Cyfrowe Metody Przetwarzania Obrazu

Projekt 2

Konrad Gieleta

297271

MT-IFO 171

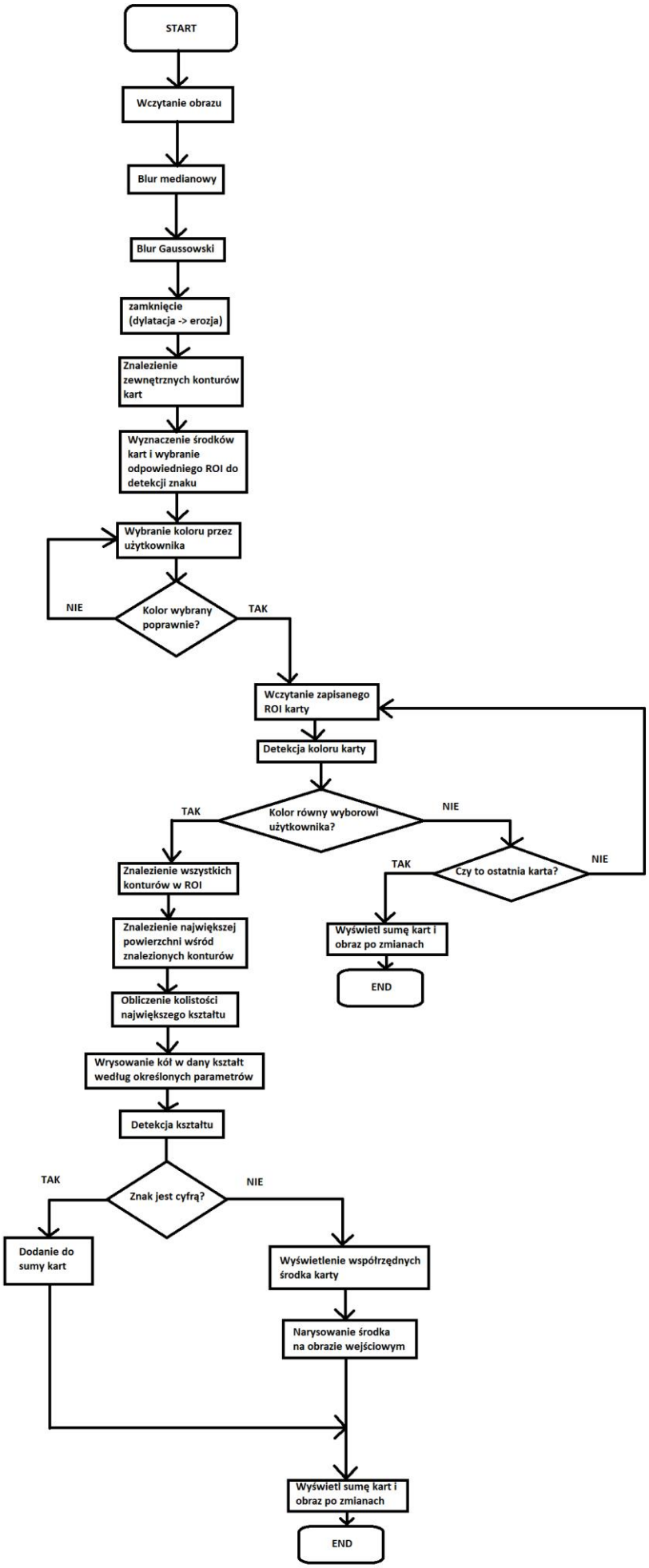
Warszawa 2021/22

1. Opis teoretyczny projektu

Celem projektu było napisanie algorytmu, który rozpoznawał będzie różne rodzaje kart do UNO na 4 róznych scenach, a każda zawierała obraz bez zakłóceń, z blurem, gradientem oraz solą i pieprzem – łącznie 16 obrazów. Zadaniem algorytmu było obliczenie sumy kart o kolorze podanym przez użytkownika lub wszystkich, a jeżeli wśód nich znajdował się znak specjalny – wyznaczenie środka danej karty we współrzędnych obrazu. Dodatkowym wymogiem projketu było przejście każdego obrazu przez dokładnie tą samą ścieżkę przetwarzania bez względu na obecny na nim rodzaj zakłócenia, a także na rozważanie każdej karty osobno, a nie całości obrazu.

Problem polegał więc na odpowiednim usunięciu zakłóceń z obrazka, odpowiej segmentacji każdej karty, a następnie detekcji koloru karty oraz widocznego na niej znaku. Każda karta musiała być rozważana pojedynczo, a więc konieczne było również wyznaczenie odpowiednich cech, które pozwolą na rozpoznanie widocznego kształtu.

2. Projekt ścieżki przetwarzania – schemat blokowy


START
Wczytanie obrazu
Blur medianowy
Blur Gaussowski
zamknięcie (dylatacja -> erozja)
Znalezienie zewnętrznych konturów kart
Wyznaczenie środków kart i wybranie odpowiedniego ROI do detekcji znaku
Wybranie koloru przez użytkownika
Kolor wybrany poprawnie?
NIE
TAK
Wczytanie zapisanego ROI karty
Detekcja koloru karty
Kolor równy wyborowi użytkownika?
TAK
NIE
Znalezienie wszystkich konturów w ROI
Znalezienie największej powierzchni wśród znalezionych konturów
Obliczenie kolistości największego kształtu
Wrysowanie kół w dany kształt według określonych parametrów
Detekcja kształtu
Znak jest cyfrą?
TAK
NIE
Dodanie do sumy kart
Wyświetlenie współrzędnych środka karty
Narysowanie środka na obrazie wejściowym
Czy to ostatnia karta?
TAK
NIE
Wyświetl sumę kart i obraz po zmianach
END
Wyświetl sumę kart i obraz po zmianach
END

3. Opis problemów i ich rozwiązania, dobór algorytmów

Algorytm podzielony jest na trzy etapy

- Preprocessing
- Segmentacja elementów
- Detekcja kształtu i koloru

Pierwszy z nich odpowiada za obróbkę wstępną całego obrazu w celu usunięcia będących na nim zakłóceń stosując na obrazie takie algorytmy jak (dostępne w bibliotece openCV): blur medianowy, blur Gaussowski, oraz progowanie adaptacyjne – w wymienionej kolejności. W wyniku tej części przetwarzania otrzymujemy czarno biały obraz, na którym widoczne są jedynie karty – otrzymujemy uwydatnienie cech obiekt / tło. Zapisywany zostaje także kolorowy obraz po wykonaniu bluru medianowego w celu detekcji kolorów kart.

Następnie rozpoczyna się segmentacja elementów. Na początku znajdowane są zewnętrzne kontury wszystkich elementów i zapisywane do wektora, po czym obliczane pola wewnątrz kontórym i wybierane są z nich jedynie pola kart – pole karty zostało oszacowane na większe niż 16000, a więc tylko takie są zapisywane w wektorze jako właściwe kontury. Następnie wybierane jest ROI zarówno obrazu czarno białego, z którego jest wyodrębniany obszar z widocznym znakiem na środku karty i przekazywany do wektora, jak i kolorowego w taki sposób, aby karta zajmowała jak największą jego powierzchnię jednocześnie z możliwie najmniej widocznym tłem – wartości również zapisywane są w osobnym wektorze. W obu przypadkach ROI dobierane było przy pomocy $\frac{pole\ karty}{N}$, gdzie *N* było dobrane w taki sposób, żeby jak najlepiej był widoczny obszar zainteresowania do danego zadania. Następnie wartość ta była dodawana i odejmowana od środka karty zarówno w kierunku x jak i y. Tworzony jest również trzeci wektor, który przechowuje obliczone w tym etapie środki kart wykorzystując do tego momenty m10, m01 oraz m00 (funkcja moments z openCV).

Ostatni etap polega na detekcji kolorów i kształtów zapisanych na poszczególnych ROI. Detekcję kolorów przeprowadzono obliczając średnią wartość trzech składowych kolorów obrazu w BGR i zapisując je do skalara. Następnie, na podstawie obserwacji wartości danych kolorów można wykryć kolory, wiedzą że:

- Jeżeli wartość czerwona jest większa niż 90 i różnica barwy czeronej i zielonej jest większa niż 40 - karta jest czerwona
- Jeżeli wartość czerwnoa jest większa niż 90, ale różnica barw czerwonej i zielonej jest mniejsza niż 40 – karta jest żółta
- Jeżeli wartość czerwona jest mniejsza niż 90, a wartość niebieska większa niż 40 – karta jest niebieska
- Jeżeli wartość czerwona jest mniejsza niż 90, a wartość niebieska jest mniejsza niż 40, karta jest zielona.

Detekcja kształtów była nieco bardziej problematyczna. Początkowo pomysłem było rozpoznawanie przy pomocy dwóch parametrów – pola powierzchni figury i liczby wrysowanych w figurę okręgów przy pomocy funkcji HoughCircles (funkcja z openCV) dobierając jej odpowiednie parametry. Okazało się to jednak niewystarczające, gdyż zawsze znajdował się obrazek, gdzie zarówno w cyfrę 2 jak i 3 wrysowywane były jeden lub dwa okręgi (rozróżnienie tych cyfr było najtrudniejsze, inne cyfry i znaki dało się rozróżnić chociażby ze względu na pole powierzchni). Drugim pomysłem było dodanie trzeciego parametru – pole wrysowywanych okręgów, jednak i to okazało się niewystarczające do odróżnienia cyfr 2 i 3, ponieważ okręgów obu cyfr znajdwały się w bardzo szerokich zakresach znacząco nachodzących na siebie. Ostatecznie zrezygnowano z pola okręgu a dodano inny parametr – kolistość danej figury liczonej ze wzoru:

$$circularity = \frac{4\Pi \cdot PolePowierzchni}{obwód^2}$$

Ten trzeci parametr okazał się wystarczający. Każdej figurze można przyporządkować określone cechy, na podstawie których możliwe jest odróżnienie ich od siebie. Wygląda to następująco:

- Jeżeli liczba okręgów wynosi 1 oraz kolistość jest mniejsza niż 0.35 – znak to 2
- Jeżeli liczba okręgów wynosci 2 lub liczba okręgów wynosi 1 i kolistość znajduje się w przedziale (0.35, 0.8) – znak to 3
- Jeżeli liczba okręgów wynosi 1 i kolistość większa niż 0.8 – znak to „stop”
- Jeżeli liczba okręgów wynosi 0, a pole figury znajduje się w przedziale (3800, 4900) – znak to 4
- Jeżeli liczba okręgów wynosi 0, a pole figury jest mniejsze niż 3800 - znak to „reverse”

W zależności od znaku program albo dodaje cyfrę do sumy, albo bierze konkretną wartość z wektora zawierającego środki kart i wyświetla je użytkownikowi.

4. Analiza i Prezentacja wyników

Algorytm pomyślnie detekuje wszystkie znaki znajdujące się na kartach. Jeżeli karta danego koloru jest cyfrą – dodaje ją do sumy, a jeżeli znakiem – wyświetla użytkownikowi współrzędne środka karty.

Algorytm obróbki wstępnej zdjęcia również spełnia swoje zadanie. W zależności od zakłócenia nie zawsze widoczne są wszystkie wewnętrzne krawędzie karty, jednak, co najważniejsze w celu dalszego przetwarzania, zawsze bardzo wyraźnie widać znak na środku karty. Detekcja kolorów również w każdym przypadku działa, co finalnie sprowadza się do działania alogrytmu na wszystkich 16 obrazach.

Prezentacja wyników:

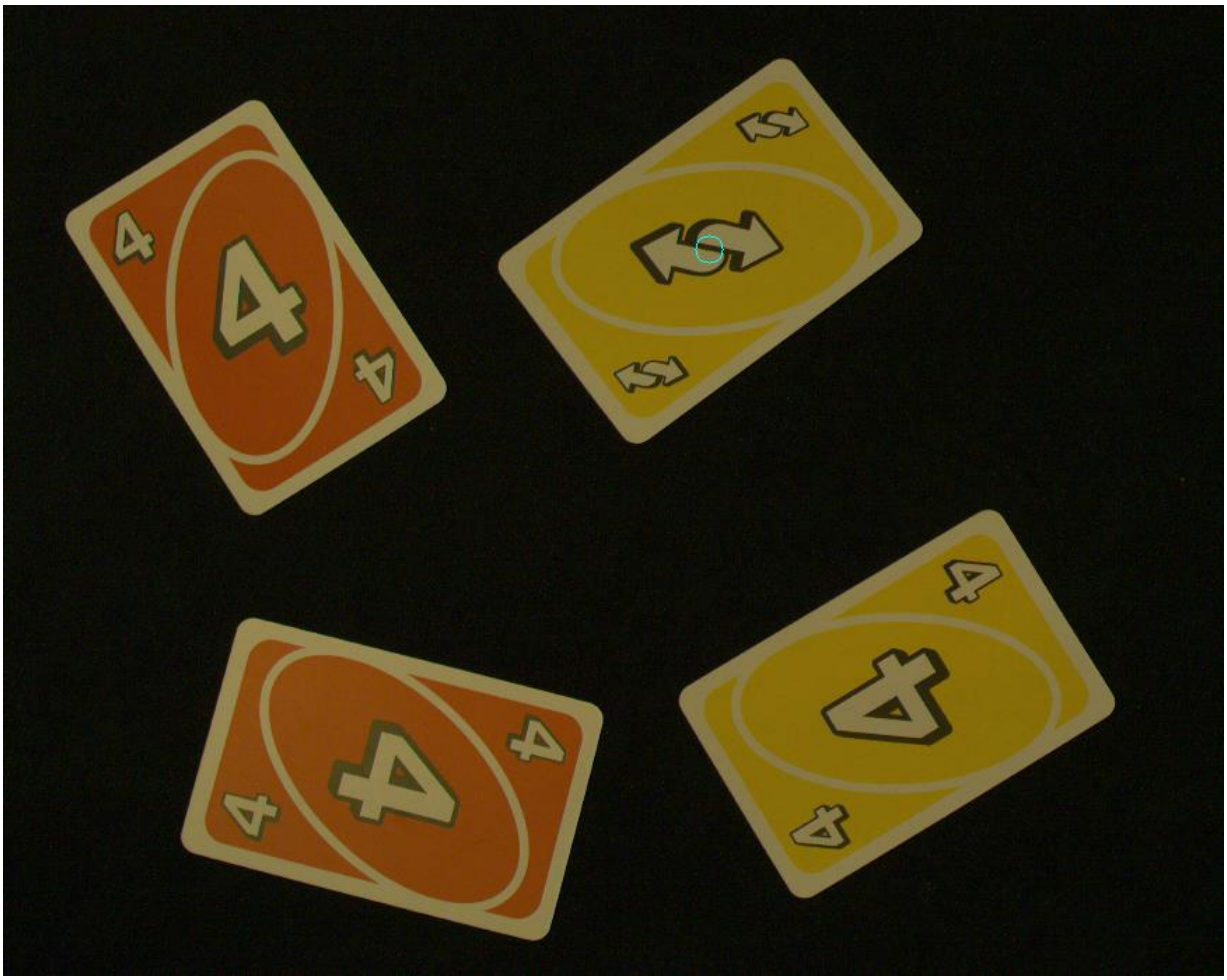
Przedstawione zostaną po 3 przykłady do każdej sceny dla różnych kolorów.

a) Przykład obrazów bez zakłóceń:

- kolor – wszystkie karty

```
detected shape: stop
card center coordinates: [511.616, 536.463]
detected shape: reverse
card center coordinates: [377.414, 222.368]
Sum of cards: 6
```

- kolor – żółty



```
detected shape: reverse
card center coordinates: [532.825, 224.491]
Sum of cards: 4
```

- kolor – niebieski

```
detected shape: stop
card center coordinates: [233.618, 453.753]
Sum of cards: 5
```

b) zakłócenie – blur

- kolor – zielony

```
Sum of cards: 2
```

- kolor – wszystkie karty

```
detected shape: reverse
card center coordinates: [567.098, 612.78]
detected shape: stop
card center coordinates: [233.925, 502.176]
Sum of cards: 5
```

- kolor – czerwony

```
Sum of cards: 8
```

c) zakłócenie – gradient

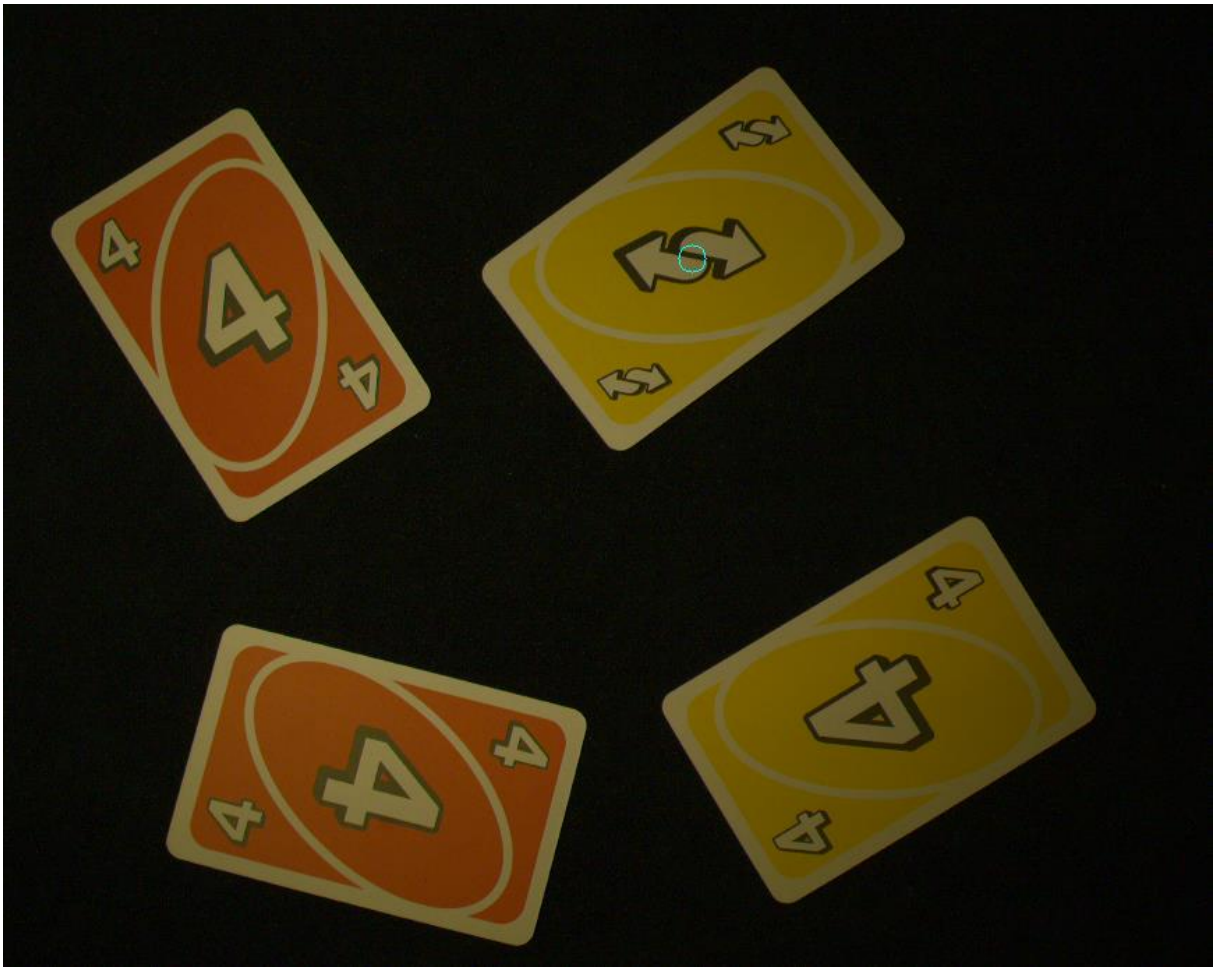
- kolor – niebieski

```
detected shape: stop
card center coordinates: [233.643, 453.751]
Sum of cards: 5
```

- kolor – zielony

```
Sum of cards: 2
```

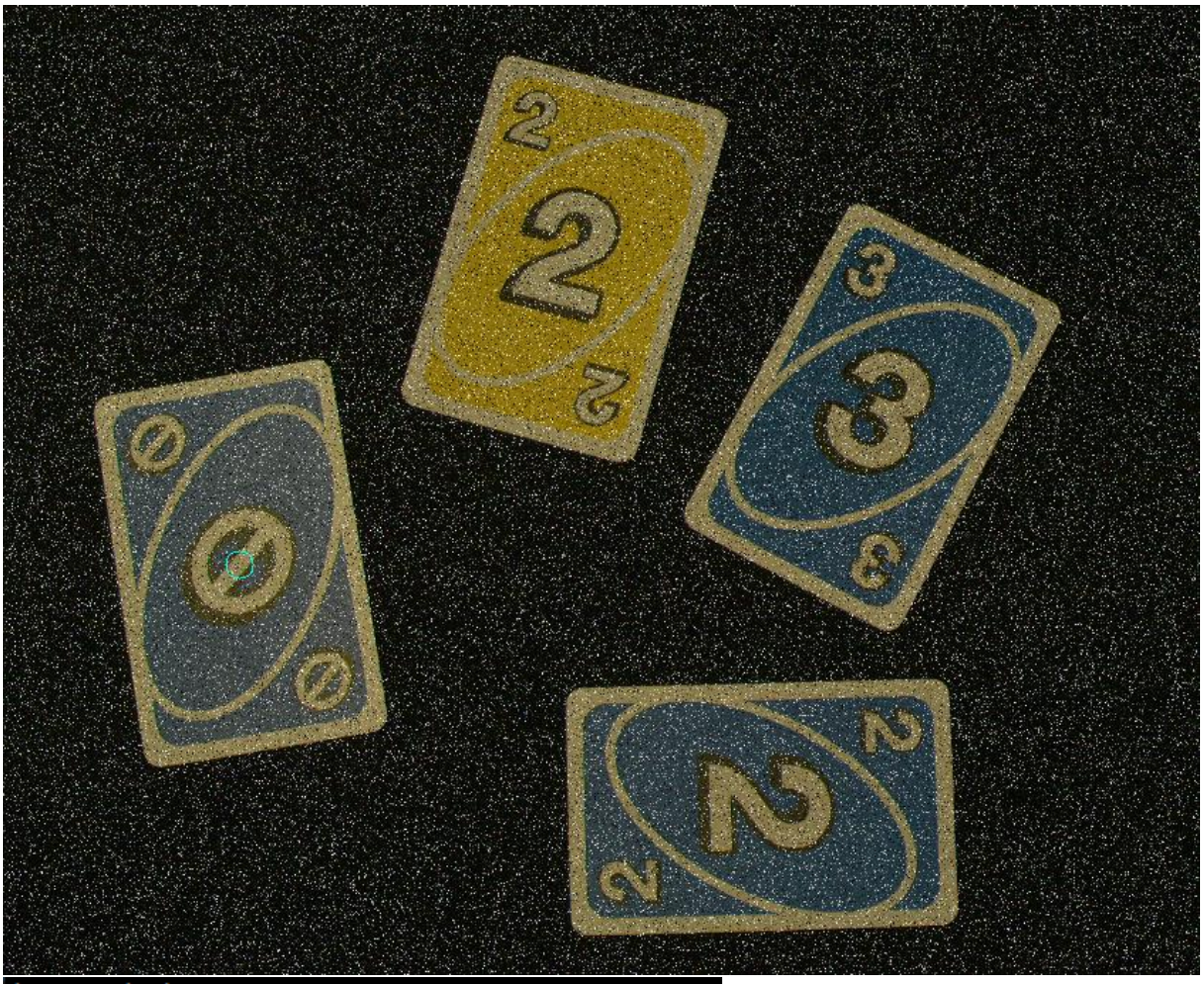
- kolor – żółty



```
detected shape: reverse
card center coordinates: [532.667, 224.628]
Sum of cards: 4
```

d) zakłócenie – sól i pieprz

- kolor – wszystkie karty

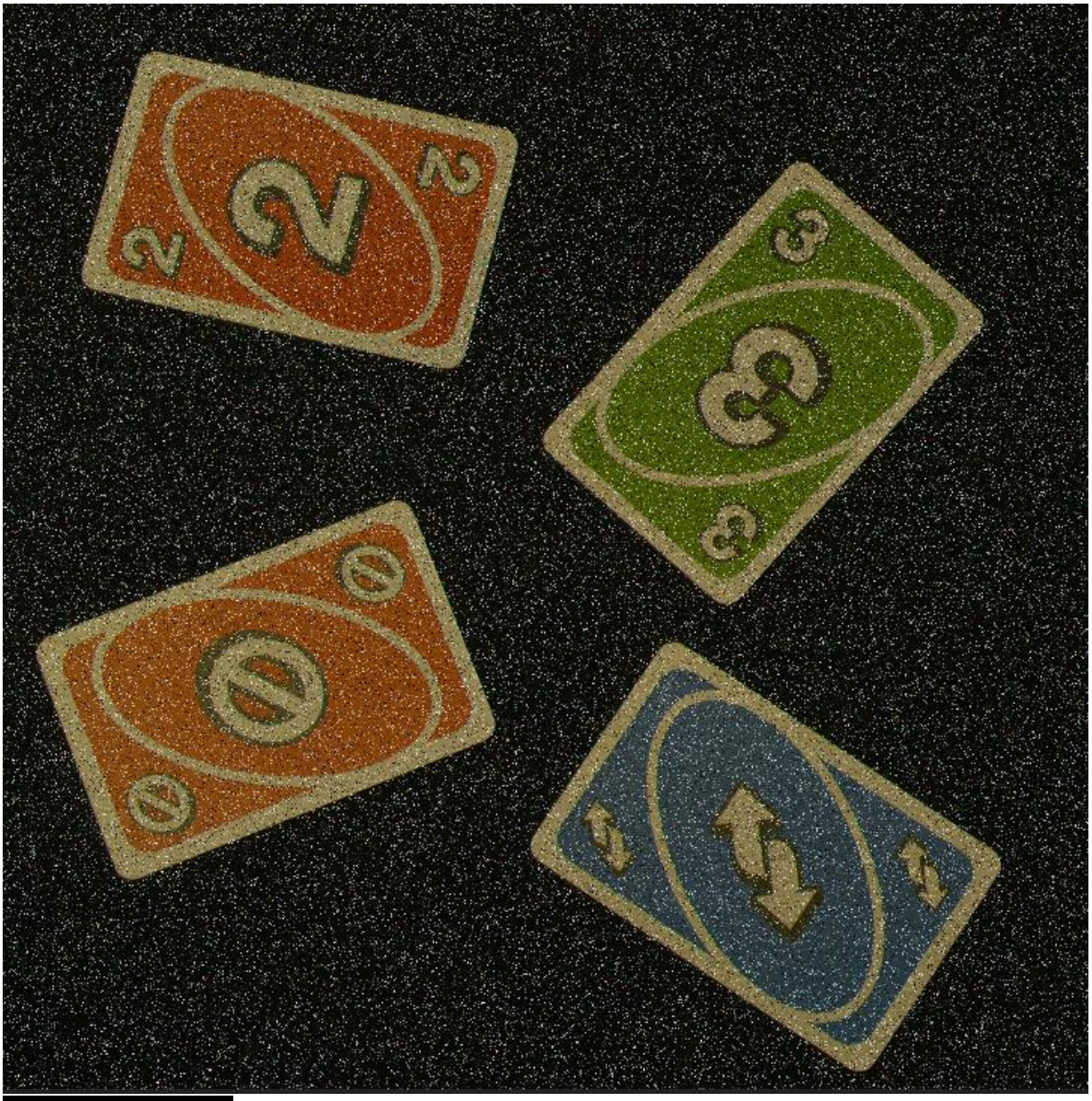


```
detected shape: stop
card center coordinates: [233.601, 453.785]
Sum of cards: 7
```

- kolor – czerwony

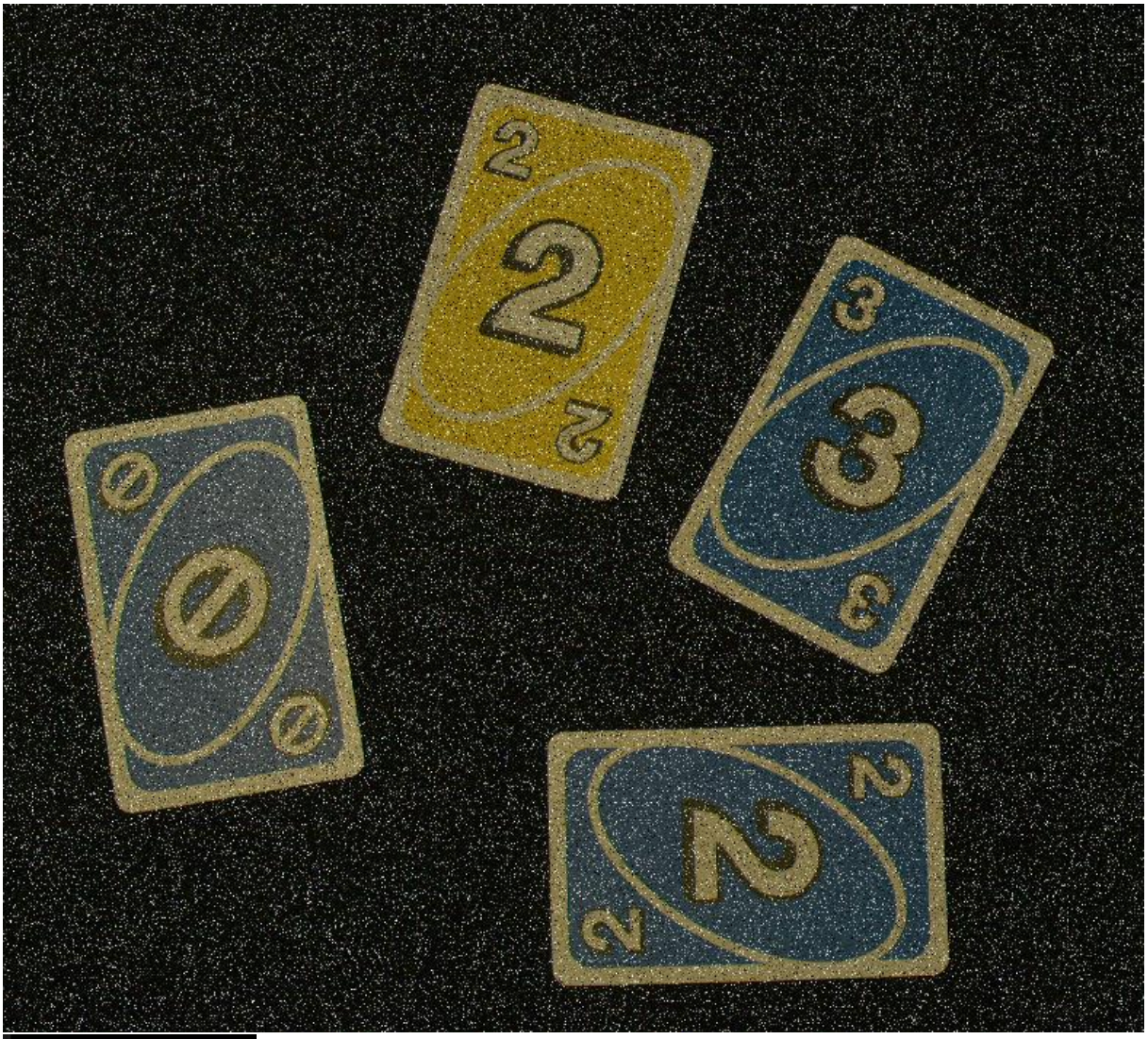
```
detected shape: stop
card center coordinates: [511.565, 536.451]
detected shape: reverse
card center coordinates: [377.483, 222.327]
Sum of cards: 0
```

- kolor – zielony



```
Sum of cards: 3
```

- kolor – czerwony



```
Sum of cards: 0
```