Grenlandia się topi

badanie rozkładu kątów pomiędzy strumykami na lądolodzie na podstawie analizy obrazu

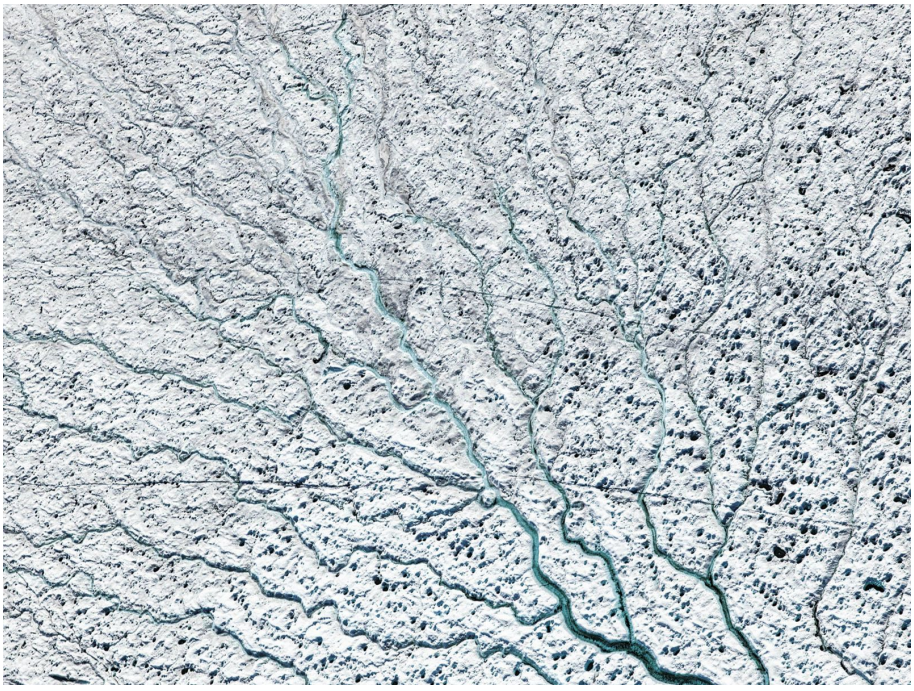
Małgorzata Bąk, Marcin Byra, Filip Chudzyński, Marcin Osiekowicz
Opiekun: dr hab. Piotr Szymczak

Zespołowy Projekt Studencki
Wydział Fizyki UW

8.06.2018

Streszczenie

Na skutek ocieplania się klimatu lądolód grenlandzki stopniowo się topi, a na jego powierzchni tworzą się strumyki. Przy pewnych założeniach grubość warstwy wody na lądolodzie można opisywać równaniem Poissona. Wówczas można się spodziewać, że kąty pomiędzy łączącymi się strumykami będą wynosiły średnio 72^o. W ramach wydziałowego hackathonu zaproponowano projekt mający na celu sprawdzenie tej hipotezy. Otrzymano jedno zdjęcie satelitarne topiącego się lądolodu i zastosowano różne metody przetwarzania obrazów, aby wyodrębnić sieć strumyków. Ze względu na specyfikę otrzymanego zdjęcia oraz ograniczenia czasowe nie udało się jednak znaleźć rozkładu kątów pomiędzy strumykami.



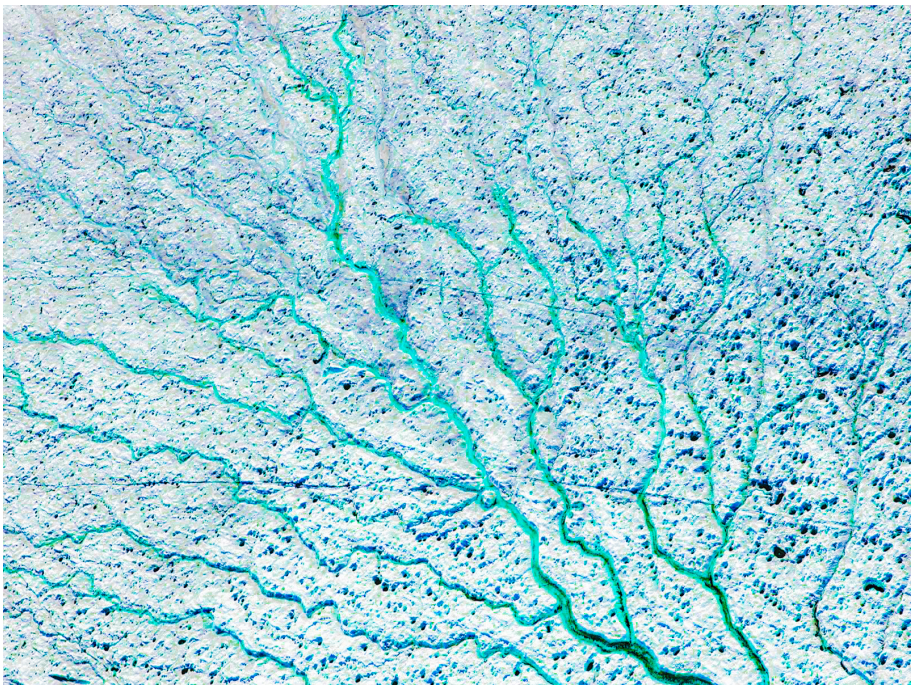
Kod źródłowy projektu został napisany w języku programowania Python. Używano biblioteki OpenCV do przetwarzania obrazów w czasie rzeczywistym. W dalszej części raportu opisano kolejne etapy przetwarzania obrazu wejściowego. Na końcu każdego rozdziału umieszczono obraz będący efektem danej części kodu.

1 Analiza kolorystyczna

Obraz wejściowy przedstawiono na stronie tytułowej raportu. Gołym okiem można z łatwością odróżnić strumyki – są ciemniejsze niż lód. Jednak na zdjęciu widać także inne ciemne punkty, które nie należą do sieci strumyków. Dlatego aby wyodrębnić wodę na zdjęciu, zdecydowano się wzmocnić jej niebieski kolor.

Zdjęcie zostało zmodyfikowane w programie do obróbki graficznej Lightroom. Wykorzystano narzędzie *hue-saturation-lightness*, które transformuje obraz do przestrzeni barw HSL. W tym modelu barwa wyraża się przez trzy liczby – odcień (*hue*), nasycenie (*saturation*) i jasność (*lightness*). Odcień odpowiada czystym i nasyconym kolorom – bez domieszek szarości, przyciemniania lub rozjaśniania. HSL jest bardzo intuicyjnym modelem opisu kolorów. Wystarczy wybrać określony odcień z koła barw (H), a następnie odpowiednio sterować nasyceniem (S) i jasnością (L), aby uzyskać pożądaną barwę.

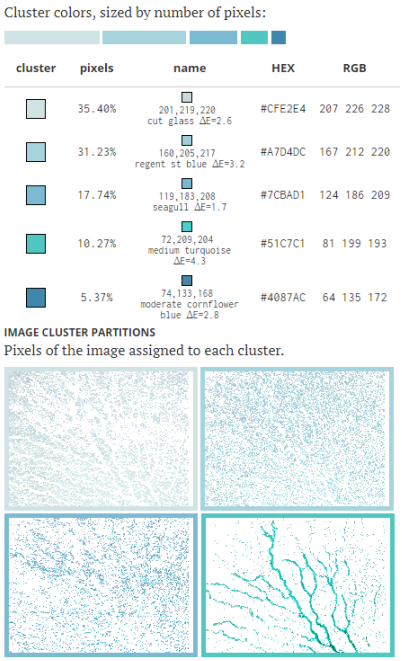
Wykorzystane narzędzie separuje odcień na sześć kanałów: czerwony, żółty, niebieski, magenta, zielony oraz cyjan. Dzięki temu możliwe było zwiększenie intensywności koloru rzeki w stosunku do koloru lądu/lodu i tym samym separacja obu obiektów w przestrzeni kolorowej.

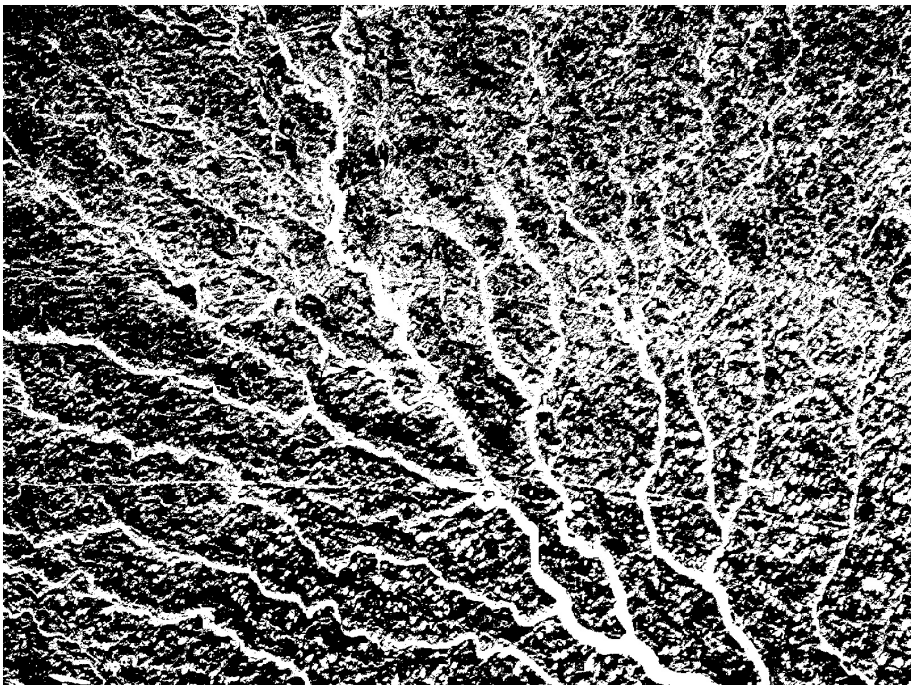


2 Przygotowanie obrazu binarnego – *thresholding*

Kolejnym krokiem była zamiana kolorowego zdjęcia na obraz binarny. Jest to rodzaj obrazu cyfrowego, którego piksele mogą przybierać tylko dwie wartości (w tym przypadku: 0 – czerń i 255 – biel). W tym celu stosuje się algorytmy progowania (ang. *thresholding*). W uproszczeniu polegają one na wyznaczeniu dla danego obrazu progu jasności. Następnie piksele jaśniejsze od wyznaczonego progu otrzymują jedną wartość, a ciemniejsze drugą.

Aby wyodrębnić zaznaczone na niebiesko strumyki, wykorzystano funkcję *inRange* z biblioteki OpenCV realizującą progowanie oparte na kolorach. Funkcja wyodrębnia na obrazie punkty o kolorze należącym do przedziału zadanego dwoma kolorami granicznymi (w modelu RGB). Aby dowiedzieć się jakie kolory dominują na obrazie, zastosowano narzędzie online *image color summarizer*, które bazując na algorytmie centroidów podaje klastry kolorów widocznych na rysunku. Wynik działania narzędzia umieszczono na rysunku po prawej. Spośród widocznych pięciu klastrów wybrano trzy, które nie odpowiadają strumykom. Następnie wykorzystując funkcję *inRange*, wybraną część obrazu zamalowano na czarno, a pozostałą część na biało – w ten sposób otrzymano obraz binarny.



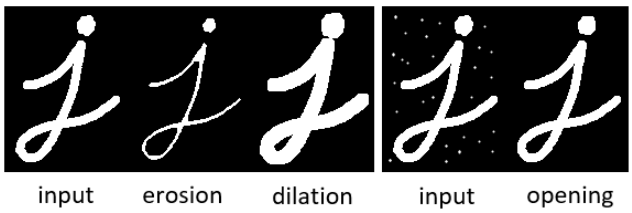


3 Operacje morfologiczne – otwarcie

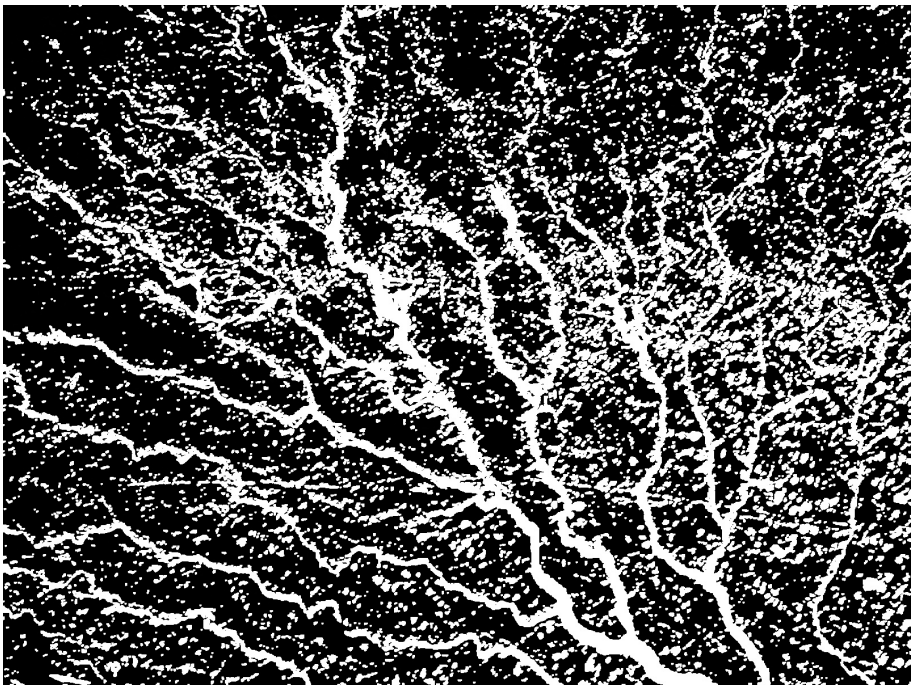
Poprzedni krok pozwolił na wykorzystanie metod cyfrowego przetwarzania obrazów binarnych, a w szczególności operacji morfologicznych. Przekształcenia morfologiczne wywodzą się z morfologii matematycznej – działu matematyki opartego na teorii zbiorów. Ta rodzina funkcji pobiera dwa argumenty – przetwarzany obraz binarny oraz element strukturalny, czyli podstawowy kształt określający działanie funkcji.

Podstawowymi operacjami morfologicznymi są erozja i dylatacja. W algorytmie erozji element strukturalny przesuwa się po obrazie. Piksel obrazu wejściowego będzie biały tylko wtedy, gdy wszystkie piksele pod elementem strukturalnym będą białe. W przeciwnym razie zostanie erodowany (ustawiony jako czarny). Algorytm dylatacji jest przeciwny do erozji – piksel będzie ustawiony jako biały, jeśli co najmniej jeden piksel pod elementem strukturalnym będzie biały.

Algorytm otwarcia jest złożeniem erozji i następującej po niej dylatacji. W ten sposób eliminuje się małe struktury na obrazie, dlatego otwarcie jest wykorzystywane w usuwaniu szumów. Na poniższym rysunku przedstawiono działanie algorytmów erozji, dyfuzji oraz otwarcia.

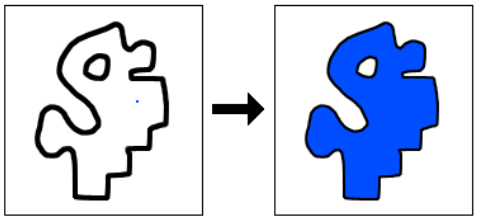


Na analizowanym obrazie otwarcie usunęło część małych białych struktur, które nie są rzekami.



4 Algorytm wypełniania *flood fill*

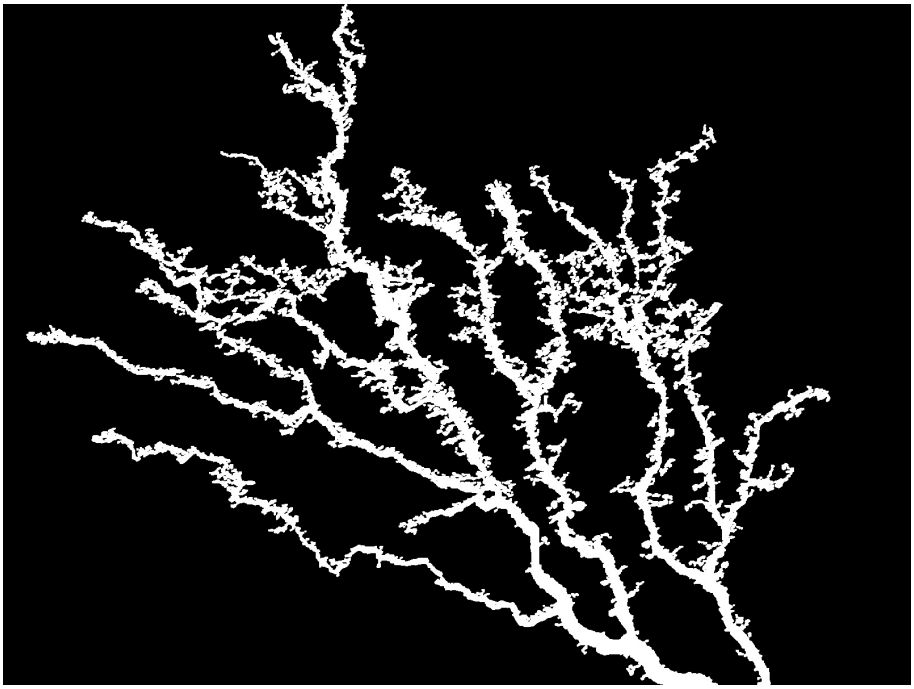
Aby odseparować sieć rzeczną od pozostałych struktur zaznaczonych na biało, zastosowano algorytm *flood fill*, który służy do wypełniania zamkniętych obszarów bitmapy kolorem. Na poniższym rysunku przedstawiono jego działanie.



Flood fill pobiera trzy parametry: początkową pozycję, zamieniany kolor i nowy kolor. Jest algorytmem rekurencyjnym, którego działanie w pseudokodzie można zapisać następująco:

```
floodFill (pozycja, zamienianyKolor, nowyKolor)
      jeżeli (kolorNaPozycji(pozycja) = zamienianyKolor)
            ustawKolor(pozycja, nowyKolor);
            floodFill(lewo(pozycja),zamienianyKolor,nowyKolor);
            floodFill(prawo(pozycja),zamienianyKolor,nowyKolor);
            floodFill(góra(pozycja),zamienianyKolor,nowyKolor);
            floodFill(dół(pozycja),zamienianyKolor,nowyKolor);
```

Na analizowanym obrazie wywołano algorytm flood fill dwa razy, jako punkty początkowe przyjmując środki strumyków na dole obrazu. Dwa powstałe obrazy dodano i otrzymano odszumiony obraz binarny.

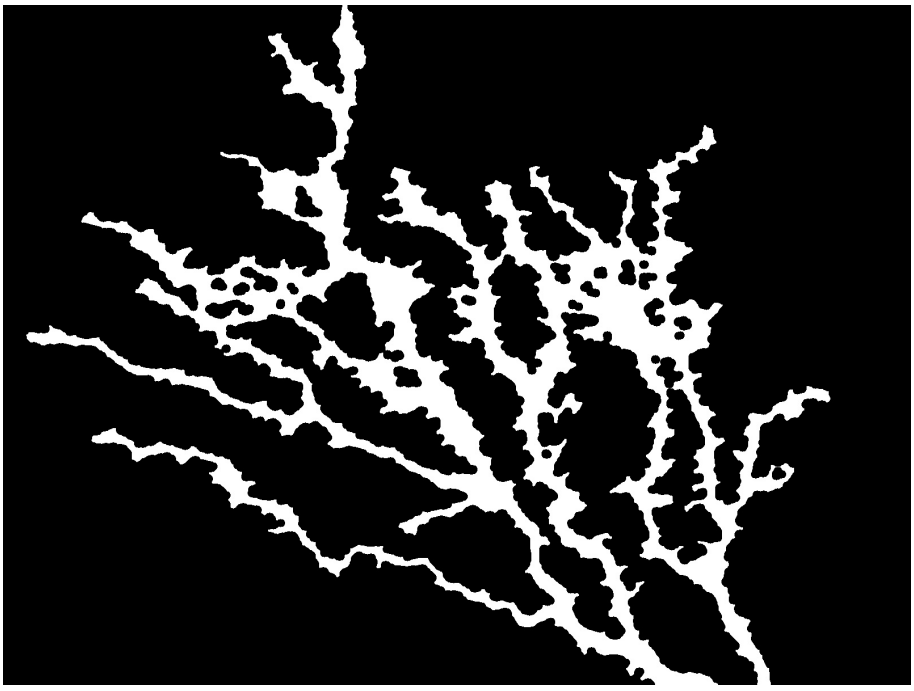


5 Operacje morfologiczne – domknięcie

Ponieważ obraz otrzymany w wyniku algorytmu wypełniania miał wiele małych rozgałęzień, które nie były strumykami, przeprowadzono jeszcze jedną operację morfologiczną – domknięcie. Składa się ona z dylatacji, po której następuje erozja. To przekształcenie domyka małe nieciągłości kształtu. Jego działanie przedstawiono na poniższym rysunku.

Na analizowanym obrazie przeprowadzono domknięcie. Jako element strukturalny przyjęto koło o średnicy 15 pikseli. Dla porównania, w algorytmie otwarcia użyto koła o średnicy 4 pikseli:

```
0 0 0 0 0 0
0 0 1 1 0 0
0 1 1 1 1 0
0 1 1 1 1 0
0 0 1 1 0 0
0 0 0 0 0 0
```



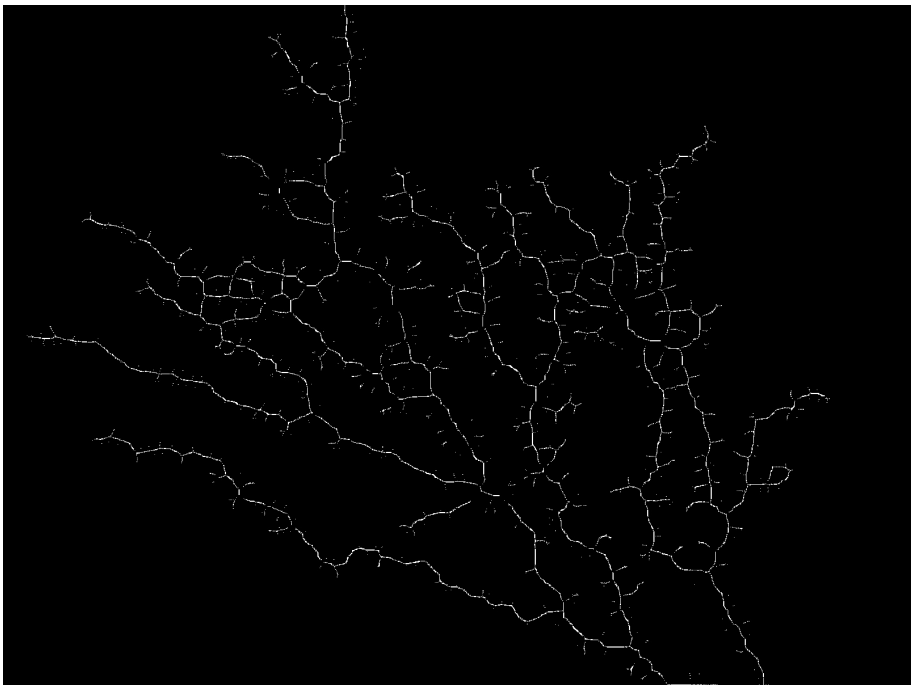
6 Szkieletyzacja

Kolejnym krokiem było pocienienie otrzymanego kształtu sieci strumyków. Celem było otrzymanie szkieletu, który zachowuje kształt pierwotnego obiektu, ale nie zawiera zbędnych pikseli.

Aby przeprowadzić szkieletyzację, wystarczą dwie podstawowe operacje morfologiczne – erozja i dylatacja. W pseudokodzie algorytm ma postać

```
img = ...
while notEmpty(img)
      skel = skel | (img & !open(img))
      img = erosion(img)
```

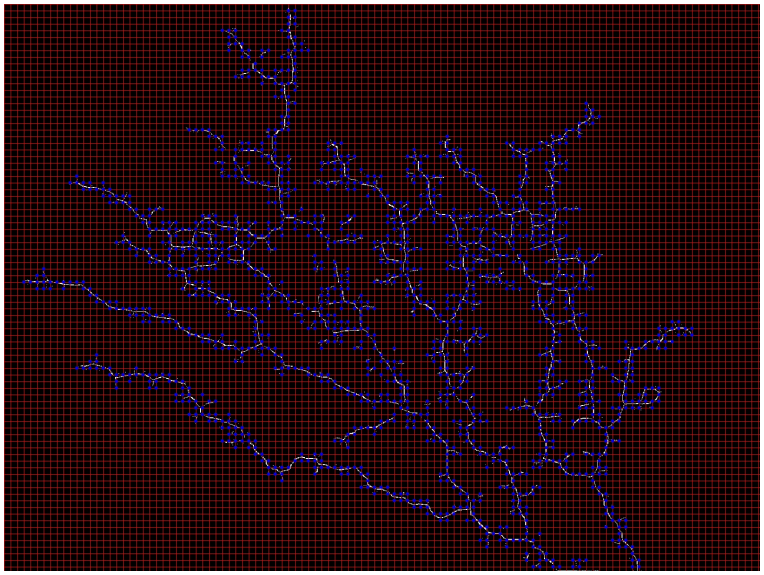
W każdej kolejnej iteracji kształt jest pocieniany przez erozję, a do szkieletu jest dodawana różnica między aktualną erozją a jej otwarciem. Pętla zatrzymuje się, gdy po kolejnej erozji kształt znika. Poniżej przedstawiono działanie algorytmu na przykładzie napisu "OpenCV" oraz szkielet sieci strumyków.



7 Zmniejszanie rozdzielczości

Następnym problemem, jaki pojawił się po uzyskaniu szkieletu, była jego reprezentacja w programie. Aby móc swobodnie operować na sieci (m.in. wyszukiwać rozwidlenia), wygodnie byłoby przechowywać jej punkty w odpowiedniej strukturze danych, np. drzewie.

Aby zmniejszyć liczbę punktów sieci strumyków, a więc "zdyskretyzować" rzekę, postanowiono wybrać z niej punkty o współrzędnych leżących na siatce o zadanych wymiarach. Na obraz szkieletu naniesiono kwadratową siatkę o boku długości 14 pikseli (zaznaczona na czerwono). Dla każdego węzła siatki sprawdzano, czy w jego okolicy znajduje się biały punkt. Jeśli tak, zaznaczono ten węzeł na niebiesko.



Na drugim rysunku przedstawiono same zaznaczone węzły siatki. W ten sposób uzyskano punkty, którymi można łatwo operować w programie. Kolejnym, kluczowym krokiem powinno być zapisanie ich w postaci drzewa. Nie jest jednak oczywiste jak to zrobić, ponieważ otrzymana sieć rzeczna ma grubość większą niż jeden punkt, zatem nie da się na tym etapie uszeregować punktów w odpowiedniej kolejności lub znaleźć rozgałęzień. Aby doprowadzić do postaci drzewa, należy poprawić dotychczasową procedurę lub dodać kolejne kroki – niestety nie udało się tego zrobić ze względu na ograniczenia czasowe.