Akademia Górniczo-Hutnicza w Krakowie

Wydział Fizyki i Informatyki Stosowanej

Analiza obrazów

Segmentacja i Koloryzacja Obrazów

Krzysztof Konieczny, Andrzej Świętek, Marcin Knapczyk, Aleksandra Samborek

26 stycznia 2025

Spis treści

1. Opis projektu

Aplikacja umożliwia segmentację obrazów na podstawie wykrytych regionów kolorystycznych, a następnie generuje kolorowankę z zaznaczonymi obszarami, które należy pokolorować odpowiednimi kolorami. Prezentuje również wersję obrazka z naniesionymi kolorami na poszczególnych segmentach - pokolorowaną kolorowankę.

2. Kod źródłowy

Kod źródłowy projektu, przykładowe dane testowe, jak również instrukcja jak go uruchomić znajdują się w repozytorium na githubie pod linkiem `https://github.com/Andrzej-Swietek/AO-Final-Project`.

3. Jak działa algorytm

Algorytm wykorzystuje techniki k-means clustering oraz klasyczne metody przetwarzania obrazów, aby segmentować obrazy i analizować ich strukturę. Oto szczegółowy opis działania:

1. **Skalowanie** - Pierwszą operacją jest zmniejszenie obrazka zachowując proporcję do rozmiarów w których ani jego szerokośc ani wyskość nie przekraczają 512 pikseli. Ze względu na specyfikę algorytmu mniejsze rozmiary dają lepsze efekty i przyśpieszają działanie.
2. **Filtrowanie** - Na obrazie stosowany jest silny filtr bilateralny. Powoduje on rozmycie podobne do filtra Gaussa, ale zachowuje przy tym krawędzie.
3. **K-means clustering** Na wejściu obraz zostaje poddany klastrowaniu k-means, co redukuje jego kolory do k kolorów. Wynikowy obraz zawiera piksele tylko k kolorów.
4. **Tworzenie masek binarnych** Dla każdego koloru z k kolorów generowana jest maska binarna. Każda z nich reprezentuje obszary obrazu o danym kolorze.
5. **Filtry na maskach binarnych** Na każdej z mask wykonywane jest morfologiczne otwarcie, żeby usunąć małe samotne grupy pikseli, które w późniejszym etapie wprowadzałyby szum.
6. **Wykrywanie konturów** Aplikujemy dylatacje na maski i odejmujemy oryginalną maskę. Otrzymujemy w ten sposób kontury dla danego koloru.
7. **Łączenie konturów** Wykonujemy operację dodawania logicznego na wszystkich konturach z masek i otrzymujemy kontury dla całego obrazku. Następnie negujemy wynik by kontury były czarne i wykonujemy morfologiczne otwarcie aby pozbyć się potencjalnych podwojonych krawędzi.
8. **Uzgadnianie koloru obszarów** Dla każdego obiektu z każdej maski szukamy w środku punktu w którym postawimy kropkę określającą na jaki kolor pomalować dany segment. Punkt ten określany jest na podstawie transformacji odległościowej - szukamy miejsca którego suma odległości od krawędzi jest największa.
9. **Nakładanie znaczników kolorów** Na podstawie poprzedniego kroku, na obraz kontur nakładamy "kropki"pełniące funkcję znaczników na jaki kolor pomalować dany obszar. Rozmiar kropki określany jest na podstawie pola obiektu do którego należy znacznik.
10. **Tworzenie podglądu** Dodatkowo tworzymy podgląd ilustrujący jak będzie wyglądać obraz po pokolorowaniu. Aby to osiągnąć nakładamy kolory na odpowiadające im maski i łączymy je już jako trójkanałowe obrazy przy użyciu operacji dodawania logicznego. Następnie nakładamy maskę z kontur i otrzymujemy obraz poglądowy.

4. Instrukcja użytkowania

4.1. Strona główna (/home)

Po otwarciu aplikacji użytkownik zostaje przywitany na stronie głównej: "Welcome to the Colorization App"z przyciskiem "Get Started", który prowadzi do strony /upload.

4.2. Segmentacja i koloryzacja (/upload)

1. Upload obrazka:
   - Kliknij "Choose file", aby wczytać plik z dysku lokalnego
   - Jeśli obrazek nie został jeszcze załadowany, widnieje komunikat "No file chosen".
   - Po załadowaniu obrazek zostanie wyświetlony pod etykietą "Image preview".
2. Wybór liczby kolorów:
   - Wybierz maksymalną liczbę kolorów (od 2 do 32) za pomocą menu rozwijanego "Select color count". Aktualna liczba kolorów na obrazku może być mniejsza, jeżeli nie będzie ich na tyle dużo.
3. Przetwarzanie obrazu:
   - Kliknij przycisk "Process image", aby uruchomić algorytm.
   - Wynik:
     - Po lewej: kolorowanka z kropkami w odpowiednich kolorach.
     - Po prawej: gotowa pokolorowana wersja obrazka.
4. Pobieranie wyniku:
   - Kliknij "Download", aby pobrać pokolorowany obrazek.
5. Reset:
   - Przyciskiem "Reset"można powrócić do początkowego widoku strony /upload.

4.3. Informacje o Projekcie (/info)

Strona zawiera opis projektu, technologii używanych w jego realizacji oraz zespół odpowiedzialny za jego stworzenie.

5. Podział zadań w zespole

- Krzysztof Konieczny: Implementacja algorytmu przetwarzania obrazów.
- Andrzej Świętek: Backend API w Flasku i integracja z algorytmem.
- Marcin Knapczyk: Frontend w React i TypeScript oraz UX/UI.
- Aleksandra Samborek: Optymalizacja treści i interfejsu użytkownika oraz dokumentacja projektu.

6. Ograniczenia:

- Założenia dotyczące formatu obrazu: Aplikacja przyjmuje jedynie obrazy w formacie .jpg lub .png.
- Metoda może zawodzić przy skomplikowanych obrazach z dużą liczbą detali. Problematyczne jest również nierównomierne oświetlenie. Rekomendowane są minimalistyczne obrazy z dużym kontrastem i z jasnymi kolorami, aby lepiej kontrastowały z krawędziami na ostatecznym wyniku.
- Odpowiednia ilość segmentów - Algorytm najlepiej działa, jeśli użytkownik wybierze rzeczywistą liczbę znaczących kolorów na obrazie.
- Dokładny wybór ilości kolorów - Ustalenie liczby k nie zawsze będzie skutkowało k kolorami na obrazie wynikowym. Dzieje się tak, ponieważ niektóre kolory mogą zajmować tak mało obszarów, że zostają całkowicie usunięte przez morfologiczne otwarcie.

7. Błędy:

- Brak znacznika w niektórych miejscach - Czasami niektóre obszary mogą powstawać z powodu nałożenia się kontur z różnych masek na siebie tworząc nowy zamknięty obszar

- Zanikanie niektórych kolorów widocznych jako czarne obszary na obrazku wynikowym - problem wynika z faktu, że po wykonaniu morfologicznego otwarcia na wszystkich maskach ich suma logiczna może nie pokryć oryginału. Niepokryte obszary objawiają się jako czarne "plamy"na obrazie wynikowym
- Znacznik może wychodzić po za krawędzie - jeżeli obszar jest podłużny to warunek pola może nie być wystarczający

8. Dane testowe

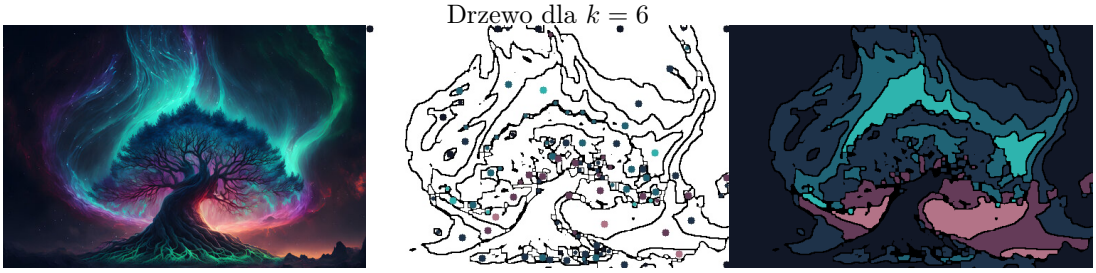
Dane wykorzystane w projekcie pochodzą z publicznych zestawów obrazów o różnorodnej tematyce jak również z generatorów AI:

- Rozdzielczość obrazów: dowolna (dłuższa krawędź skalowana do rozdzielczości 512px, a krótsza skalowana proporcjonalnie do obrazu wejściowego).
- Zestawy testowe: Zdjęcia roślin i zwierząt, grafiki, schematy kolorystyczne.

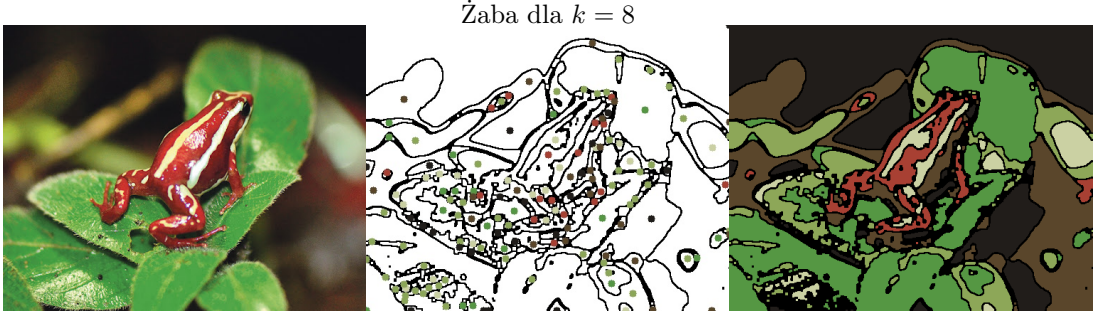
Najlepsze wyniki algorytm uzyskuje dla obrazów z wyraźnie widocznymi różnicami w kolorystyce.

Przykłady:

Drzewo dla $k = 6$



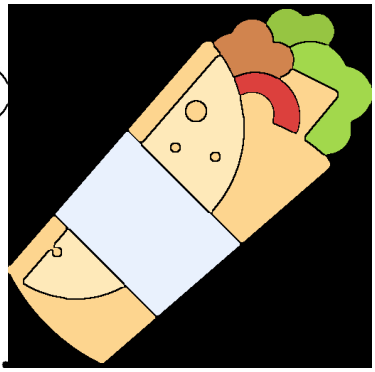
Żaba dla $k = 8$



Grafika dla $k = 6$

Kebab dla $k = 16$, przeźroczyste tło zostało zastąpione czarnym





Żółw kosmonauta dla $k = 8$



Postać z bajki dla $k = 4$