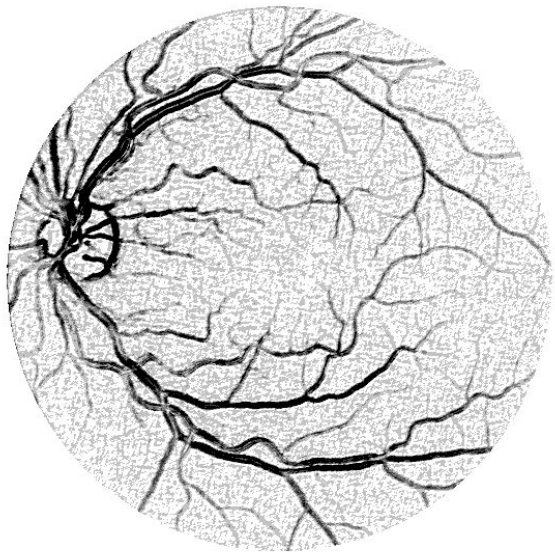
Wykrywanie naczyń dna siatkówki oka

Słowa kluczowe: przetwarzanie obrazów, uczenie maszynowe, analiza danych

Opis: Należy napisać aplikację, która dla zadanego obrazu wejściowego przedstawiającego dno siatkówki oka (przykład poniżej), wykrywa (automatycznie) naczynia krwionośnie. Formalnie: dla każdego piksela obrazu algorytm musi stwierdzić czy ten piksel stanowi naczynie krwionośne czy nie.



Wymagania (obowiązkowe):

- Algorytm w podstawowej wersji (na 3.0) powinien wykorzystywać techniki przetwarzania obrazu (poznane między innymi na przedmiocie KCK – zadanie z samolotami/projekt z obrazów) do detekcji naczyń krwionośnych. W ramach takiego procesu przetwarzania można wyróżnić 3 główne elementy:
    - **Wstępne przetworzenie obrazu**: wejściowy obraz może być zaszumiony/zbyt ciemny/jasny. Można tutaj wykorzystać takie techniki jak: rozmycie (splot – np. filtr gaussowski, morfologia – np. filtr medianowy), wyostrzenie, normalizacja histogramu kolorów itp.
    - **Właściwe przetworzenie obrazu w celu wyodrębnienia naczyń krwionośnych.** Można zastosować różne techniki wykrywania krawędzi, jednak podstawowe wersje tych metod mogą okazać się wysoce nieskuteczne. Należy zastanowić się nad uwzględnieniem piksela i jego najbliższego sąsiedztwa, jako podstawy klasyfikacji (np. jeżeli piksel jest mocno czerwony, a większość pikseli z jego sąsiedztwa ma inny kolor, to piksel reprezentuje naczynie krwionośne).
    - **Ponowne przetwarzanie**: Przetwarzanie w celu poprawy skuteczności wykrywania naczyń. Należy zwrócić uwagę na takie elementy jak np. połączenie naczyń (choroby mogą powodować ich pękanie), naczynia zazwyczaj nie zmieniają gwałtownie swojej grubości i kierunku. Można zastanowić się nad tymi cechami i wykorzystać proste techniki do odfiltrowania błędów false-positive bądź naprawy błędów typu false-negative.
- **Wynik obowiązkowo** należy wizualizować na oryginalnym (kopii ☺) obrazie, np. zamalowując wyróżniającym się kolorem piksele zaklasyfikowane jako naczynie krwionośne.

W tym celu najlepiej wygenerować binarną maskę odpowiedzi algorytmu, która zostanie potem wykorzystana do analizy statystycznej.

- **Ważnym elementem oceny jest skuteczność algorytmu:** Należy dokonać podstawowej analizy statystycznej jakości działania algorytmu. Działanie programu należy przetestować na minimum 5 obrazach. Do każdego obrazu dana jest binarna maska reprezentująca oczekiwaną (najlepszą) odpowiedź algorytmu. Można policzyć takie statystyki jak:
    - Błąd średniokwadratowy różnicy oczekiwanej maski i otrzymanej maski odpowiedzi.
    - Tablica pomyłek (True/False Positive/Negative) wraz z takimi miarami jak: accuracy, precision, sensitivity, itp (link poniżej).

Wymagania na 4.0:

- Należy wykorzystać proste metody klasyfikacji - po wstępnym przetworzeniu obrazu należy podzielić go na niewielkie części (np.: 5x5 px) i dla każdej z nich dokonać ekstrakcji cech z obrazu: np. wariancja kolorów, momenty centralne, momenty Hu itp. Z użyciem tak uzyskanych cech oraz masek z ręcznie zaznaczonymi naczyniami krwionośnymi należy zbudować wybrany klasyfikator, np.:
    - k-NN
    - Klasyfikacja Rocchio
- Trafność klasyfikacji tak opracowanego klasyfikatora należy zweryfikować na niezależnym zbiorze testowym.

Wymagania na 5.0:

- Tak samo jak punkt na 4.0, jednak należy wykorzystać bardziej zaawansowany klasyfikator, np.: sieć neuronowa, drzewo decyzyjne C4.5, algorytm indukcji reguł LEM (i pochodne) - można wykorzystać gotowe implementacje klasyfikatorów.
- Należy wykorzystać k-krotną walidację skrośną (k-fold cross validation) w celu oceny zbudowanego klasyfikatora i uniknięcia przeuczenia.

Linki

- Baza obrazów HRF: https://www5.cs.fau.de/research/data/fundus-images/
- Baza obrazów STARE: http://cecas.clemson.edu/~ahoover/stare/probing/index.html
- Baza obrazów CHASE: https://staffnet.kingston.ac.uk/~ku15565/CHASE_DB1/