# Sprawozdanie z realizacji projektu

# "Liczenie sum pieniędzy widocznych na zdjęciu"

Szymon Sroka 141312, Weronika Radzi 141303

Listopad 2020

# 1 Wstęp. Informacje o temacie projektu

Naszym zadaniem było stworzenie programu, który policzy sumy pieniędzy widoczne na zdjęciach. W projektcie skupiliśmy się na detekcji monet 1gr, 10gr, 1zł, 2zł i 5zł.

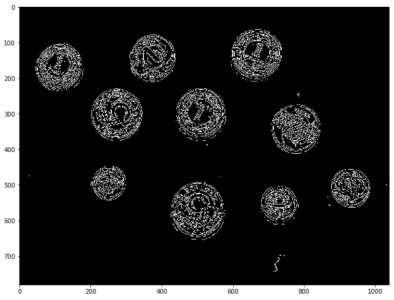
# 2 Sposób działania programu

Na początku wczytujemy zdjęcie i dokonujemy wstępnych przekształceń, przygotowujących obraz do wykrywania krawędzi. Wśród użytych na tym etapie przekształceń możemy wymienić

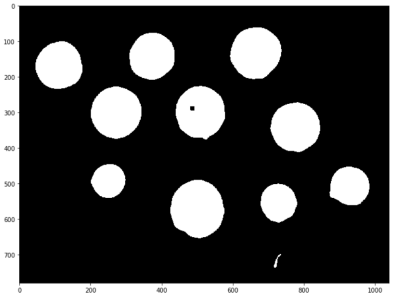
- konwersję do odcieni szarości
- filtr Canny (pozwalający na detekcję krawędzi)
- dylatację (która pozwoliła nam na rozszerzenie wykrytych konturów i w połączeniu z erozją - na ich zamknięcie)
- filtr medianowy (do usunięcia zakłóceń)
- erozję (do zniwelowania nadmiarów powstałych w wyniku dylatacji)
- binary_fill_holes, odpowiadającą za wypełnienie przestrzeni (dosł. "dziur") wewnątrz kształtu, który powstał w wyniku wcześniej wymienionych przekształceń.
- niewielkie zwiększenie temperatury kolorów na zdjęciu (uzyskane poprzez zmianę wartości składowych RGB pikseli; wiele z naszych zdjęć było wykonane przy sztucznym, "zimnym" świetle i zaszła potrzeba ich ocieplenia w celu lepszego rozpoznawania kolorów w późniejszym etapie)

Rysunek 1: Zdjęcie oryginalne

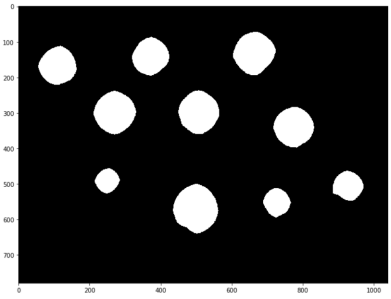
Rysunek 2: Zdjęcie w odcieniach szarości



Rysunek 3: Zdjęcie po zastosowaniu *filtru Canny*



Rysunek 4: Zdjęcie po zastosowaniu dylatacji, filtru medianowego i erozji



Rysunek 5: Zdjęcie po ponownej dylatacji, użyciu funkcji *fill_holes* oraz erozji

W następnym kroku **wykrywamy kontury** przy pomocy funkcji *find_contours* i przechodzimy do etapu zbierania informacji o pikselach znajdujących się wewnątrz konturu; wyznaczamy środek i skrajne punkty każdego konturu ("skrajny" lewy, prawy, górny i dolny piksel) i na tej podstawie **wyznaczamy średnią wielkość promienia koła**, który tworzy moneta. Korzystając z nierówności koła i biorąc pod uwagę wyznaczone wielkości **iterujemy po punktach należących do monety**.

Zauważyliśmy, że w modelu barw RGB odcienie koloru szarego mają zawsze zbliżone do siebie wartości składowych R, G i B (np. [50,50,50], [180,180,180]) Na podstawie tej informacji dla każdego konturu wyznaczamy stosunek żółtego i szarego koloru w całej monecie oraz stosunek tych kolorów w obrębie koła tworzonego w obrębie połowy promienia monety wraz z średnią jansność pikseli w tym obrębie. Obliczone wielkości przechowujemy w obiektach klasy *Moneta()*.

W kolejnym kroku **analizujemy różnice między promieniami monet**. Ta analiza pozwala nam wykryć sytuację, gdy na stole znajdują się monety o jednakowej wielkości (tzn. takie z zestawu 1gr, 10gr lub 1zł, 2zł, 5zł). Zaimplementowaliśmy takie rozwiązanie w celu zminimalizowania szansy popełnienia błędu przy rozpoznawaniu monet 10gr/1zł, które są w 100 procentach szare. Bardziej szczegółowo wspomniany problem i rozwiązanie opisaliśmy w dalszej części sprawozdania.

W następnym kroku na podstawie zebranych informacji, kierując się zasadą drzewa decyzyjnego **przypisujemy monetom odpowiednie etykiety**, to znaczy gdy na przykład w danej monecie jest 60 procent koloru żółtego i 40 szarego oraz środek obszar w obrębie koła o promieniu równym połoweie promienia monety ma przeważającą zawartość koloru szarego, to program przydziela takiej monecie etykietę "5zł".

W trakcie pisania programu pojawił się problem, **jak rozróżnić monety 1zł i 10gr**. Problem rozwiązaliśmy w następujący sposób: program wykrywa sytuację, gdy na stole są monety podobnych wielkości (same 'małe' lub same 'duże'). Gdy zachodzi taka sytuacja, program sprawdza, czy wśród monet rozpoznano monetę 2 lub 5zł (takie rozpoznanie implikuje, że nierozpoznaną dotąd szarą monetą jest 'duża' 1zł) lub monetę 1gr (wtedy z pewnością nierozpoznaną szarą monetą jest 'mała' 10gr).

Gdy jednak zdarzy się, że na stole pojawią się wyłącznie monety 10gr lub wyłączne monety 1zł, program wykrywa taką sytuację i analizuje jasność pikseli w samym centrum monet (ponieważ w tym obszarze są największe różnice między 10gr i 1zł) i na tej podstawie rozpoznaje monetę.

Gdy na stole są monety różnych wielkości, program rozpoznaje monety 10gr i 1zł na podstawie analizy różnic w wielkości promieni poszczególnych monet.

Po rozpoznaniu monet program nadaje etykiety, **oblicza łączną wartość monet i wypisuje wynik na ekran**:


Rysunek 6: Zdjęcie z informacjami do debugowania i nadanymi etykietami


Rysunek 7: Zdjęcie z nadanymi etykietami - wynik działania programu

# 3 Wyniki działania programu dla różnych grup obrazów

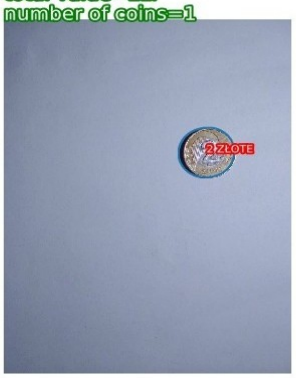
Poniższe zdjęcia prezentują wyniki działania programu uzyskane na różnych obrazach. Z lewej strony widoczne są zdjęcia wejściowe, z prawej strony odpowiadający mu wynik. Obrazy podzieliliśmy na 3 kategorie: **łatwe** (zdjęcia 1-10), **średnie** (zdjęcia 11-20) oraz **trudne** (zdjęcia 21-30).

(1) INPUT:

OUTPUT:
total value=11.1zł
number of coins=4


(2) INPUT:

OUTPUT:
total value=2zł
number of coins=1




(3) INPUT:

OUTPUT:
total value=6zł
number of coins=6


(4) INPUT:

OUTPUT:
total value=3.1zł
number of coins=4


(5) INPUT:

OUTPUT:
total value=19zł
number of coins=9

(6) INPUT:

OUTPUT:
total value=4.1zł
number of coins=5


(7) INPUT:

OUTPUT:
total value=8zł
number of coins=3


(8) INPUT:

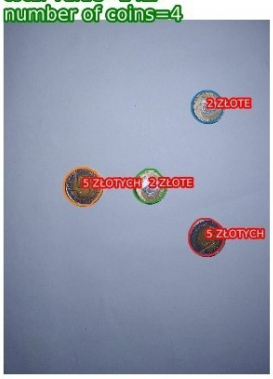
OUTPUT:
total value=22zł
number of coins=10


(9) INPUT:

OUTPUT:
total value=1.01zł
number of coins=2


(10) INPUT:

OUTPUT:
total value=1.4zł
number of coins=4



(11) INPUT:

OUTPUT:
total value=15zł
number of coins=6


(12) INPUT:

OUTPUT:
total value=25.01zł
number of coins=14


(13) INPUT:

OUTPUT:
total value=9.01zł
number of coins=5


(14) INPUT:

OUTPUT:
total value=12zł
number of coins=4


(15) INPUT:

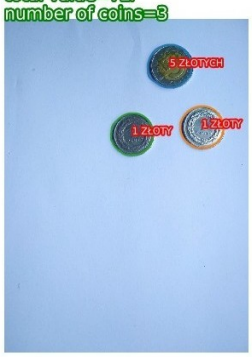
OUTPUT:
total value=2.2zł
number of coins=4

(16) INPUT:

OUTPUT:
total value=12.01zł
number of coins=5


(17) INPUT:

OUTPUT:
total value=7zł
number of coins=3



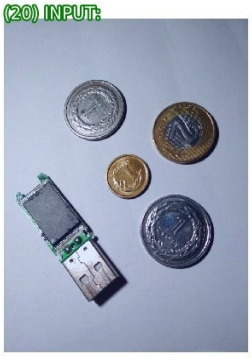

(18) INPUT:

OUTPUT:
total value=5.1zł
number of coins=5


(19) INPUT:

OUTPUT:
total value=9.12zł
number of coins=7


(20) INPUT:



OUTPUT:
total value=4.01zł
number of coins=4

(21) INPUT:

OUTPUT:
total value=0.01zł
number of coins=4


(22) INPUT:

OUTPUT:
total value=0zł
number of coins=1

(23) INPUT:

OUTPUT:
total value=3.2zł
number of coins=14


(24) INPUT:

OUTPUT:
total value=4zł
number of coins=2


(25) INPUT:

OUTPUT:
total value=5zł
number of coins=2

(26) INPUT:

OUTPUT:
total value=5.23zł
number of coins=6


(27) INPUT:

OUTPUT:
total value=12.52zł
number of coins=13


(28) INPUT:

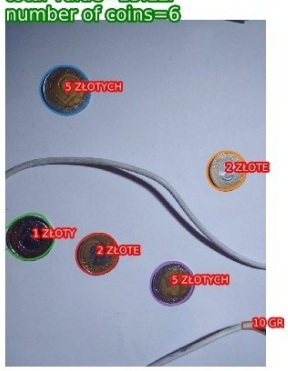
OUTPUT:
total value=0.02zł
number of coins=3


(29) INPUT:

OUTPUT:
total value=6.1zł
number of coins=3


(30) INPUT:

OUTPUT:
total value=15.1zł
number of coins=6



# 4 Omówienie uzyskanych wyników

**Zdjęcia łatwe (1-10)** - poprawnie rozpoznano 100% wszystkich monet

Nasz program bardzo dobrze sprawdził się w anazlizie zdjęć, które zaklasyfikowaliśmy jako 'łatwe' - poprawnie rozpoznał wszystkie 47 monet. Na zdjęciach w tym zestawie monety były dobrze oświetlone i tło było jednolite, na niektórych zdjęciach pojawiła się zmiana perspektywy, większa liczba monet (10) czy występowanły monety wyłącznie 1zł/wyłącznie 1zł i 10gr, co było utrudnieniem ze względu na to, że obie te monety są całkowicie szare. Ponadto program prawidłowo oddzielał monety od tła - na żadnym ze zdjęć nie zaobserwowaliśmy wykrycia elementów innych niż moneta.

**Zdjęcia średnie (11-20)** - poprawnie rozpoznano $92,7\%$ wszystkich monet

Spośród 55 monet znajdujących się na zdjęciach 'średnich' program poprawnie rozpoznał 51. Ta grupa charakteryzowała się między innymi występowaniem monet widocznych rewersem, obcych elementów (długopisy, latarka, pendrive), oddaleniem aparatu, występowaniem wyłącznie monet o jednakowej wielkości (1gr, 10gr / 1zł, 2zł, 5zł) oraz cienia i prześwietleń, które potencjalnie mógłby zakłócić rozpoznawanie monet. W tej katgorii program poprawnie rozpoznał 51 spośród 55 monet, czyli wykazał się niemal 93 procentową skutecznością. 4 monety zostały źle rozpoznane, ponieważ były widoczne pod kątem (nasze monety w obliczeniach traktowaliśmy jako koła - rozwiązaniem mogłoby być traktowanie ich jako elipsy) lub niedoświetlone (wtedy algorytm nieprawidłowo klasyfikował piksele jako szare/żółte - można to rozwiązać poprzez lepsze przygotowanie obrazu do obróbki, rozjasnienie go, zwiększenie nasycenia i kontrastu). W jednym ze zdjęć program rozpoznał kartę SIM jako monetę (aby wyeliminować ten problem można dodać w programie funckję sprawdzającą, czy wykryty kontur jest okręgiem oraz badać piksele podą kątem większej ilości kolorów niż szary/zółty i na tej podstawie eliminować odpowiednie kontury). Program dobrze poradził sobie ze zignorowaniem obiektów takich jak długopisy, pendrive oraz z pozostałymi utrudnieniami, które wymieniliśmy w tym akapicie.

**Zdjęcia trudne (21-30)** - poprawnie rozpoznano $17,5\%$ wszystkich monet

"Trudny" zestaw zdjęć pokazał granicę skuteczności naszego programu. 7 z 40 widocznych na zdjęciach monet zostało rozpoznanych poprawnie, co daje mu 17,5 procentową skuteczność. W tej kategorii testowaliśmy rozpoznawanie monet na różnych tłach, niestety na żadnym z nich (czerwona kratka, koc, gazeta, drewno, kostka brukowa, dłoń) program nie sprawdził się, nie potrafił wykryć położenia monet. Aby to rozwiązać, uważamy, że należałoby użyć bardziej zaawanowanego filtru niż Canny oraz zastosować maski, które uwydatniłyby monety. Pomimo wspomnianych wad są również plusy: programowi udało się poprawnie rozpoznać monety ze zdjęcia (30), które były dosyć słabo oświetlone i w rozpoznawaniu przeszkadzał przewód oraz rzucany przez niego cień oraz bardzo słabo oświetloną monetę 1gr ze zdjęcia (28).

# 5 Krótkie podsumowanie

Nasz program bardzo dobrze poradził sobie z łatwymi zdjęciami, a w przypadku średnich wykazał niemal 93 procentową poprawność detekcji monet. Nie poradził sobie jednak w przypadku zdjęć trudnych, na których monety znajdowały się na nieregularnym tle, bardzo słabo oświetlone lub widoczne pod dużym kątem - uważamy, że w przypadku kontynuacji pracy nad programem jego skuteczność można by poprawić z pomocą przytoczonych przez nas w punkcie 4. rozwiązań i pomysłów.