LABORATORIUM SIECI KOMPUTEROWYCH

Data wykonania ćwiczenia:	**21.02.2024**
Rok studiów:	**2023/2024**
Semestr:	**1**
Grupa studencka:	**1**
Grupa laboratoryjna:	**b**

Ćwiczenie nr *1*

Temat: *Restauracja starych fotografii*

Osoby wykonujące ćwiczenia:

1. Michał Herczyński

Katedra Informatyki i Automatyki

1. Cel ćwiczenia

Celem ćwiczenia jest zaprojektowanie skryptu do przywracania i poprawiania starych i uszkodzonych fotografii poprzez eliminację zarysowań oraz wypełnienie brakujących elementów obrazu. W tym celu należy wykorzystać język programowania Python oraz bibliotekę open CV.

2. Przebieg ćwiczenia

W celu realizacji zadania wykorzystano zdjęcie o rozdzielczości 800x1119 px pobrane z internetu:

Illustration 1: Uszkodzone zdjęcie wymagające poprawek

W celu jego naprawy, metodą prób i błędów zbadano działanie różnych narzędzi dostarczanych przez bibliotekę open CV oraz zaprojektowano pipeline który prowadzić ma do zakładanego celu:

a) Załadowanie uszkodzonego zdjęcia w odcieniach szarości,

b) Normalizacja oraz skalowanie jasności pikseli,

c) utworzenie maski w procesie progowania w celu wyodrębnienia dużych, białych braków w zdjęciu oraz zwiększenie powierzchni fragmentów maski,

d) utworzenie maski na podstawie wyszukiwania krawędzi metodą Canny oraz zwiększenie powierzchni fragmentów maski,

e) Sumowanie utworzonych masek,

f) Wypełnienie fragmentów wskazanych przez maskę,

g) rozmycie zdjęcia oraz odwrócenie rozmycia w celu usunięcia niewielkich zagięć papieru.

Po każdym z tych etapów skrypt zwraca efekt działania w celu analizy działania algorytmów.

Punkty a i b realizuje kod:

```
img = cv2.imread(PHOTO_PATH, cv2.IMREAD_GRAYSCALE)
stretched_image = cv2.normalize(img, None, 0, 230, cv2.NORM_MINMAX)
scaled_image = cv2.multiply(stretched_image,ALPHA_VALUE)
```

Wskazany proces tworzy kontrastowy obraz w odcieniach szarości ułatwiający wyodrębnienie uszkodzonych fragmentów i utworzenie maski

Illustration 2: Zdjęcie w odcieniach szarości ze zwiększonym kontrastem

W kolejnym kroku tworzone są binarne maski uszkodzeń na podstawie dwóch operatorów dostępnych w bibliotece openCV a następnie są one sumowane operatorem or:

```
__, damage_mask = cv2.threshold(scaled_image, SEGMENTATION_MIN, 255, cv2.THRESH_BINARY)
damage_mask = cv2.dilate(damage_mask, np.ones((5,5)))
canny_edges = cv2.Canny(scaled_image, threshold1=254, threshold2=255)
canny_edges = cv2.dilate(canny_edges, np.ones((4,4)))
damage_mask_sum = cv2.bitwise_or(canny_edges, damage_mask)
```

Powyższy fragment tworzy maski przy użyciu progowania oraz wykrywania krawędzi metodą Canny. Każda masek jest niezależnie rozszerzana w celu lepszego pokrycia uszkodzonej

powierzchni. Operacja jest wykonywana niezależnie w celu optymalizacji i ochrony przed nadmiernym rozmyciem obrazu. W ostatnim kroku maski są sumowane.

Illustration 3: Efekt działania operatora Canny

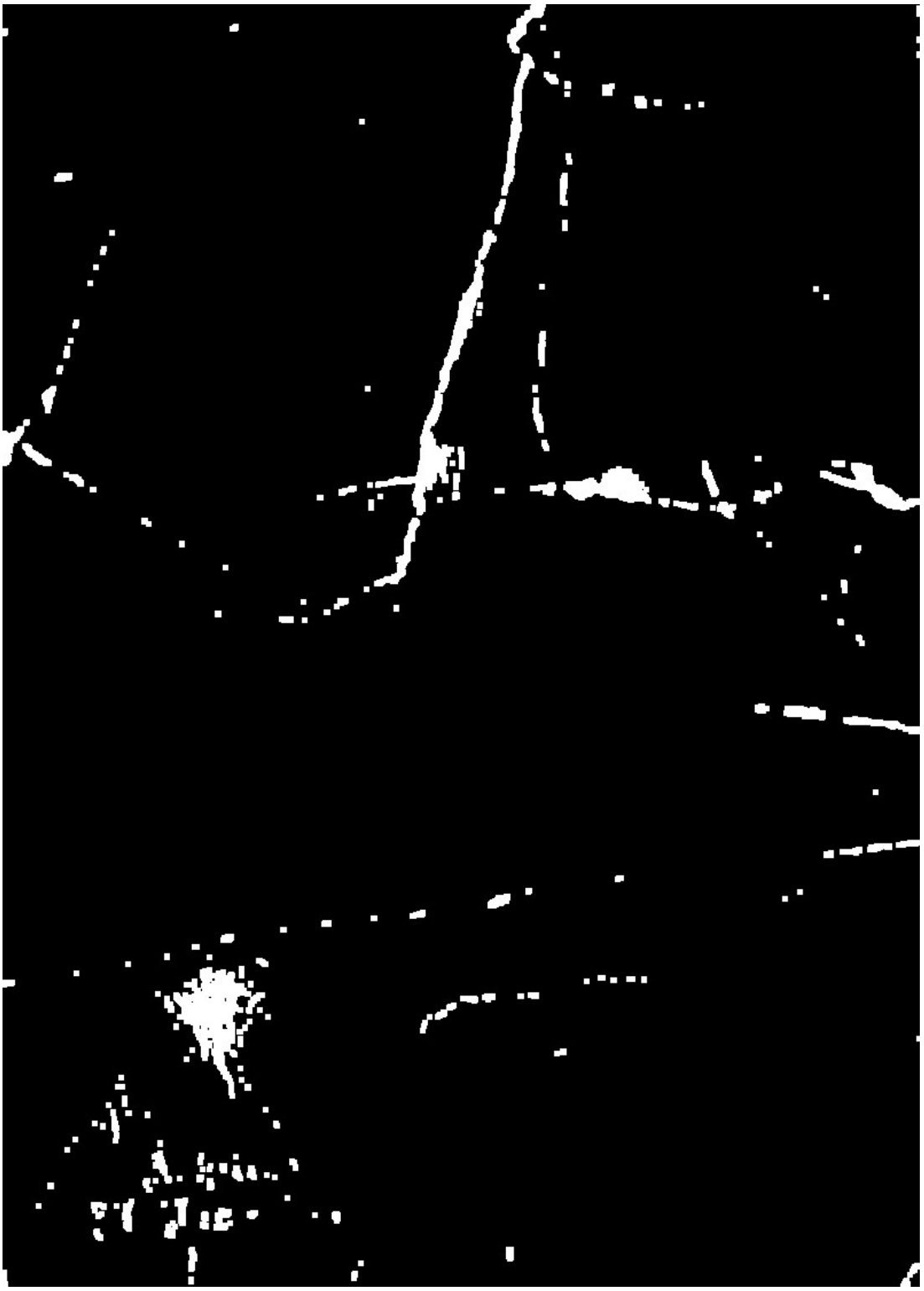

Illustration 4: Efekt działania progowania binarnego

Illustration 5: Suma utworzonych masek tworząca obraz uszkodzeń zdjęcia

W dalszym kroku skrypt wykonuje wypełnienie pozytywnych pikseli maski a następnie używa rozmycia Gaussa oraz wyostrza zdjęcie, co redukuje niewielkie zarysowania papieru. Finałowy efekt wygląda następująco:

Illustration 6: Efekt działania skryptu

3. Wnioski

W toku wykonanego ćwiczenia utworzono skrypt pozwalający na naprawę starych, zniszczonych zdjęć w sposób bezobsługowy przy użyciu narzędzi dostarczanych przez pakiet openCV. Pozwala on na zautomatyzowaną naprawę wielu zdjęć o podobnej charakterystyce znajdujących się w jednym folderze, co pozwala na zaoszczędzenie czasu.