Podstawy Teleinformatyki

*GuziecCheckers* - rozpoznawanie obrazu z gry w warcaby oraz wizualizacja stanu gry na komputerze

Politechnika Poznańska

Informatyka, BSI-1, rok III

Poznań, 11.05.2017

Spis treści

[1. Autorzy projektu 3](#_Toc482196455)

[2. opis i uzasadnienie dlaczego wybrany został dany temat, 3](#_Toc482196456)

[2. podział prac pomiędzy członków zespołu, 3](#_Toc482196457)

[3. wylistowanie i opisanie funkcjonalności oferowanej przez aplikację, 3](#_Toc482196458)

[4. wybrane technologie wraz z uzasadnieniem dlaczego, 3](#_Toc482196459)

[5. architektura rozwiązania (jak jest zbudowane), 3](#_Toc482196460)

[6. interesujące problemy i rozwiązania ich na jakie się natknęliście, 3](#_Toc482196461)

[7. instrukcja użytkowania aplikacji, 4](#_Toc482196462)

# Autorzy projektu

|  |  |  |
| --- | --- | --- |
| Imię i nazwisko | Indeks | Email |
| Tomasz Braczyński | 122053 | tomekbraczynski@o2.pl |
| Daniel Hildebrand | 122048 | daniel20-04@wp.pl |
| Mateusz Korolow | 122030 | mateusz.korolow@gmail.co |

# Opis projektu

# Wybrany przez nas projekt z założenia ma rozpoznawać obraz z gry w warcaby, wizualizować stan bieżącej rozgrywki na komputerze oraz sprawdzać poprawność wykonywanych ruchów. Projekt zakłada umieszczenie przez użytkownika kamery nad planszą do gry podłączonej do komputera. Podsystem do rozpoznawania obrazów wyszukuje pozycje pionków i przekazuje je do podsystemu wizualizacji. Część aplikacji, odpowiadająca za wizualizację, przedstawia planszę wraz z umieszczonymi na niej pionkami będące odzwierciedleniem rzeczywistości. Użytkownikowi ma możliwość wybrania ruchu zgodnego z zasadami gry oraz wizualizacji tego ruchu w aplikacji.

1. Uzasadnienie wyboru tematu

Temat ten wybraliśmy z kilku powodów, pierwszym z nich jest chęć przetestowania systemu wizyjnego w oparciu o bibliotekę EmguCV, a był to jedyny przedmiot, który pozwalał nam na takie przedsięwzięcie. Chcemy również stworzyć aplikację, która pomogłaby niedoświadczonym graczom rozwinąć swoje umiejętności poprzez sprawdzanie poprawności wykonywanych ruchów. Ostatnim z powodów jest fakt, że lubimy warcaby.

# Podział prac pomiędzy członków zespołu

1. Projektowanie interfejsu – cała grupa
2. Implementacja interfejsu graficznego – D. Hildebrand
3. Analiza poprawności wykonywanych ruchów – T. Braczyński
4. Obsługa systemu wizyjnego OpenCV – M. Korolow

# Wymagania

# Funkcjonalne

* 1. Pozafunkcjonalne
     1. System operacyjny Windows Vista i nowszy.
     2. Zewnętrzna kamera o rozdzielczości minimum FHD lub smartfon wraz z zainstalowaną aplikacją DroidCam.
     3. Minimum 512 MB wolnej pamięci operacyjnej.
     4. Do zachowania płynności przetwarzania obrazu wymagany jest dwurdzeniowy, wielowątkowy procesor o taktowaniu minimum 2 GHz.
     5. Szachownica o wymiarach 10x10.
     6. 24 piony, po 12 na każdego uczestnika rozgrywki znacząco różniące się kolorami. Piony te muszą posiadać mniejszą średnicę niż bok pojedynczego pola planszy.

# Wybrane technologie W tej sekcji znajdują się wybrane przez nas technologie użyte do realizacji projektu wraz z wyjaśnieniem naszego wyboru. Są to:

1. Obiektowy język programowania C# został wybrany, ponieważ jest opanowany przez nasz zespół w największym stopniu.
2. Silnik graficzny WPF (Windows Presentation Foundation) został użyty ze względu na wcześniej wybrany język programowania – C#.,
3. Wrapper EmguCV bazujący na bibliotece OpenCV został użyty do obsługi kamery. Do tego wyboru skłoniła nas duża ilość dostępnych materiałów elearningowych pozwalających na lepsze opanowanie przechwytywania obrazu z kamery.
4. .NET Framework w wersji 4.5.2 został wybrany ponieważ jest w pełni kompatybilny z wrapperem EmguCV.

# architektura rozwiązania (jak jest zbudowane),

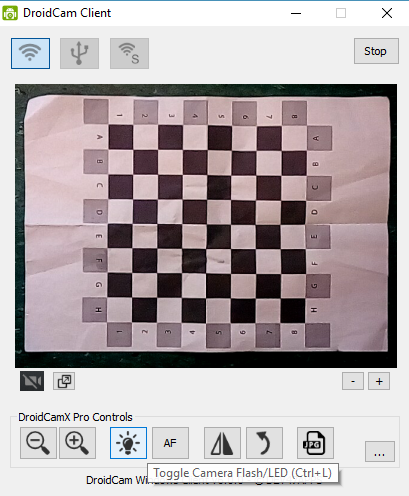
# Wybrane problemy i ich rozwiązania

1. Sposób kalibracji:
   * 1. Specjalnie przygotowana szachownica o wymiarach 10x10 (wymiary użytkowe 8x8) wykorzystywana przez metodę FindChessBoardCorners pozwala na bezproblemowe znalezienie (punktów) narożników wszystkich pól użytkowych.
     2. Na podstawie wektora punktów otrzymanego w poprzednim punkcie, tworzone są instancje struktury Field reprezentującej pojedyncze pole szachownicy. Instancje te są dodawane do listy pól będącej składową klasy ChessBoard.
     3. Na podstawie danych uzyskanych na etapie kalibracji znajdywane są okręgi reprezentujące piony (znajdujące się na szachownicy) poszczególnych graczy.
     4. Dla każdego ze znalezionych pionów odnajdywane jest pole, do którego przynależy, a informacja ta jest przechowywana w obiekcie tego pola.

# Instrukcja użytkowania aplikacji

W celu poprawnego rozpoczęcia rozgrywki używając naszej aplikacji należy wykonać poniższe kroki:

* 1. Przygotować odpowiednio spreparowaną szachownicę o wymiarach 10x10 przedstawioną na rys, (WSTAWIĆ NUMER RYSKUNKU PRZEDSTAWIAJĄCY PLANSZĘ). Dodatkowe pola są używane przez program jako markery by w łatwiejszy oraz szybszy sposób odnaleźć planszę.
  2. Podłączyć kamerę do komputera i umiejscowić ją nad polem rozgrywki. Najlepszą pozycją będzie ustawienie kamery nad środkowym polem tak by obiektyw kamery skierowany był na dół. Aplikacja wymaga, aby pole A1 znajdowało się w lewym górnym rogu.   
     Wskazówka: polecamy użycie aplikacji DroidCam (Rysunek 1.) umożliwiającej użycie kamery zamontowanej w smartfonie jako kamery wykorzystywanej w aplikacji.



Rysunek 1 – Interfejs aplikacji DroidCam Client udostępniającej obraz z kamery smartfona. Na tym zrzucie ekranu możemy również zobaczyć odpowiednie ustawienie kamery oraz planszy do gry w warcaby.

* 1. Uruchomić aplikację.
  2. Po uruchomieniu aplikacji należy ją skalibrować postępując zgodnie z poniższymi krokami:
     1. Ustawić piony pierwszego gracza na pozycjach A1, B8, E5,G1,H8.
     2. Ustawić piony drugiego gracza na pozycjach A7, B2, D4, G7, H2.
     3. Manipulując wartościami liczbowymi odpowiadającymi składowym systemu barw RGB należy je zmienić tak, aby odpowiadały one kolorom pionów danego gracza.
     4. Zatwierdzić kalibrację poprzez naciśnięcie przycisku „zapisz konfigurację”.
  3. Rozpocząć rozgrywkę, którą można obserwować na ekranie komputera, dodatkowo można wyświetlać podpowiedzi prawidłowych ruchów.

1. Mockupy

**dokumentacja: około 30 stron, czcionka 12 pt,**

**pojedyncze odstępy między wierszami powinna zawierać:**

**w przypadku umieszczania obrazków w dokumentacji należy do każdego z nich dodać co najmniej jeden akapit (4-5 zdań opisu),**

**nie należy umieszczać kodu źródłowego w dokumentacji, co najwyżej krótkie, charakterystyczne fragmenty (powiedzmy do 30 linii kodu),**