Algorytmy przetwarzania i analizy obrazów medycznych

Raport projektowy: DicomVis – oprogramowanie do wizualizacji obrazów medycznych

Michał Kożuch

Inżynieria Biomedyczna, WEAIiIB, AGH

Reykjavik 2016

Wstęp i cel projektu

Celem projektu było stworzenie oprogramowania do wizualizacji serii obrazów DICOM w języku Python za pomocą bibliotek VTK i PyQt4.

Dane wejściowe stanowiła seria obrazów w formacie DICOM stanowiąca zapis badania MRI.

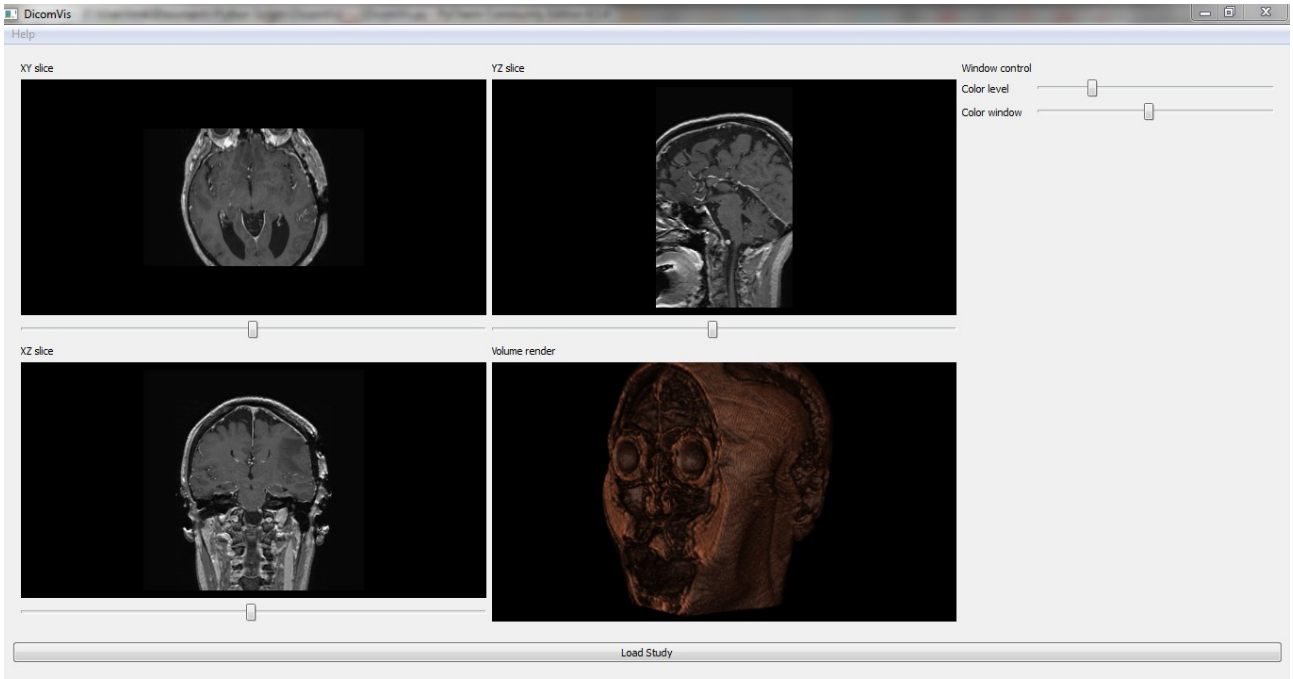
Założenia projektu

Założenia obejmowały wyświetlanie obrazu w trzech oknach reprezentujących przekroje w trzech płaszczyznach oraz widok 3D. Interakcja z aplikacja powinna zachodzić za pomocą suwaków umieszczonych na interfejsie, które umożliwiają zmianę aktualnie wyświetlanego przekroju oraz zakresu jasności pikseli. Ponadto architektura programu powinna umożliwić ponowne jego użycie jako widżetu Qt osadzonego w większej aplikacji.

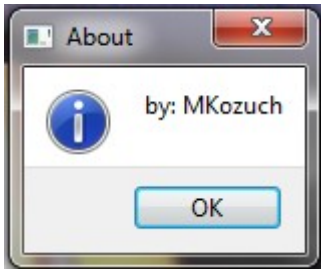
Realizacja

Opis aplikacji:

Aplikacja zaprojektowana została do użycia z VTK w wersji 6.2, Qt 4 i Pythonem 2.7.10. Program napisany został pod środowisko Windows, natomiast ponieważ nie zawiera bezpośrednio wywołań do funkcji specyficznych dla platformy powinien równie dobrze działać w środowisku Linux.



Uruchomienie aplikacji następuje poprzez wywołanie skryptu `MainWindow.py`. Powoduje to uruchomienie głównego okna aplikacji i załadowanie przykładowego obrazu. Z głównego okna możliwe jest załadowanie serii obrazów poprzez kliknięcie przycisku „Load study” i wskazanie folderu zawierającego pliki DICOM. Okno główne umożliwia również wyświetlenie informacji o programie w wyskakującym oknie poprzez wybranie opcji „Help” > „About”



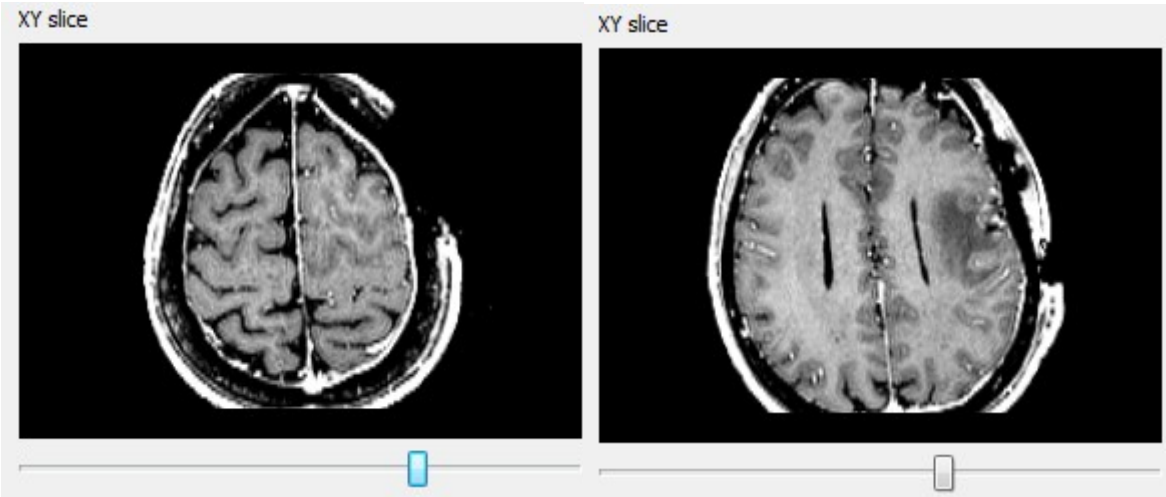
Ponieważ część programu odpowiedzialna za wizualizację stanowi samodzielny moduł, możliwe jest również wywołanie skryptu `DicomVis.py` z konsoli podając jako argument ścieżkę do folderu z serią obrazów. Taka konstrukcja programu umożliwia użycie modułu wizualizacji w innej aplikacji, składającej się z szeregu niezależnych modułów, która jest przedmiotem pracy magisterskiej.

Wyświetlanie obrazów zrealizowane jest za pomocą widżetu Qt o nazwie `QVTKRenderWindowInteractor`, który stanowi Pythonowy wrapper dla klasy `vtkRenderWindowInteractor`. Widżet ten umożliwia wyświetlanie danych VTK w oknie QT oraz obsługę interakcji. Za wczytywanie danych w formacie DICOM odpowiada klasa `vtkDICOMImageReader`. Wyświetlaniem przekojów zajmuje się klasa `vtkImageViewer2`, która łączy w sobie funkcje klas `vtkRenderWindow`, `vtkRenderer`, `vtkImageActor` oraz `vtkImageMapToWindowLevelColors`.

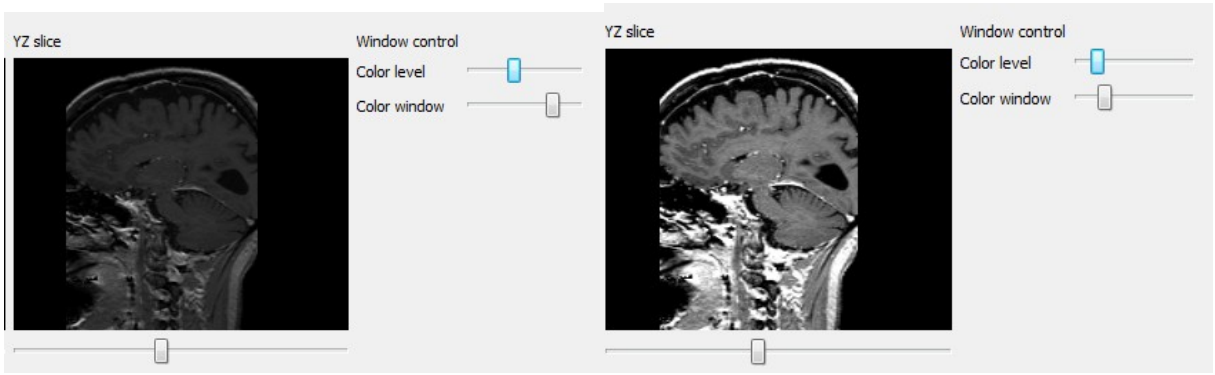
W implementacji wizualizacji 3D skorzystano z przykładu Medical4.py

Suwaki umieszczone poniżej każdego z okien przekrojów umożliwiają zmianę aktualnie wyświetlanego przekroju za pomocą metody `setSlice` klasy `vtkImageViewer2`

```
@QtCore.pyqtSlot(int)
def on_XYSlider_valueChanged(self, value):
    self.viewerXY.SetSlice(value)
```



Suwaki w prawej części okna służą do poprawy percepcji analizowanego obrazu dzięki regulacji parametrów okna.



Kod odpowiedzialny za zmianę spamerów colorlevel i colorwindow w zależności od położenia suwaków.

```
@QtCore.pyqtSlot(int)
def on_WindowCenterSlider_valueChanged(self, value):
    for x in [self.viewerXY, self.viewerXZ, self.viewerYZ]:
        x.SetColorLevel(value)
        x.Render()
@QtCore.pyqtSlot(int)
def on_WindowWidthSlider_valueChanged(self, value):
    for x in [self.viewerXY, self.viewerXZ, self.viewerYZ]:
        x.SetColorWindow(value)
        x.Render()
```

Do testowania aplikacji wykorzystano set Brainix z bazy OsiriX.

Wyniki i wnioski

Wynikiem projektu jest działająca aplikacja realizująca podstawowe założenia projektowe tzn. wyświetlanie serii obrazów DICOM w rekonstrukcjach stanowiących różne przekroje oraz w rekonstrukcji 3D. Praca nad aplikacją pozwoliła zapoznać się z bibliotekami VTK oraz Qt. W realizacji projektu głównymi przeszkodami okazały się marna dokumentacja VTK, w szczególności jeżeli chodzi o Pythonowe bindy oraz skromna ilość przykładów.