Przegląd dostępnych frameworków, bibliotek (free/open source) dla Javy oraz adaptacja wybranych do implementacji aplikacji wspomagającej obrazową diagnostykę medyczną

Kraków 2017

1. Wstęp

Java jako język programowania może zostać wykorzystana jako narzędzie wspomagające diagnostykę medyczną, pod warunkiem posiadania odpowiednich bibliotek. Praca przedstawia przegląd dostępnych, opartych na licencji open source. Zaprezentowane zostaną 3 przykładowe biblioteki: dcm4che, OpenCV oraz Bio-Formats. Zawierają one wiele narzędzi i algorytmów pozwalających na przetwarzanie obrazów medycznych i nie tylko, dodatkowo bowiem mogą usprawniać pracę, przykładowo dokonując konwersji formatów. Ze względu na dużą popularność, biblioteki są regularnie poprawiane oraz dodawane są nowe funkcje, które pozwalają na jeszcze bardziej złożoną analizę danych medycznych. Ogólnie trudno uznać Javę za środowisko dedykowane do przetwarzania i analizy danych medycznych. Mimo tego jest powszechnie stosowana do zapisu, odczytu i konwersji różnych formatów w jakich przechowuje się te dane. Głównymi tego powodami są popularność środowiska, duża ilość bibliotek obsługujących macierze, przydatnych szczególnie przy konwersji oraz fakt, że Java jest niezależna od platformy, co nadaje jej bardzo szeroką elastyczność.

2. Biblioteka OpenCV

OpenCV (Open Source Computer Vision Library) to biblioteka, która zawiera kilkaset algorytmów do przetwarzania obrazów. Ma ona strukturę modularną, co oznacza, że pakiet zawiera oddzielne, funkcjonalne biblioteki. Jej mocną stroną jest wydajność obliczeniowa, a szczególny nacisk położono na aplikacje działające w czasie rzeczywistym. Dzięki zastosowaniu frameworka OpenCL, który wspomaga pisanie aplikacji na platformach składających się z różnego rodzaju jednostek obliczeniowych [1]. Wykorzystanie do projektów biblioteki OpenCV może być bardziej efektywne niż zastosowanie szybszego procesora lub karty graficznej. Z biblioteki korzysta około 47 tysięcy ludzi z całego świata. Poniżej lista dostępnych modułów:

- **core** – kompaktowy moduł definiujący podstawowe struktury danych, w tym wielowymiarową tablicę Mat;
- **imgproc** – moduł do przetwarzania obrazów, który pozwala na liniową i nieliniową filtrację, transformacje geometryczne, konwersję przestrzeni kolorów, wykreślanie histogramów itp.;
- **video** – analiza materiałów video, która zawiera ocenę ruchu, odcięcie tła oraz wyznaczanie toru ruchu obiektów**;**
- **calib3d** – podstawowe algorytmy geometryczne, estymacja pozycji obiektu, kalibracja widoku panoramicznego, elementy rekonstrukcji 3D
- **features2d** – detekcja charakterystycznych cech, identyfikatorów pliku;
- **objdetect** – wykrywanie obiektów takich jak twarz, oczy, uszy, ludzie, samochody itp.;
- **highgui** – łatwy w użyciu interfejs do przechwytywania wideo;
- **gpu** – algorytmy dla procesorów graficznych;

Poniżej przedstawiono przegląd niektórych zaimplementowanych funkcji oraz ich przykładową realizację.

Konwersja do skali szarości

Jest to najczęściej etap wstępny do dalszej analizy obrazów. W celu konwersji koloru obrazu do skali szarości należy odczytać wszystkie piksele obrazu używając obiektów **File** oraz **ImageIO**, a następnie przechować je w obiekcie **BufferedImage.** Następnie należy odczytać wartość pikseli wykorzystując metodę **getRGB()** i wykonać metodę **GrayScale()**. Wartości kolorów uzyskuje się przy pomocy klasy Color przypisując do utworzonych zmiennych wyniki uzyskane przez metody **getRed()**, **getGreen()** i **getBlue()** [2]. Inne metody znajdujące się w klasie Color, to:

- **brigther()** – tworzy nową klasę Color, która jest jaśniejszą wersją poprzedniej,
- **darker()** – tworzy nową klasę Color, która jest ciemniejszą wersją poprzedniej,
- **getAlpha()** – zwraca komponent alfa (głównie używany jako odpowiednik współczynnika pochłaniania światła) z zakresu 0-255,
- **getHSBColor(float h, float s, float b)** – tworzy nową klasę Color opartą na wartościach specyficznych dla przestrzeni barw HSB,
- **HSBtoRGB(float hue, float saturation, float brightness)** – konwertuje składowe koloru właściwe dla przestrzeni barw HSB na odpodwiadające im wartości w przestrzeni barw HSB
- **toString()** – zwraca reprezentację klasy Color w ciągu znaków

Zwiększanie ostrości obrazu

Polepszanie ostrości obrazu odbywa się za pomocą filtru gaussowskiego zaimplementowanego w funkcji **GaussianBlur**, którą można znaleźć w pakiecie **imgproc** [2]. Jej składnia wygląda następująco:

```
Imgproc.GaussianBlur(source, destination, new Size(0,0), sigmaX);
```

Opis parametrów:

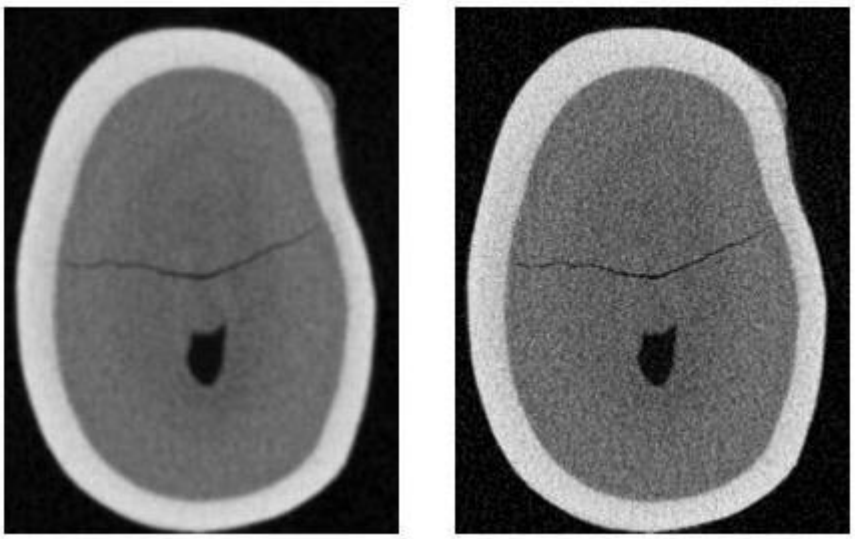
- **source** – obraz źródłowy,
- **destination** – obraz docelowy,
- **Size** – wielkość jądra gaussowskiego,
- **sigmaX** – odchylenie standardowe jądra gaussowskiego.

Następnie należy użyć funkcji **addWeighted**, aby dodać znaki wodne do obrazu. Jej składnia została pokazana poniżej:

```
Core.addWeighted(InputArray src1, alpha, src2, beta, gamma, OutputArray dst);
```

Opis parametrów:

- **src1** – pierwsza wejściowa tablica,
- **alpha** – waga elementów pierwszej tablicy,
- **src2** – druga wejściowa tablica takiej samej wielkości co tablica src2,
- **beta** – waga elementów drugiej tablicy,
- **gamma** – wielkość skalarna dodawana do każdej sumy,
- **dst** – tablica wyjściowa takiej samej wielkości co tablice wejściowe.



Rys. 1. Zwiększenie ostrości z wykorzystaniem powyższych funkcji

Progowanie

Progowanie pozwala uzyskać segmentację obrazu w najprostszy sposób. Segmentacja to podzielenie obrazu na zbiory pikseli w taki sposób, że piksele danego zbiory mają określone cechy charakterystyczne. Jest ona bardzo pomocna w rozpoznawaniu elementów na obrazie oraz ich granic.

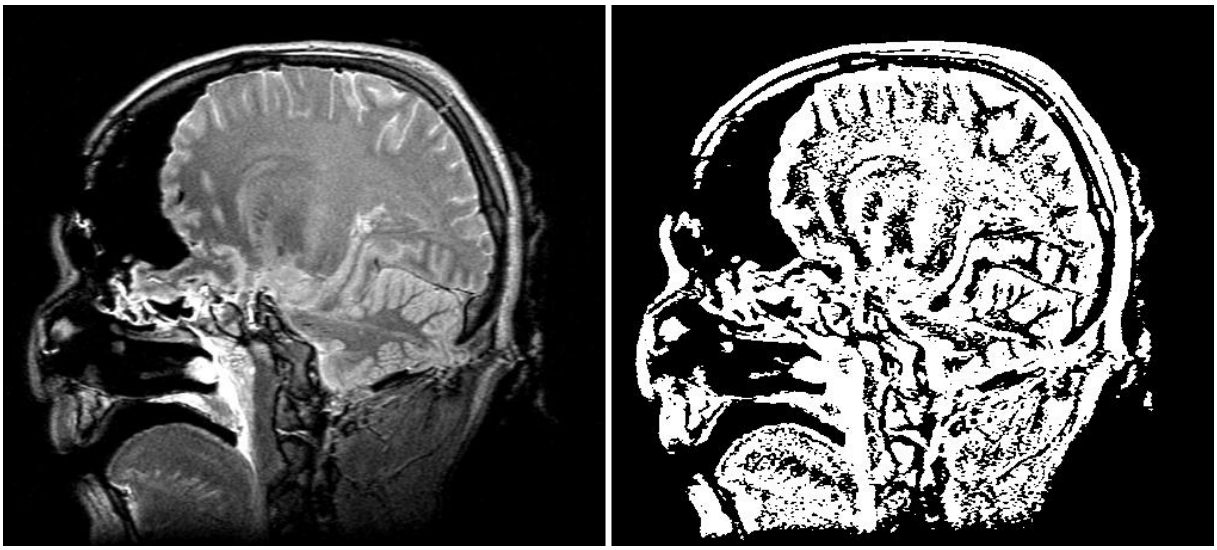
Wykorzystano funkcję **treshold**, którą można znaleźć w pakiecie **imgproc** [2]. Jej składnia wygląda następująco:

Imgproc.threshold(source, destination, thresh , maxval , type);

Opis parametrów:

- **source** – obraz źródłowy,
- **destination** – obraz po przetworzeniu,
- **tresh** – wartość liczbowa progu,
- **type** – możliwe rodzaje tego parametru to TRESH_BINARY, TRESH_BINARY_INV, TRESH_TRUNC, TRESH_TOZERO.

Wykorzystano funkcję wymienioną powyżej i otrzymano następujące rezultaty:



Rys. 2. Progowanie binarne

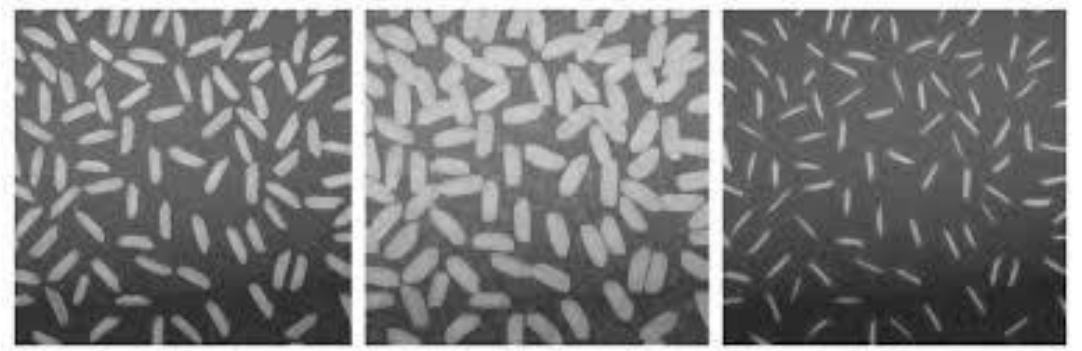
Przekształcenia morfologiczne

Są to przekształcenia, w wyniku których struktura obrazu zostaje zmieniona. Podstawowymi operacjami są erozja i dylatacja. Pierwsza z nich polega na obcinaniu brzegów obiektu, druga z kolei na zamykaniu małych otworów oraz zatok we wnętrzu figury. Biblioteka OpenCV posiada zaimplementowane funkcje erode i dilate, które znajdują się w pakiecie imgproc [1]:

```
Imgproc.erode(source, destination, element);
Imgproc.dilate(source, destination, element);
```

Opis parametrów:

- **source** – obraz źrodłowy,
- **destination** – obraz po przetworzeniu,
- **element** – element strukturalny (jeśli tym elementem jest Mat() – wykorzystany zostaje prostokątny element 3x3).



Rys 3. Obraz źródłowy (w środku) poddany erozji (z lewej) i dylatacji (z prawej).

Operator Prewitta

Jest on wykorzystywany w detekcji krawędzi pionowych i poziomych na obrazie. Aby tego dokonać, należy wykorzystać funkcję **filter2D** dostępną w pakiecie imgproc. Jej składnia została przedstawiona poniżej:

filter2D(src, dst, ddepth , kernel, delta);

Opis argumentów [1]:

- **src** – obraz źródłowy,
- **dst** – obraz po przetworzeniu,
- **ddepth** – głębokość obrazu po przetworzeniu. Jeśli występuje wartość ujemna, głębokość dst jest taka sama jak źródła,
- **kernel** – jądro przekształcenia, które zostanie wykorzystane do operacji dwuwymiarowego dyskretnego splotu,
- **delta** – wartość dodana do każdego piksela podczas splotu. Domyślnie jest nią 0.

Dla detekcji krawędzi pionowych i poziomych definiowane są różne jądra przekształcenia. Jądro dla detekcji pionowej i poziomej:

$$ver = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad hor = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

Rezultat wykonania powyższej funkcji przedstawiony został poniżej:

Rys. 4. Detekcja krawędzi

Kompresja obrazów

Obraz można łatwo skompresować i przechowywać. Kompresja oznacza konwersję obrazu do formatu jpg. Wykorzystana w tej operacji klasa to ImageIO. Następnie wykorzystuje się **ImageWriter** z metody **getImageWriterByFormatName()**. Z ImageWriter'a możemy utworzyć obiekt **ImageWriteParam**. Składnię przedstawiono poniżej:

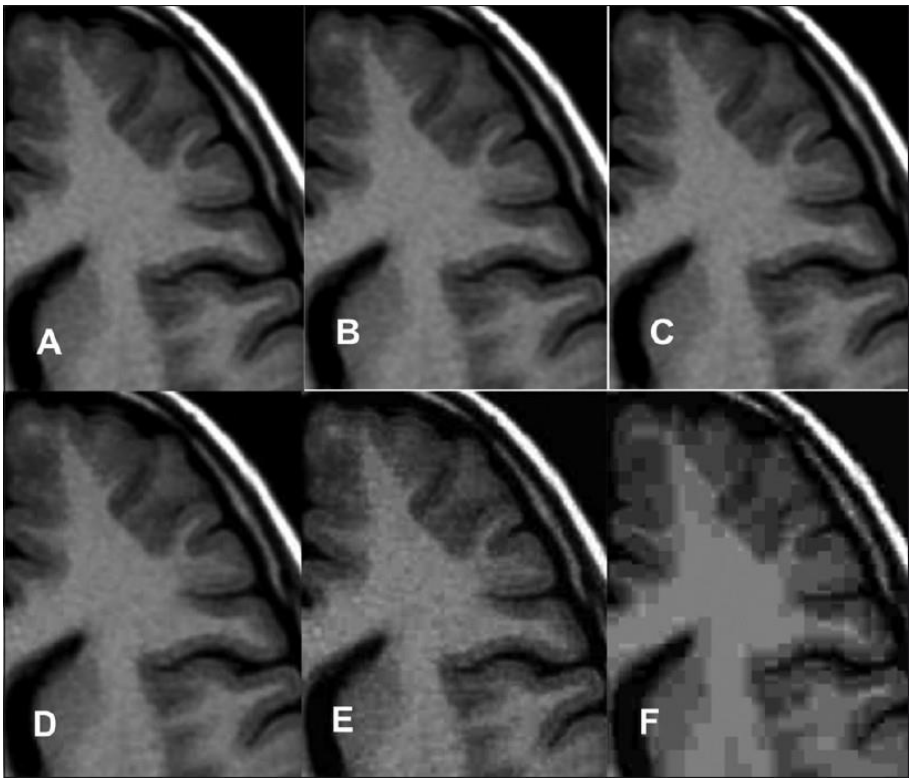
```
Iterator<ImageWriter> list = ImageIO.getImageWritersByFormatName("jpg");
ImageWriteParam obj = writer_From_List.getDefaultWriteParam();
```

Z obiektu ImageWriteParam można ustawic kompresję wywołując metody setCompressionMode() I setCompressionQuality(). Ich składania wygląda następująco:

```
obj.setCompressionMode(ImageWriteParam.MODE_EXPLICIT);
obj.setCompressionQuality(0.05f);
```

Metoda setCompressionMode() zawiera różna parametry, niektóre zostały opisane poniżej [2]:

- **MODE_DEFAULT** – obraz będzie taflowy, progresywny lub skompresowany zgodnie z tym jak zostanie to ustawione przez programistę
- **MODE_DISABLED** – obraz nie będzie taflowy, progresywny ani skompresowany,
- **MODE_EXPLICIT** – obraz będzie skompresowany bądź taflowy zgodnie z dodatkowymi informacjami dostarczonymi w metodzie set wewnątrz tej samej klasy.



Rys. 5. Kompresja obrazu z coraz większym współczynnikiem kompresji (quality factor)

3. Biblioteka dcm4che

Biblioteka dcm4che jest stworzona w taki sposób, by mogła być stosowana na sprzęcie różnego rodzaju i wieloma typami oprogramowania. Została napisana w Javie i rozpowszechniona jako wbudowany komponent w serwerze Java Enterprise Edition (JEE). Narzędzie zostało wdrożone na kilka systemów operacyjnych [3]:

- Microsoft Windows,
- Linux,
- Apple OSX,
- Sun Solaris,
- IBM AIX.

Dcm4che wykorzystuje bazę danych w celu przechowywania informacji z nagłówków DICOMowych, indeksuje dane tak aby można je było łatwo zlokalizować, a także posiada inne ważne funkcje potrzebne w zarządzaniu danymi medycznymi. Bazy danych które są wykorzystane w projekcie, to m.in. MySQL, Oracle, SQL Server, Firebird. Narzędzie posiada rozbudowany interfejs użytkownika, który pozwala na wszystkie niezbędne opcje potrzebne osobie administrującej systemem, takie jak dodanie nowego pacjenta, wyszukiwanie, eksport danych, edycję danych itp. Biblioteka zawiera wiele przydatnych przykładowych aplikacji, które można wykorzystać w połączeniu z innymi bibliotekami.

Konwersja obrazu DICOM do formatu XML

Funkcja dcm2xml pozwala na konwersję obrazu DICOM do formatu XML oraz opcjonalnie do XLS. Wartości cech określone przez –x <tag> są wyłączone z wygenerowanego pliku. Atrybut –o <xmlfile> umożliwia przechowywanie wyłączonych cech i zapisanie ich w tym samym folderze co plik XML. Pliki z wyodrębnionymi wartościami zagnieżdżonych atrybutów są przechowywane w podfolderach. Nie wpisując –o <xmlfile>, ale dodając –d <basedir> wyłączone wartości będą przechowywane w folderze basedir. Jeśli nie zostanie użyte ani –o <xmlfile>, ani –d <basedir>, wyłączone wartości nie będą przechowywane [3]. Przykładowa komenda:

```
dcm2xml -Xi image.dcm -o image.xml
```

gdzie –Xi oznacza wyłączenie informacji o pikselach.

Narzędzie dcmecho

Dcmecho jest programem wiersza poleceń, które wykonuje komendę C-ECHO jako klient (Service Class User) w kierunku serwera (Service Class Provider). Celem komendy C-ECHO jest weryfikacja, czy serwer jest aktywny i czy przyjmuje asocjacje. To narzędzie prześle Echo standardu DICOM do określonej zdalnej jednostki aplikacji. Jeśli <port> nie jest sprecyzowany, brany jest domyślny port 104. Jeśli <host> nie jest sprecyzowany, przyjmowany jest localhost [3]. Składnia komendy:

```
dcmecho [Options] <aet>[@<host>[:<port>]]
```

Niektóre opcje to:

- repeat <num> - powtarzanie echa kilka razy,
- reuseassoc – ponowne użycie asocjacji dla powtórzonego C-ECHO,
- closeassoc – zamknięcie asocjacji po każdym C-ECHO,

Przykład użycia:

```
dcmecho STORESCP@localhost:11112
```

Narzędzie dcmsnd

Ta aplikacja funkcjonuje jako miejsce przechowywania obrazów klienta (SCU) i wysyłania ich do serwera (SCP). Załadowuje złożone obiekty w standardzie DICOM z określonych plików lub folderu i wysyła je do zdalnej aplikacji (AET). Jeżeli folder jest sprecyzowany, wszystkie obiekty w standardzie DICOM z tego folderu i jego podfolderów zostaną wysłane. Jeśli <port> nie jest określony, przyjmowany jest domyślny port 104. Jeśli

<host> nie jest określony, przyjmowany jest localhost. Komendy używa się w następujący sposób [3]:

```
dcmsnd [Options] <aet>[@<host>[:<port>]] <file>|<directory>
```

Przykład wykorzystania:

```
dcmsnd DCMRCV@localhost:11112 image.dcm
```

Oto niektóre opcje:

- fileref – wysyła obiekty bez danych na temat pikseli, ale z odnośnikiem do pliku DICOM używającego dcm4che,
- async <maxops> - maksymalna liczba zaległych operacji, domyślnie nielimitowana,
- connectTO <ms> - koniec czasu dla połączenia TCP, domyślnie brak limitu,
- truststorepw <password> - hasło dla zaufanych plików,
- highprior – wysoki priorytet dla operacji C-STORE, domyślnie średni (operacja C-Store pozwala na przesłanie wiadomości z SCU do SCP).

Narzędzie dcmrcv

Wykonanie komendy spowoduje uruchomienie serweru DICOM listener na określonym porcie. Jeżeli żaden adres IP sieci lokalnej nie jest uwzględniony, połączenia na wszystkie adresy lokalne są akceptowane [3]. Użycie funkcji:

```
dcmrcv [Options] [<aet>[@<ip>]:]<port>
```

Przykład użycia:

```
dcmrcv DCMRCV:11112 -dest /tmp
```

Niektóre opcje to:

- dest <dir> - przechowywanie otrzymanych obiektów w plikach znajdujących się w określonym folderze <dir>,
- defts – akceptuje tylko domyślną składnię,
- native – akceptuje tylko nieskompresowane wiadomości.

Współpraca ze standardem HL7

Omawiana biblioteka pozwala na przesyłanie danych pacjenta w standardzie HL7. Narzędzie do przesyłania wiadomości w standardzie HL7 (HL7 Send Service) może być skonfigurowane w ten sposób, aby przesyłać dalej wiadomości otrzymane przez Serwer HL7

do innych systemów. Można także wykorzystać je do wysłania zapytania, aby „wylegitymować” pacjenta w innych domenach, o ile w nich istnieje [3].

4. Biblioteka Bio-Formats

Bio-Formats jest biblioteką służącą do odczytu i zapisu oprazów, obsługującą przede wszystkim formaty stosowane w naukach biologicznych. Najważaniejszym celem jest ułatwienie przesyłania, wymiany danych w postaci zdjęć mikroskopowych pomiędzy różnymi ośrodkami medycznymi i naukowymi. Osiąga to poprzez konwersję danych mikroskopowych do otwartego standardu OME data model, dokładnie do formatu OME-TIFF. Najważniejszym aspektem standardu są metadane, a dokładnie ich ujednolicenie. Metadane - szczegółowe informacje dotyczące danego pliku, mogą być bardzo złożone i zapisane w różnym formacie w zależności na przykład od producenta urządenia, z którego pozyskano dane. Bio-Formats, konwertuje metadeane, do jednego standardowego formatu (OME data model).

OME-TIFF został stworzony w celu maksymalizacji mocnych stron formatu OME-XML (również służącego do przechowywania metadanych) oraz TIFF (jeden z popularnych formatów obrazów).

Obecnie biblioteka wspiera 144 różne formaty plików, niestety w różnym stopniu. Oznacza to, że nie każda funkcjonalność jest dostępna dla wszystkich rodzajów plików, a nawet jeżeli, to nie odbywa się to jednakowo niezawodnie. Przykładowo format .tiff (Tagged Image File Format), wspierany jest dla wszystkich (chociaż w różnym stopniu zaawansowania) operacji. Kontrastem może być format .lim (Laboratory Imagong/Nikon), obsługiwany bardzo słabo, lub wcale. Pełną listę wraz z opisami formatów można znaleźć na stronie projektu [4]. Co ciekawe, możliwe jest zarówno wczytywanie kolejno poszczególnych obrazów, jak też wszystkich jednocześnie, również przy użyciu do tego różnych wątków. Podobnie wygląda sytuacja z zapisem plików [4].

5. Aplikacja

Podczas pisania aplikacji napotkano na znaczny problem dotyczący frameworku dcm4che, który bardzo poważnie ogranicza go na niektórych systemach operacyjnych. Biblioteka imageio, służąca do odczytywania i zapisaywania obrazów rróżnych formatów, składa się z części Java oraz native ($DCM4CHEE_HOME/bin/native/). Niestety ta ostatnia istnieje i dział tylko w następujących systemach operacyjnych:

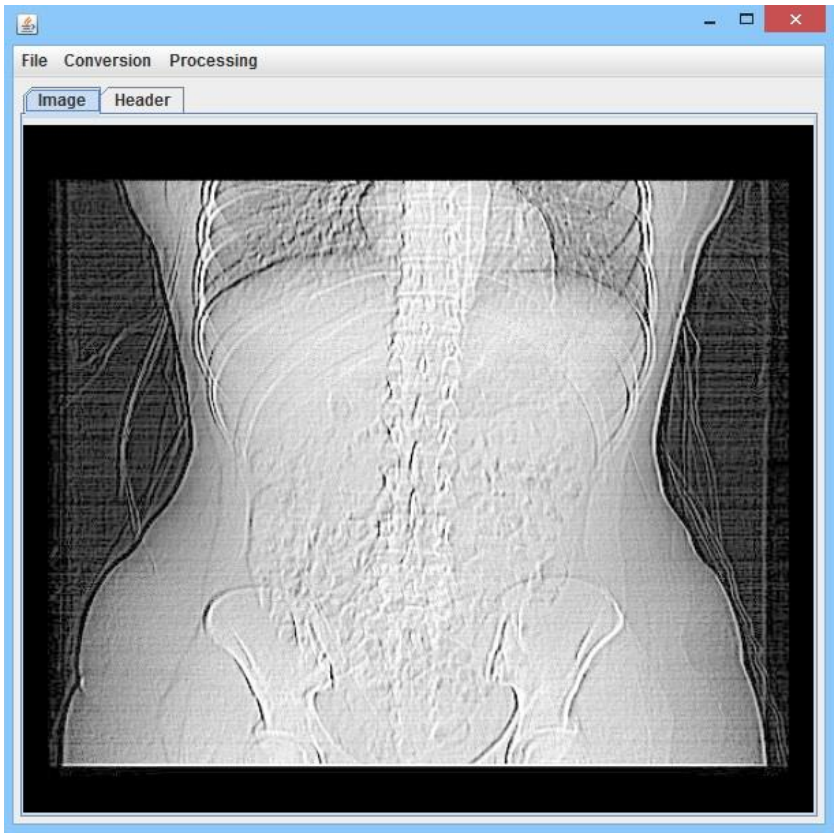
- Windows 32-bit
- Linux x86 32 i 64-bit
- Solaris

Aplikacja była pisana z kolei na systemie Windows 8 64 bitowym. Stąd też część enkoderów i dekoderów należących do części native nie jest obsługiwana, co skutkuje brakiem możliwości wczytania części plików dcm [3]. Niestety jest to problem dotyczący samego framewrka i pomimo licznych prób związanych między innymi z instalacją 32-bitowego Java development toolkit, nie udało się go rozwiązać.

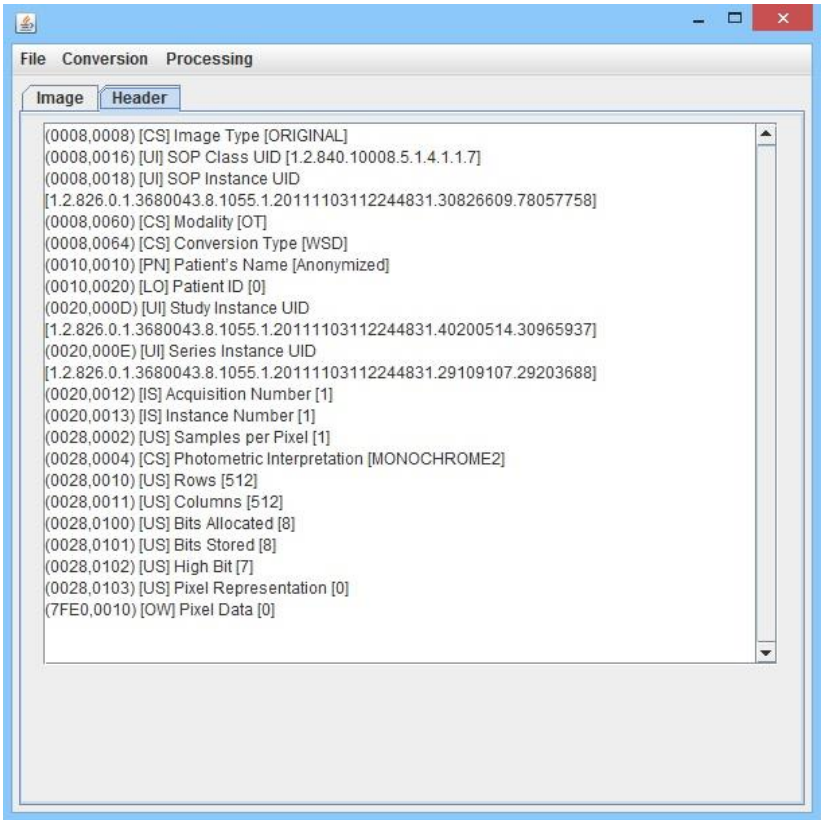
Pokazuje to bardzo istotną wadę samego frameworka, która powinna zostać w przyszłości wyeliminowana, gdyż na chwilę obecną stanowi poważne ograniczenie.

Założenia ogólne

Stworzona aplikacja ma na celu demonstrację przykładowych możliwości wykorzystywania dostępnych frameworków w analizie i przetwarzaniu obrazów medycznych. Została w całości napisana w środowisku Java, z wykorzystaniem frameworków opencv i dcm4che oraz narzędzia WindowBuilder do stworzenia graficznego interfejsu użytkownika. Widok ogólny głównego okna aplikacji przedstawia Rysunek 1 (zakładka Image) i Rysunek 2 (zakładka Header).



Rys. 6. Główne okno aplikacji wraz z wczytanym przykładowym obrazem DICOM



Rys.7. Główne okno aplikacji wraz z wyświetloną informacją o wczytanym obrazie (Header)

Aplikacja posiada kilka wbudowanych funkcji, prezentujących niektóre sposoby wykorzystywania zastosowanych toolkitów zarówno do analizy, jak również przetwarzania obrazów.

Opis funkcji

- Wczytywanie obrazu
  Program umożliwia wczytywanie i wyświetlanie obrazów w formacie dcm oraz jpg. W przypadku pierwszym, automatycznie pobierane są również informacje o obrazie i wyświetlane w zakładce „Header". Dla plików jpg, które informacji nie zawierają, zakładka pozostaje pusta.

- **Konwersja DICOM-jpg, jpg-DICOM**
  Format dcm nie jest formatem wygodnym do samego przeglądania obrazów, ponieważ większość standardowych przeglądarek nie jest w stanie poprawnie go odtworzyć. Z tego względu przydatnym może okazać się możliwość konwersji obrazu DICOM, do bardziej popularnego formatu jpg. Oczywiście, ogranicza to pewne możliwości jeśli chodzi o dalszą analizę, niemniej jednak w pewnych sytuacjach może okazać się przydatne, chociażby w przypadku, gdy znaczenie ma tylko sam obraz, a nie na przykład dodatkowe informacje które zawiera (header). Aplikacja umożliwia również konwersję odwrotną, to znaczy jpg-dcm. DICOM jest podstawową normą do przesyłania i interpretacji danych medycznych, stąd często może zachodzić konieczność konwersji obrazu z popularnego jpg właśnie do formatu dcm. Konwersja w prawdzie nie doda do zdjęcia informacji, które zawiera plik dcm, ale pozwoli na ujednolicenie i ustandaryzowanie formatu pliku.

- **Splot obrazu (Kernel convolution)** – filtracja, sprowadza się do przetworzenia obrazu w następujący sposób:
  - Przygotowanie tzw. jądra splotu (ang. kernel) o wielkości 3x3 piksele
  - Poszczególne piksele obrazu przeliczane są w ten sposób, że dla każdego piksela przemnażana jest każda wartość z kernela przez każdą wartość odpowiadającego temu pikselowi fragmentu obrazu o wielkości takiej jak kernel.
  - Sumowanie poszczególnych wyników
  - Jeżeli otrzymany w ten sposób wynik przekracza dopuszczalną wartość, można go zaokrąglić do odpowiedniej wartości granicznej
  - Ostateczny wynik umieszczony zostaje w wynikowej tablicy pikseli

  Filtracja stosowana może być jako metoda wydobycia z oryginalnego obrazu szeregu informacji jak przykładowo położenie krawędzi w celu ich dalszej obróbki. Innym zastosowaniem filtracji jest usuwanie szumów. Może mieć to praktyczne znaczenie

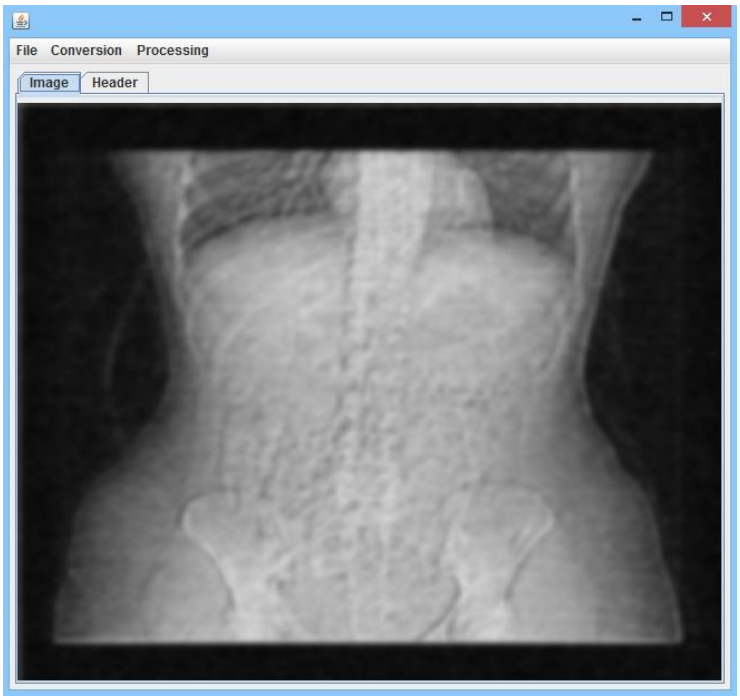
dla pozyskiwania użytecznych danych z obrazu złej jakości. Przykład użycia przedstwaiony został na Rysunku 3. Funkcjonalność został zeralizowana w całości przu użyciu toolkitu dcm4che. Zaletą tego jest fakt, że można jej dokonać bezpośrednio na obrazie typu dcm, bez konieczności konwersji. Wadą natomiast, że sam toolkit nie jest dobrze dostosowany do operacji na obrazach, które w jakiś sbosób by je zmieniały, a roczej do działania już na gotowych plikach. Stąd konieczne jest samodzielne wczytanie obrazu jako tablicy wartości reprezentujących konkretne piksele, zdefiniowanie jądra splotu (tu zastosowano filtr uwypuklający) itp. Niemneiej jednak przetwarzanie obrazów jest możliwe i daje zadowalające rezultaty.

Rys.8. Wynik filtracji obrazu. Widoczne znacznie lepiej zarysowane krawędziw w stosunku do Rysunku 1

- **Filtracja obrazu filtrem Gaussa.**
  Zastosowanie podobne do tego z poprzedniego punktu, różnica polega jednak na zastosowaniu konkretnego, zdefiniowanego filtru oraz zastosowanego toolkitu, gdyż ta funkcjonalność została zrealizowana w całości przy pomocy toolkitu opencv i prezentuje jedną z jego wielu możliwości dotyczących przetwarzania obrazów. Ogólnie w kwestii samej obróbki obrazów toolkit ten prezentuje się bardzo dobrze, posiadając wiele wbudowanych funkcji w postaci gotowych filtrów i operacji. Pozwala więc na wygodną i łatwą w implementacji analizę. Preoblemem jednak, jeżeli chodzi o zastosowania medeyczne jest fakt, że obsługuje on tylko najpopularniejsze formaty obrazów (jpg, png, tiff itp), nie można więc bezpośrednio obrabiać na nim plików

dcm, możliwe jest to jedynie po konwersji do jednego z odpowiednich formatów. Filtra Gaussa jest macierzą, w której jeżeli kolejne przedstawionoby za pomocą słupków o wysokości odpowiadającej przypisanej wadze to w efekcie otrzymanoby bryłę podobną do krzywej rozkładu normalnego - krzywej Gaussa. Tego typu działania dają dobre efekty przy filtracji obrazów zaszumionych szumem Gaussa, przykład dla maski 15x15 pokazuje Rysunek 4 i 5.

Rys.9. Obraz zaszumiony szumem Gaussa



Rys.10. Obraz przefiltrowany filtrem Gaussa

Powyższy przykład tylko pokazuje efekt działania, nie przykład praktycznego zastosowania. Obraz wynikowy jest bardziej zamazany, co jest zależne od wielkości maski.

- **Erozja i dylatacja**
  Erozja jest jednym z podstawowych przekształceń morfologicznych. Jej działanie polega na obcinaniu brzegów obiektu na obrazie. Dylatacja służy z kolei do zamykania małych otworów oraz zatok we wnętrzu figury. Obiekty zwiększają swoją objętość i jeśli dwa lub więcej obiektów położonych jest blisko siebie, zrastają się w większe obiekty. Obydwie operacje mogą mieć praktyczne znaczenie dla przetwarzania obrazów medycznych, szczególnie dla preparatów mikroskopowych

6. Podsumowanie

Obecnie istnieje wiele bibliotek i frameworków typu open source na potrzeby aplikacji medycznych, które dają programistom duże możliwości. Zawierają one pokaźną liczbę zaimplementowanych funkcji, które w znaczny sposób ułatwiają analizę i przesyłanie danych między ośrodkami. Szczególnie istotna jest jednolitość struktury danych w standardzie DICOM, który jest bogatą i złożoną normą stworzoną dla potrzeb interpretacji danych. Omówione biblioteki bardzo dobrze zachowują strukturę tego standardu mimo złożonych operacji wykonywanych na plikach. Można zauważyć, że wiele z omówionych funkcji jest zaimplementowanych także w innych środowiskach (np. MATLAB), jednakże zaletą tego typu bibliotek jest nie tylko ich darmowość, ale także możliwość modyfikacji kodu źródłowego na własny użytek.

Źródła:

1. http://opencv.org (odwiedzono 20.12.2016r.)
2. https://www.tutorialspoint.com/java_dip/introduction_to_opencv.htm (odwiedzono 20.12.2016r.)
3. http://www.dcm4che.org (odwiedzono 20.12.2016r.)
4. http://www.openmicroscopy.org (odwiedzono 20.12.2016r.)