Projekt: rozpoznawanie liści – Etap I

Katarzyna Markowska, Jędrzej Kuczyński

Wykrywanie liścia

Identyfikacja liścia na obrazie rozpoczyna się od użycia funkcji **threshold_and_label**, która najpierw zamienia obraz kolorowy na obraz w skali szarości. Następnie przeprowadza progowanie według zadanej wartości, która w przypadku naszego projektu wynosi 0,7. Ostatnim zadaniem tej funkcji jest wywołanie polecenia **label** oraz zwrócenie utworzonego obiektu.

Kolejnym krokiem jest użycie polecania **regionprops**, dzięki któremu możliwe jest otrzymanie większej ilości informacji z wcześniej poetykietowanych obszarów. Dla każdego zdjęcia przeprowadzana jest selekcja obrazów na podstawie 3 kolejno zagnieżdżonych warunków:

- Region nie znajduje się poza 88% górnej i lewej części całego zdjęcia
- Region zaczyna się w 60% górnej i lewej części całego zdjęcia
- Pole powierzchni regionu wynosi co najmniej 4000 pixeli

Kolejne analizy znalezionego liścia są wykonywane tylko jeśli region przejdzie powyższe filtry. Ze wszystkich testowanych obrazów nie przechodzą ich 4 zdjęcia: 2 Carya glabra oraz 2 Cornus florida.

Pole powierzchni

Funkcja **calculate_area** odczytuje wartość pola **area** danego regionu, które zawiera w sobie ilość pixeli składających się na region.

Stosunek pola powierzchni do obwodu

W celu uzyskania tej cechy wywoływana jest funkcja **calculate_contour**. Działa ona na polu **image** danego regionu. Region ten jest najpierw poddany operacji zamknięcia w celu wygładzenia nierówności spowodowanych wycinaniem. Następnie dodane są marginesy, aby kontur mógł zostać znaleziony jako jeden ciąg pixeli. Szukanie granic liścia realizowane jest z użyciem funkcji **find_contours**, która zwraca wszystkie znalezione kontury, a dokładniej należące do nich punkty. Jeśli znalazła ona więcej niż 1 kontur, to wybierany jest ten najdłuższy, ponieważ w większości jest to ten prawidłowy, podczas gdy reszta to tylko artefakty powstałe przy progowaniu.

Ostatnim etapem jest użycie funkcji **calculate_area_to_contour**, która zwraca stosunek pola powierzchni do długości obwodu zaokrąglony do 5 miejsca po przecinku.

Długość ogonka

Za wyliczenie tej cechy odpowiada funkcja **calculate_tail**, która podobnie jak ta w poprzednim punkcie otrzymuje jako argument wejściowy obraz po operacji zamknięcia z dodanymi marginesami. Na podstawie takiego obrazu tworzony jest następnie obraz otwarty za pomocą maski o promieniu 4. Kolejnym krokiem jest odjęcie od obrazu wejściowego obrazu otwartego, dzięki czemu uzyskujemy sam ogonek. Wartością zwracaną jest suma pixeli należących do wyszukanego regionu.

Liczba listków składowych

Funkcja **subleaves (leaf_img, mask)** - oblicza liczbę "składowych" listków w danym liściu. Funkcja eroduje dany liść maską dyskową o promieniu 10 funkcją **morphology.binary_erosion** (empirycznie stwierdzono, że maska o promieniu 10 najlepiej różnicuje gatunki). Następnie definiuje regiony **(measure.label i measure.regionprops**), określa ich liczbę i zwraca tę wartość.

Cechę tę wybrano, ponieważ niektóre liście są tzw. liśćmi złożonymi. Liście złożone mają wiele blaszek liściowych (listków) osadzonych na wspólnej osi.

Struktura krawędzi liścia

Funkcja **contour_histogram (contour_points, list_of_angles)** - dla każdej liczby *angle* z listy *list_of_angles* do listy *contour_points*, zawierającej punkty należące do konturu (obliczonego za pomocą funkcji **calculate_contour**) zostają dodane odpowiednie końce tej listy, aby "zapętlić" kontur w celu bezproblemowej iteracji. Następnie dla każdego punktu konturu wybierani są jego dwaj sąsiedzi: "sąsiad lewy" oddalony o *angle* indeksów w lewo oraz "sąsiad prawy" oddalony o *angle* indeksów w prawo. Następnie funkcja oblicza cosinus kąta według wzoru:

$$\cos \alpha = \text{dot\_product}(va, vb) / \text{len}(va) * \text{len}(vb)$$

gdzie:
- dot_product - iloczyn skalarny
- va - wektor od prawego sąsiada do punktu środkowego
- vb - wektor od lewego sąsiada do punktu środkowego
- len() - długość wektora zdefiniowana jako pierwiastek kwadratowy z sumy kwadratów współrzędnych wektora

Za pomocą funkcji **np.arccos** oraz **np.degrees** uzyskany zostaje kąt w stopniach między danymi trzema punktami. Następnie z wszystkich obliczonych kątów dla wszystkich zmiennych *angle* liczony jest histogram za pomocą funkcji **np.histogram(bins=20)**, który jest zwracany przez tę funkcję.