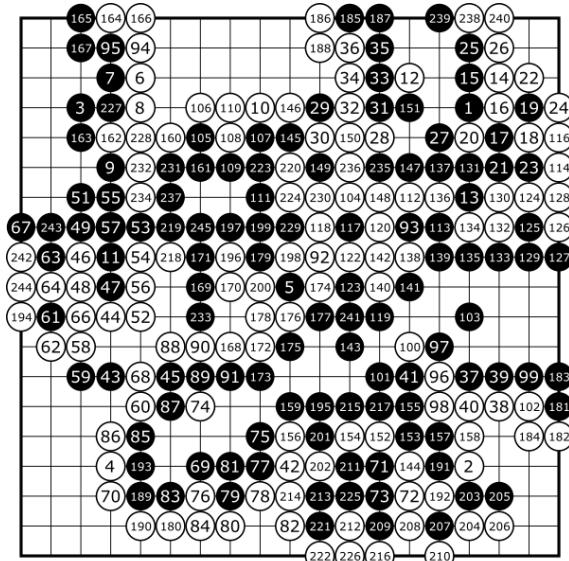


Figura 18 – Quarta partida registrada.

4.3 Discussão

Os testes com fotos apresentaram alguns problemas, particularmente em relação à presença de sombras nas imagens e ao tipo de fonte de luz. Contudo, o registro de partidas se mostrou bastante robusto. Possibilidades de melhoria para as dificuldades encontradas são discutidas no capítulo seguinte.

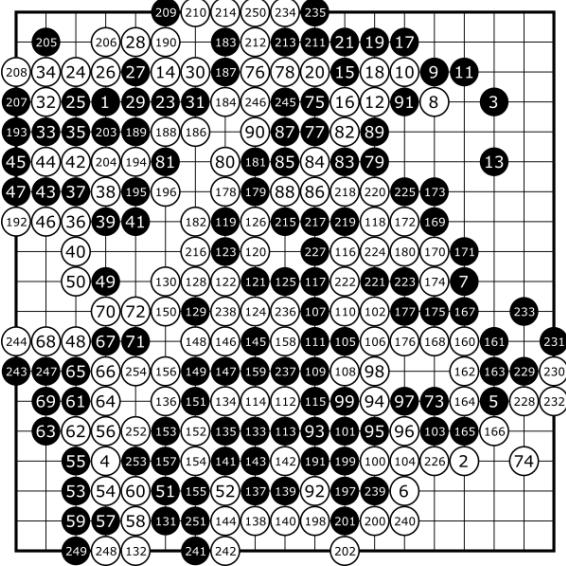


Figura 19 – Quinta partida registrada.

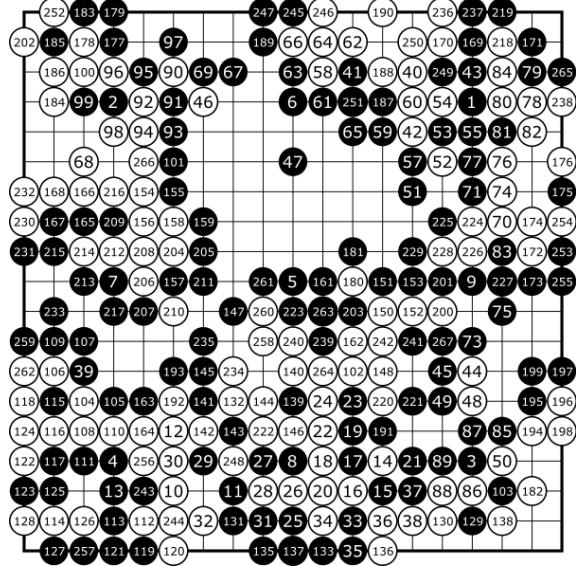


Figura 20 – Sexta partida registrada.

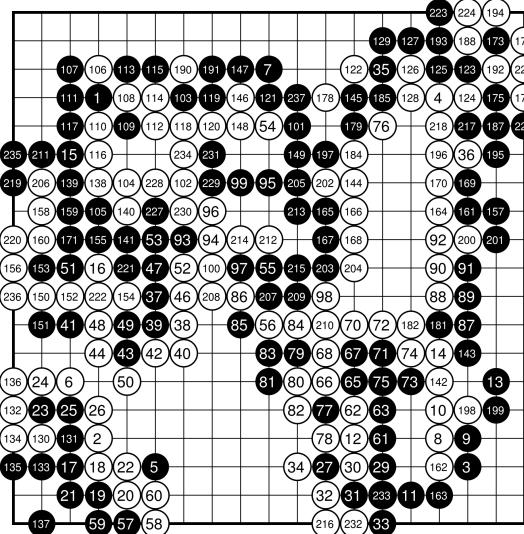


Figura 21 – Sétima partida registrada.

Eventuais erros na etapa de detecção do tabuleiro são mitigados pois o usuário deve confirmar a posição do tabuleiro detectado antes de iniciar o registro da partida. Dessa forma, mesmo que a detecção inicial tenha sido incorreta, o usuário pode mudar a posição da câmera até que o sistema detecte a posição do tabuleiro corretamente.

A etapa de detecção de pedras foi a que mais ocasionou erros, na forma de falsos positivos, especialmente em relação às pedras brancas. A detecção de pedras pretas é mais simples pois o contraste delas com o tabuleiro é alto, independentemente da iluminação. As brancas, contudo, são menos distinguíveis, dependendo das condições de luz do ambiente. Muitas variações na luminosidade sobre o tabuleiro prejudicam a correta detecção das pedras, já que esse processo se baseia em cálculos de médias de cores.

Por sua vez, a detecção de jogadas - considerando que a detecção das pedras tenha sido feita corretamente - ocorreu sem problemas. A restrição de dois segundos para detectar uma jogada poderia ser diminuída para um segundo e meio ou um segundo, mas testes adicionais são necessários para determinar o tempo ideal.

Também é importante ressaltar que diferentes smartphones e tablets podem trazer resultados distintos dos apresentados devido à diversidade de resoluções e qualidades de câmera.

Um último ponto de importância é que o aplicativo faz uso intensivo do processador do smartphone, consumindo bastante bateria e aumentando a temperatura do dispositivo. Melhoramentos no sentido de diminuir o processamento para aumentar a autonomia do aplicativo são possibilidades de trabalhos futuros.

4.3.1 Comparação com trabalhos relacionados

O presente trabalho é o primeiro a fazer o registro de partidas de Go utilizando exclusivamente dispositivos móveis e um dos poucos que realiza o registro completo de uma partida. Além disso, os métodos de avaliação e os dados utilizados nos experimentos são distintos dos demais. Essas diferenças dificultam a comparação deste trabalho com as pesquisas relacionadas citadas.

Contudo, algumas comparações gerais são possíveis. Comparado a Yanai e Hayashiyama (2006), que registra jogos a partir de transmissões de partidas pela televisão, o método proposto neste trabalho é mais versátil e prático, já que a câmera não precisa estar localizada exatamente acima do tabuleiro.

O trabalho atual combina características das pesquisas de Hirsimäki (2005) (transformada de Hough, média de cores ao redor das interseções), Scher, Crabb e Davis (2008) (transformação ortogonal do tabuleiro), Srisuphab et al. (2012) (biblioteca OpenCV, câmera fixa), Musil (2014) e Carta e Corsolini (2015). Uma comparação geral do desempenho dos trabalhos relacionados com a do sistema desenvolvido neste trabalho é apresentada na Tabela 4.3.1 a seguir.

Ao analisar os dados da tabela é importante levar em consideração que foram utilizados diferentes métodos de avaliação e dados em cada trabalho. Além disso, o Kifu Recorder impõe algumas restrições para seu funcionamento que tornam a tarefa de anotação de partidas um pouco mais simples. Porém, em linhas gerais, os resultados apresentados pelo aplicativo são expressivos e comparáveis ao estado da arte (CARTA; CORSOLINI, 2015), considerando que os registros foram completados com sucesso e foram feitos sobre partidas reais, com tabuleiros de tamanho normal (19x19) e utilizando exclusivamente dispositivos móveis.

O Kifu Recorder não pôde ser comparado aos aplicativos pesquisados na Seção 2.1

| Trabalho | Aplicado sobre | Precisão | Precisão calculada sobre |
|-----------------------------|-----------------------|-----------------|---|
| (YANAI; HAYASHIYAMA, 2006) | Jogos televisionados | 95,7% | Jogadas detectadas corretamente |
| (HIRSIMÄKI, 2005) | Fotos | – | – |
| (SCHER; CRABB; DAVIS, 2008) | Sequências de fotos | 91,34% | Jogadas detectadas corretamente em jogos com poucas jogadas |
| (SEEWALD, 2010) | Fotos | 72,7% | Imagens com todas as pedras identificadas corretamente |
| (SRISUPHAB et al., 2012) | Partidas ao vivo | – | – |
| (MUSIL, 2014) | Fotos | 76,4% | Imagens com todas as pedras identificadas corretamente |
| (CARTA; CORSOLINI, 2015) | Sequências de fotos | 95%-100% | Jogadas detectadas corretamente em condições ideais |
| Kifu Recorder | Partidas ao vivo | 98,43% | Jogadas detectadas corretamente em boas condições de iluminação |

Tabela 2 – Comparaçao da precisão dos resultados

pois eles não disponibilizam dados de precisão. Entretanto, é importante mencionar que o Kifu Recorder não necessita de uma etapa de calibração manual da posição do tabuleiro (obrigatória no KifuSnap e GoEye) e nem de calibração de cores, (necessária no Baduk Cap e PhotoKifu).

5 Conclusão

Este trabalho apresentou um método para o registro automático de partidas de Go utilizando a câmera de dispositivos móveis. Um aplicativo que implementa esse método foi desenvolvido para dispositivos Android e testado em imagens e partidas reais. Os resultados obtidos nos experimentos foram bastante positivos, indicando que, com a imposição de algumas restrições, é possível realizar o registro de partidas reais de forma confiável e sem erros.

Uma das principais diferenças deste trabalho em relação aos apresentados na revisão bibliográfica é a separação completa da etapa de detecção de tabuleiro da etapa de detecção das pedras. Os trabalhos relacionados buscam fazer a detecção do tabuleiro e das pedras em conjunto, uma abordagem que se torna mais complicada à medida que o tabuleiro vai sendo preenchido. Por outro lado, a separação da etapa de detecção do tabuleiro das demais também limita a mobilidade da câmera que registra a partida, já que o tabuleiro é detectado uma única vez e não pode mais ser movido.

Outro diferencial do método apresentado é que na detecção de jogadas é feita a verificação da validade da jogada segundo as regras do Go, um fator que traz maior robustez ao sistema ao eliminar jogadas inválidas.

Este é, até onde se pesquisou, o primeiro trabalho que realiza o registro automático de partidas de Go completas utilizando exclusivamente dispositivos móveis, preenchendo uma importante lacuna nessa área de pesquisa.

5.1 Trabalhos futuros

Como já mencionado, uma das limitações do trabalho é que a câmera e o tabuleiro devem permanecer imóveis durante o registro da partida. Qualquer movimento de um dos elementos pode acarretar em erros na detecção. Uma possibilidade de melhoria é implementar a detecção de pequenos deslocamentos de tabuleiro descrita em (CARTA; CORSOLINI, 2015), o que melhoraria bastante a usabilidade do aplicativo.

Outra limitação é que as peças devem ser colocadas de forma bem centralizada sobre as interseções do tabuleiro, caso contrário problemas de detecção das pedras poderão ocorrer. Em jogos reais é comum que algumas pedras acabem ficando um pouco fora do centro das interseções. Uma possibilidade de melhoramento nesse ponto seria realizar a detecção da forma e localização de cada pedra individual, de forma que, mesmo que ela esteja descentralizada, possa ser detectada corretamente. Outra possibilidade seria utilizar técnicas de aprendizado de máquina, como redes neurais, para treinar classificadores

robustos que detectariam pedras mesmo quando estivessem deslocadas.

Um outro ponto importante que traria grandes melhorias ao sistema seria aumentar sua robustez em relação às diferentes condições de luz do ambiente, especialmente na etapa de detecção de pedras. Uma possibilidade seria realizar um processamento de cores mais complexo, como feito por Musil (2014), que aplica *clusterização* sobre as cores da imagem detectada, ou a estratégia de combinação de diversos algoritmos feita por Carta e Corsolini (2015).

Por fim, como o método foi implementado na forma de um aplicativo para smartphone, as questões de uso de processamento e consumo de bateria são relevantes. A versão atual do Kifu Recorder utiliza intensamente o processador, o que aumenta consideravelmente a temperatura do dispositivo e consome bastante bateria. Melhoramentos no sentido de tornar o uso do processador mais eficiente tornariam o aplicativo mais usável.

Referências

- BRADSKI, G. The opencv library. *Dr. Dobb's Journal of Software Tools*, 2000. Citado na página 34.
- CANNY, J. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-8, n. 6, p. 679–698, Nov 1986. ISSN 0162-8828. Citado na página 26.
- CARTA, A.; CORSOLINI, M. A new approach to an old problem: The reconstruction of a go game through a series of photographs. In: *Proceedings of the Second International Go Game Science Conference*. [S.l.: s.n.], 2015. Citado 7 vezes nas páginas 21, 23, 25, 40, 41, 43 e 44.
- DUDA, R. O.; HART, P. E. Use of the hough transformation to detect lines and curves in pictures. *Commun. ACM*, ACM, New York, NY, USA, v. 15, n. 1, p. 11–15, jan. 1972. ISSN 0001-0782. Disponível em: <<http://doi.acm.org/10.1145/361237.361242>>. Citado 2 vezes nas páginas 19 e 26.
- FISCHLER, M. A.; BOLLES, R. C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, ACM, New York, NY, USA, v. 24, n. 6, p. 381–395, jun. 1981. ISSN 0001-0782. Disponível em: <<http://doi.acm.org/10.1145/358669.358692>>. Citado na página 20.
- HIRSIMÄKI, T. Extracting go game positions from photographs. *Helsinki University of Technology, Tech. Rep*, 2005. Citado 5 vezes nas páginas 20, 25, 36, 40 e 41.
- MUSIL, T. Optical game position recognition in the board game of go. 2014. Citado 4 vezes nas páginas 21, 40, 41 e 44.
- SCHER, S.; CRABB, R.; DAVIS, J. Making real games virtual: Tracking board game pieces. In: *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*. [S.l.: s.n.], 2008. p. 1–4. ISSN 1051-4651. Citado 4 vezes nas páginas 20, 30, 40 e 41.
- SEEWALD, A. K. Automatic extraction of go game positions from images: A multi-strategical approach to constrained multi-object recognition. *Applied Artificial Intelligence*, Taylor & Francis, v. 24, n. 3, p. 233–252, 2010. Citado 2 vezes nas páginas 20 e 41.
- SHIBA, K.; MORI, K. Detection of go-board contour in real image using genetic algorithm. In: *SICE 2004 Annual Conference*. [S.l.: s.n.], 2004. v. 3, p. 2754–2759 vol. 3. Citado na página 19.
- SRISUPHAB, A. et al. An application for the game of go: Automatic live go recording and searchable go database. In: *IEEE. TENCON 2012-2012 IEEE Region 10 Conference*. [S.l.], 2012. p. 1–6. ISSN 2159-3442. Citado 4 vezes nas páginas 16, 21, 40 e 41.
- SUZUKI, S.; ABE, K. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, v. 30, n. 1, p. 32–46, 1985. ISSN 0734-189X. Disponível em: <<http://www.sciencedirect.com/science/article/pii/0734189X85900167>>. Citado na página 27.

YANAI, K.; HAYASHIYAMA, T. Automatic "go"record generation from a tv program. In: *Multi-Media Modelling Conference Proceedings, 2006 12th International*. [S.l.: s.n.], 2006. p. 4 pp. Citado 4 vezes nas páginas 19, 21, 40 e 41.