Laboratorium
Widzenie Maszynowe

Card detector
- program do rozpoznawania kart do gry

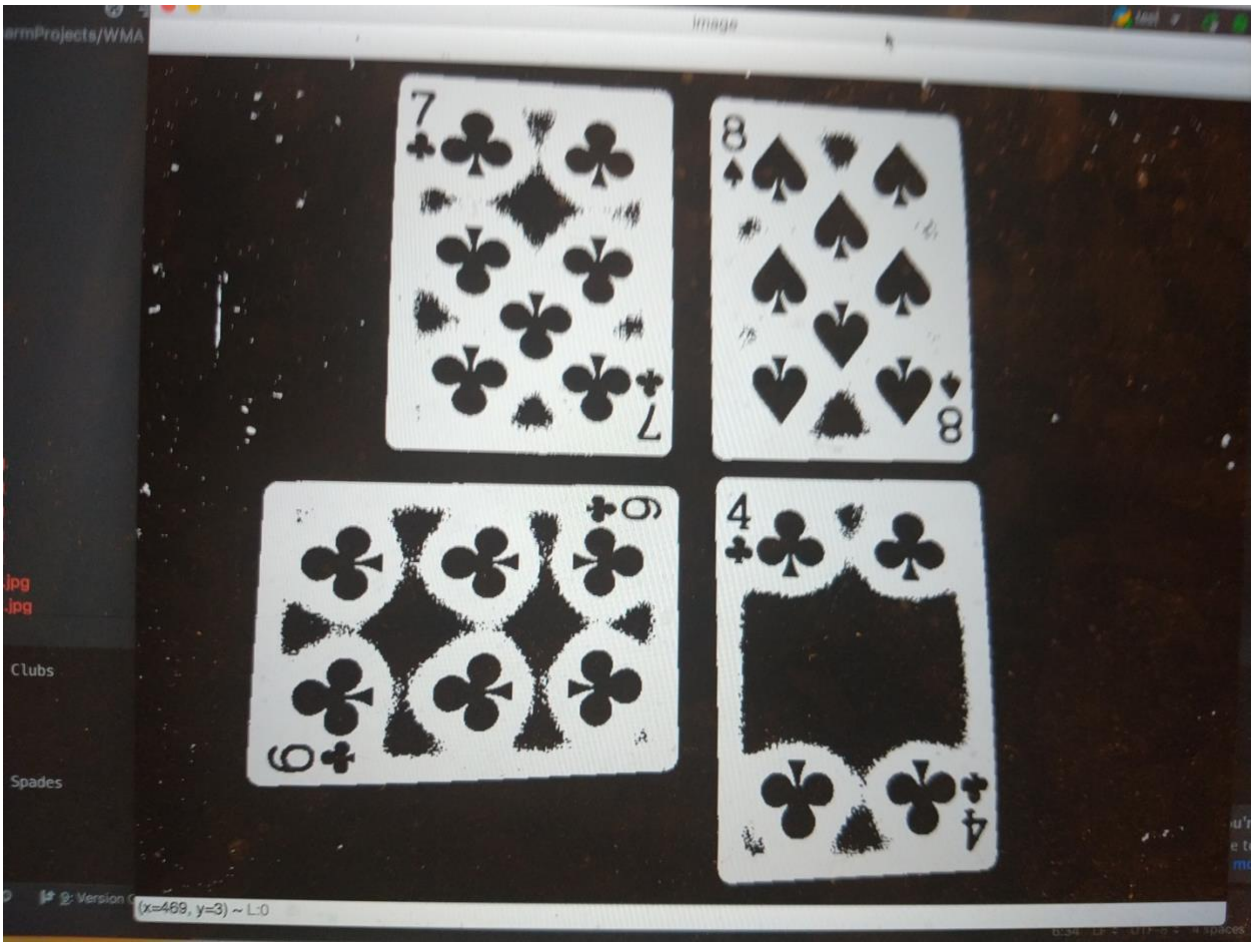
Konstanty Sajnaga gr. 33ip
284248

1. **Wstęp**

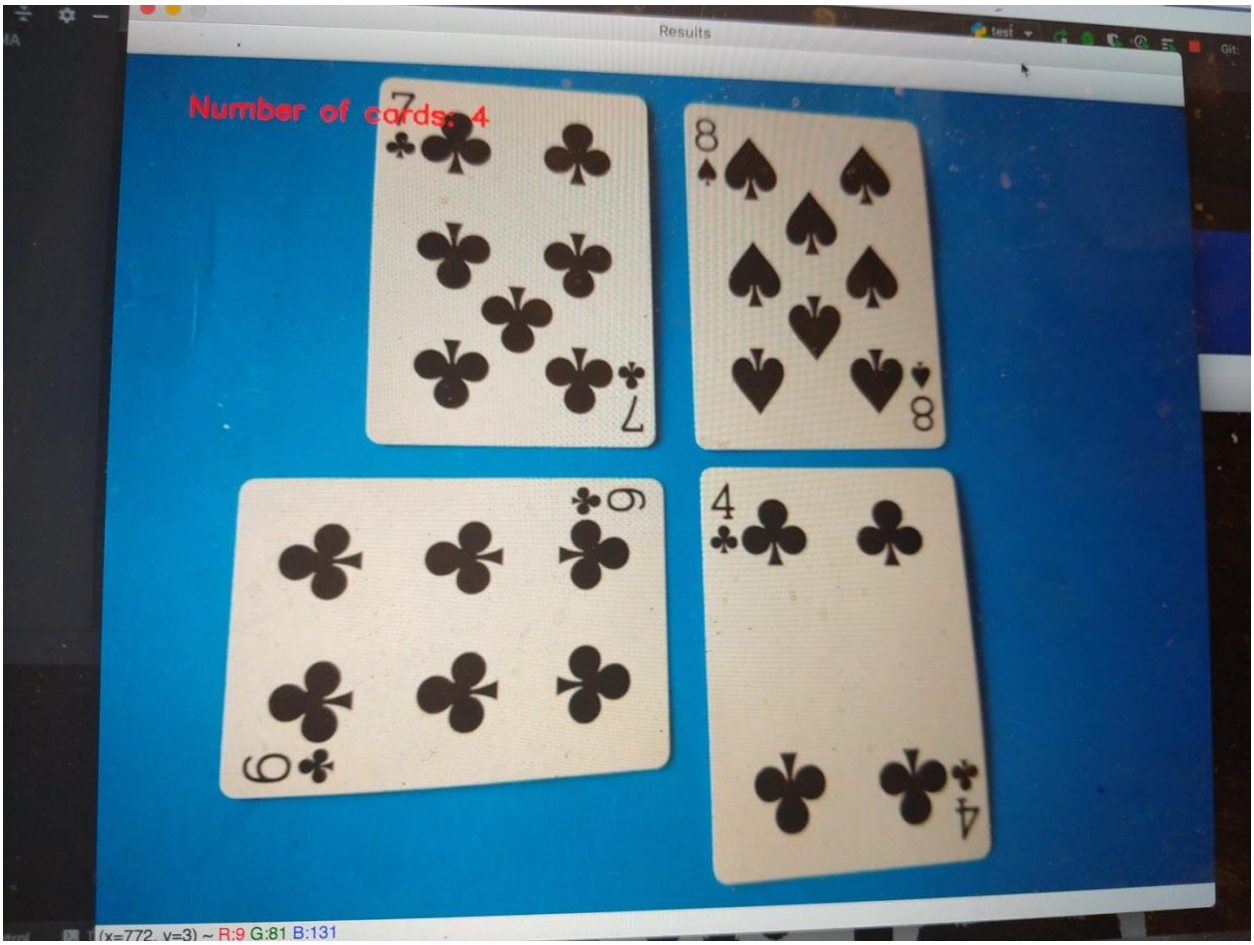
Celem programu jest rozpoznawanie rodzaju oraz koloru dowolnej spośród 52 kart do gry. Program został zrealizowany przy użyciu języka Python oraz biblioteki funkcji wykorzystywanych podczas obróbki obrazu – OpenCV. Środowisko, w którym pracowałem to PyCharm na systemie Mac OSX.

2. Przebieg prac nad projektem

Pierwszym etapem projektu było zrealizowanie możliwości wyszukania kart na zdjęciu oraz ich zliczenia. W tym celu, po wczytaniu zdjęcia dokonałem *binaryzacji* z użyciem filtru Gaussa i wartości parametrów dobranych doświadczalnie (*Zdj. 1*). Po dokonaniu *binaryzacji*, aby zliczyć karty użyłem funkcji do znajdywania konturów *findContours()* i ograniczyłem badanie znalezionych konturów jedynie do takich, które mają pole większe od minimalnego, określonego przeze mnie na bazie doświadczeń, zakładając, że na zdjęciu nie będzie więcej niż 6 kart. Następnie przy pomocy funkcji *putText(),* wypisałem liczbę kart na ekranie z oryginalnym zdjęciem kart (*Zdj. 2*).

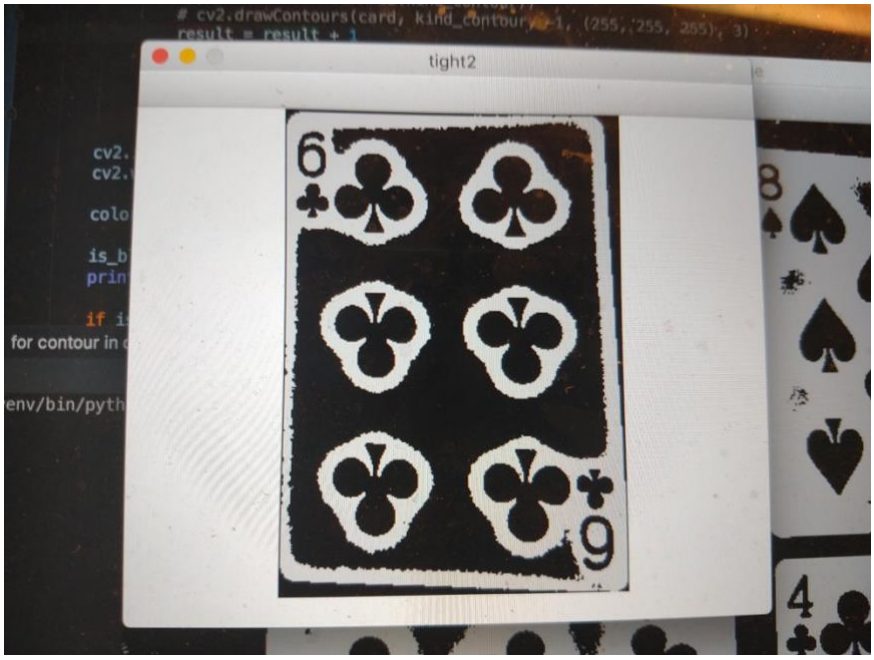


Zdjęcie 1. *Obraz kart po binaryzacji*



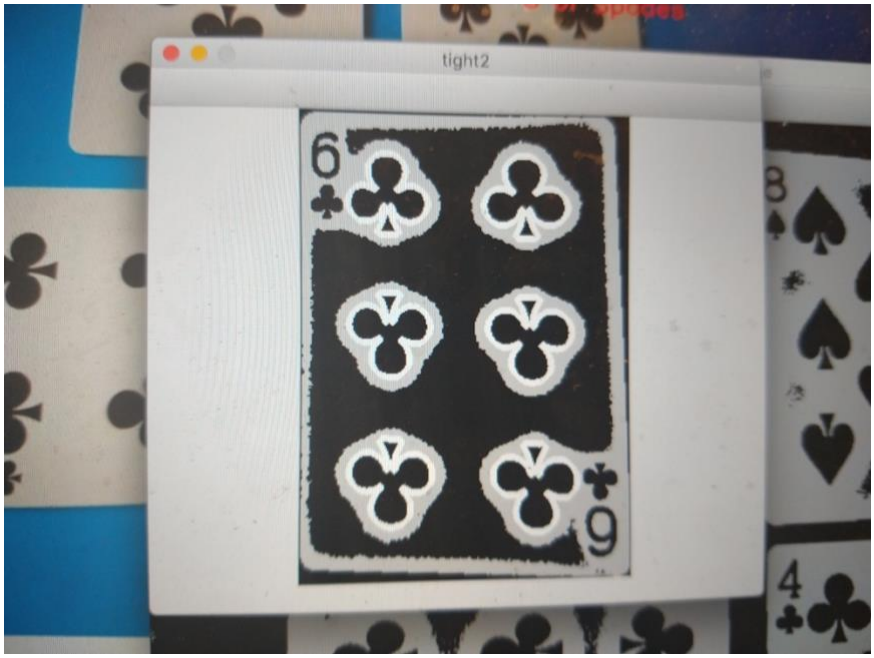
Zdjęcie 2. *Obliczona ilość kart na zdjęciu*

Kolejnym wyzwaniem było wycięcie poszczególnych kart celem późniejszych obróbek. Początkowo chciałem wykorzystać przybliżenie prostokątami przy użyciu funkcji *boundingRect(),* co nie okazało dać większej korzyści. Na początku próbowałem użyć naroży otrzymanych w ten sposób prostokątów obramowujących każdą z kart, jednak bardziej efektywnym rozwiązaniem okazało się użycie funkcji *minAreaRect(),* która zwraca naroża prostokątów usytuowanych niekoniecznie w pionie. Następnie, aby wyegzekwować pojedynczą kartę ze zdjęcia, użyłem funkcji *getPerspectiveTransform()* wyznaczającej perspektywiczną transformatę przy użyciu czterech par powiązanych ze sobą punktów (wierzchołków karty) oraz użyłem funkcji *warpPerspective()* przenoszącej otrzymaną macierz nałożoną na zbinaryzowane zdjęcie do nowo utworzonego pliku o określonej wysokości i szerokości. Jako pierwszy argument funkcji *getPerspectiveTransform()* podałem punkty, będące współrzędnymi naroży karty, wyznaczonymi w odpowiednim porządku przy użyciu napisanej przeze mnie funkcji *get_right_points_order*. Szacuje ona również wysokość i szerokość obrazu poszczególnych kart i w razie potrzeby dopasowuje kartę do pionu, jeśli jest usytuowana poziomo. Wynikowe zdjęcie karty przedstawia *Zdjęcie 3*.

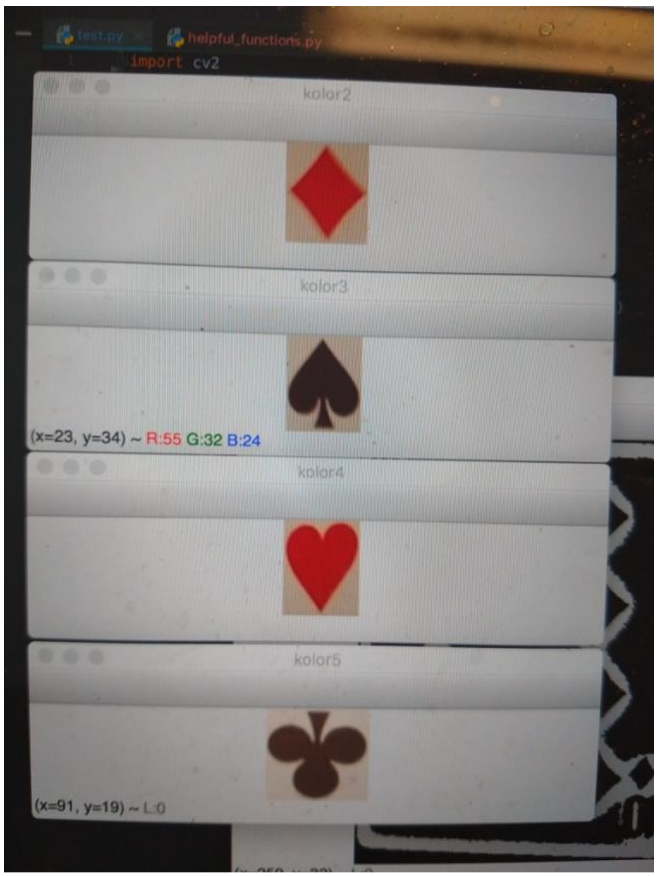


Zdjęcie 3. *Wycięte, zdjęcie danej karty*

Po uzyskaniu oddzielnego obrazu z daną kartą ponownie użyłem funkcji *findContours()*, tym razem do znalezienia liczby ‘znaczków’ na karcie celem określenia rodzaju karty, np. dwójka. Ustaliłem doświadczalnie przedział, w którym znajdują się pola ‘znaczków’ i przy pomocy funkcji *contourArea()* zliczyłem ich liczbę dla każdej karty. Na *zdjęciu 4* zaznaczone są interesujące mnie kontury przykładowej karty. Przedstawione zostały przy użyciu funkcji *drawContours().*



Zdjęcie 4. *Zdjęcie danej karty z zaznaczonymi konturami ‘znaczków’*

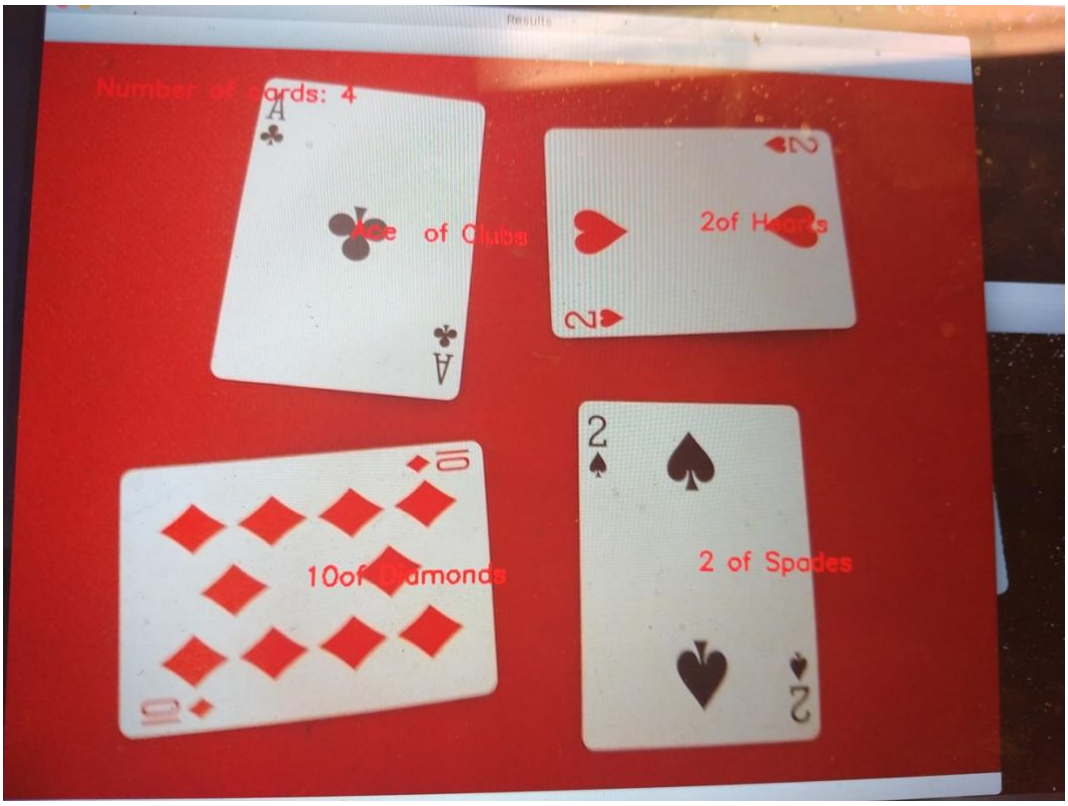


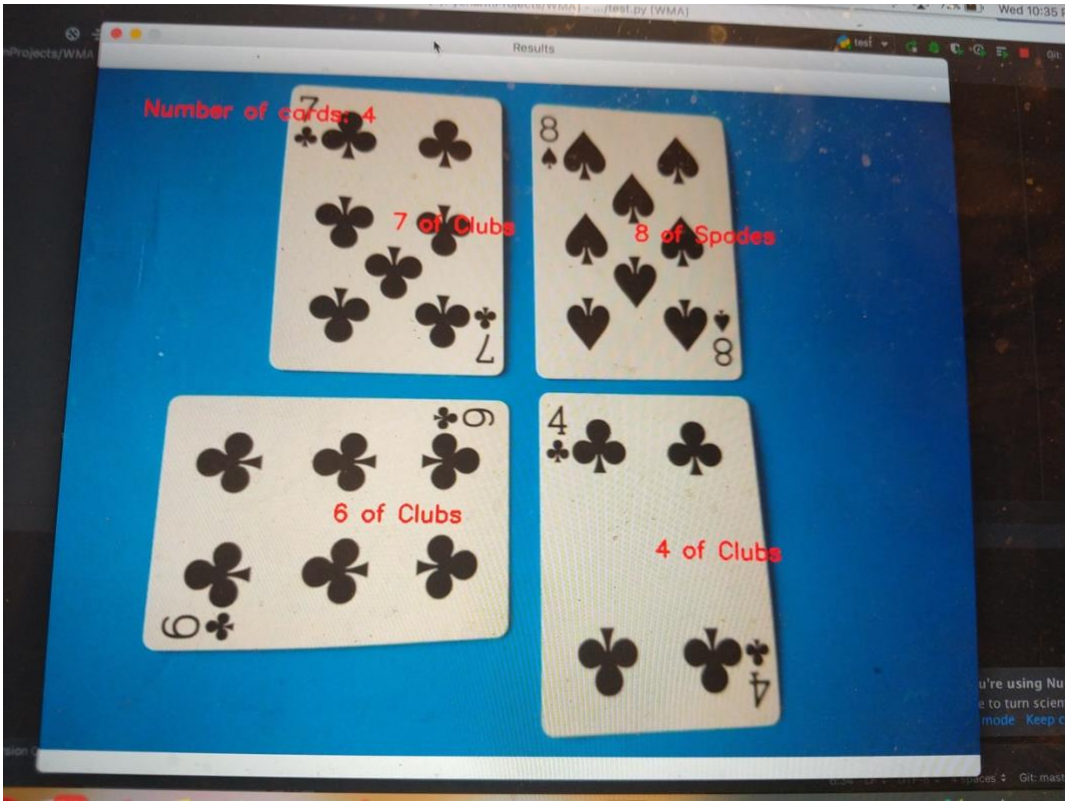
Zdjęcie 5. *Zestawienie 'znaczków'*

Kolejną rzeczą było odróżnienie kolorów kart. Najpierw przy użyciu napisanej przeze mnie funkcji *check_card_shape_if_is_black()* na podstawie środkowego piksela i określonej wartości zapisuje do flagi *is_black* True, jeśli karta jest koloru czarnego. Następnie jeśli jest to wywoływana jest funkcja *check_if_card_is_spades_or_clubes()* bazująca na proporcji szerokości do wysokości 'znaczka'. Od określonej granicy zwraca 'clubs' a w przeciwnym wypadku 'spades'. Dla kart czerwonych sprawdzałem wartości piksela (0.25 * w, 0.2 * h) oraz (0.25 * w, 0.8 * h), gdzie h – wysokość obrazu ze 'znaczkiem', a w – jego szerokość. Jeśli obie wartości są większe niż 100, czyli nie ma tam 'znaczka' – to zwraca 'diamonds' (różnica kształtów), a w przeciwnym wypadku 'hearts'. Trzeba wspomnieć, że te obrazy z czerwonymi 'znaczkami' wcześniej binaryzuję (po ustaleniu ze są one czerwone). Zdjęcie 5 przedstawia zestawienie wyciętych do oddzielnych obrazów znaczków.

Warto wspomnieć, że inna koncepcja do odróżniania rodzaju karty była różnica absolutna. Praca nad ta koncepcja okazała się pracochłonna i nieefektywna – błędy w odróżnianiu kart, licznymi operacjami na obrazach. Poza tym wymaga zestawu plików z wzorcowymi kartami. Poza tym próbowałem również użyć algorytmu Canny, jednak bez użycia odpowiednich operacji morfologicznych nie udało mi się liczyć pól konturów o odpowiednich wymiarach, ze względu na fakt ze zwracane kontury nie były zamknięte.

Rezultaty dla trzech przykładowych zdjęć znajdują się na zdjęciach 6 i 7.





Zdjęcie 6 i 7. *Wyniki programu*