

Entwurf und Implementierung eines CT/MRT-Viewers

Das Klinikum Musterberg hat in vergangenen Projekten bereits viel Geld für die Entwicklung von Komponenten ausgegeben, die eigentlich nicht konkret benötigt wurden (CT-Daten-Konvertierer, Logging-Klasse). Um die Entwicklung der Komponenten zu rechtfertigen soll nun ein kompletter CT-Viewer geschrieben werden und die bereits vorhandenen Komponenten verwendet werden. Dadurch – so hofft man – könnten in Zukunft Lizenzkosten für teure CT-Viewer-Software gespart werden. Zudem hofft die Abteilung „Medizinische Informatik“ auf einen späteren Verkauf der Software um die bisherigen Verluste wieder auszugleichen.

Theorie

Magnetresonanztomografie (MRT) ist ein bildgebendes Verfahren zur Darstellung von Strukturen im Inneren des Körpers. Mit einer MRT kann man Schnittbilder des menschlichen (oder tierischen) Körpers erzeugen, die oft eine hervorragende Beurteilung der Organe und vieler Organveränderungen erlauben. Die Magnetresonanztomografie nutzt magnetische Felder.

<http://de.wikipedia.org/wiki/Magnetresonanztomografie>

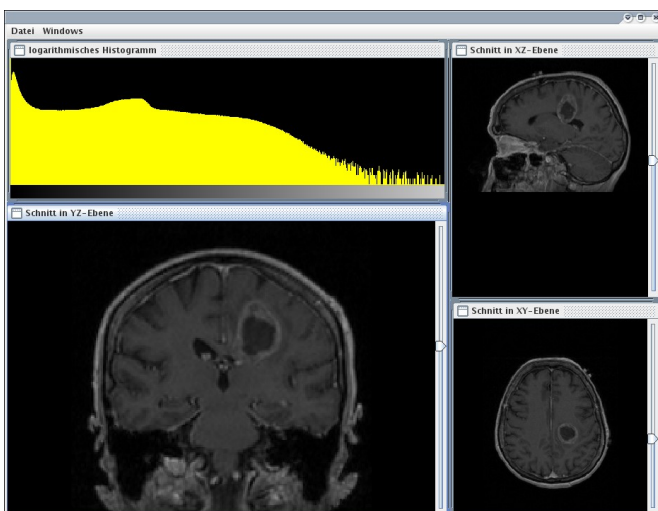
Computer-Tomographie (CT) arbeitet im Gegensatz zur MRT mit Röntgenstrahlen. Dadurch sind weiche Gewebe nur schlecht abgebildet, Knochen sehr gut. Mit Hilfe eines um den Patienten rotierenden Röntgenstrahlers und einem Röntgendetektor werden viele „normale“ Röntgenbilder von unterschiedlichen Seiten gemacht. Daraus lassen sich dann Schnittbilder berechnen.

Weitere Infos zu CT: <http://de.wikipedia.org/wiki/Computertomografie>

Mit den genannten bildgebenden Verfahren können Ärzte eine Vielzahl von Krankheiten diagnostizieren und geeignete Therapiemaßnahmen (z.B. Operation, Bestrahlung) einleiten. Voraussetzung hierfür ist allerdings, dass die vom CT- oder MRT-Gerät aufgezeichneten Daten adäquat am Bildschirm angezeigt werden.

Aufgabenstellung

1. Implementieren Sie einen CT-Viewer mit mindestens der unten genannten Funktionalität.
2. Die unten genannte Vorgehensweise ist sehr zu empfehlen, jedoch nicht bindend. Zur Aufgabenabgabe müssen jedoch alle genannten Anforderungen erfüllt sein, auch wenn nicht mit den unten genannten Schritten vorgegangen wird.



Anforderungen

1. Die Daten der Dateien müssen in allen Hauptebenen (frontal, sagittal, horizontal) als Bilder auf dem Bildschirm darstellbar sein.
2. Es muss ein Menü geben, um die einzelnen Aktionen aufrufen zu können (Beenden, Laden, ...)
3. Die Patientendaten müssen auf dem Bildschirm angezeigt und bearbeitet werden können (nicht im Screenshot)
4. Die Extraktion der Schnittbilder soll mit einem JUnit-Test auf Korrektheit überprüft werden.
5. Die bearbeiteten Daten müssen wieder in die selbe oder eine andere Datei gespeichert werden können.
6. Die Anwendung muss eine gut strukturierte Architektur haben (beispielsweise MVC, andere Architekturen auf Anfrage)
7. In einem Fenster wird eine History mit allen bereits aufgerufenen Befehlen und Aktionen angezeigt. Dabei soll die bereits vorhandene Logging-Klasse (mit eventuellen Änderungen) verwendet werden. Diese soll bei Eintreffen einer neuen Meldung automatisch (Observable) das Anzeigefenster benachrichtigen. An dieser Stelle soll also ein Observer-Pattern implementiert werden.
8. Die Anwendung ist mit Java-eigenen Mitteln zu Internationalisieren. Dies gilt für alle Strings sowie für das Datumfeld, das im jeweils sprachspezifischen Format dargestellt und bearbeitet werden können soll.
9. *Freiwillige Extraaufgaben:*
 1. *Implementieren Sie ein Histogramm (siehe Screenshot)*
 2. *Implementieren Sie eine 3D-Ansicht*
 3. *Implementieren Sie Schnitte in beliebigen Ebenen*
 4. *Implementieren Sie eine Fensterung*
 5. *Implementieren Sie weitere lineare und nicht-lineare Filter auf den Daten*

Hinweise zur Implementierung

1. Das Einlesen der Daten muss nur dann fehlerfrei funktionieren, wenn die einzulesende Datei keine Fehler enthält.
3. Achten Sie auf eine gute Architektur der Anwendung sowie auf die üblichen Programmierkonventionen.
4. Dokumentation ist mit entscheidend für eine erfolgreiche Bearbeitung!

Schritte zur Implementierung

1. Ab 25. April: Entwurf und Implementierung der GUI, noch ohne Funktionalität
 - Es soll bereits ein Fenster angezeigt werden, in dem alle benötigten Elemente der Anwendung (eventuelle Unterfenster, Schieberegler, Eingabefelder, Menüs,...) sichtbar sind.
2. Ab 2. Mai: Entwurf der Architektur der Anwendung, Implementierung der Datenansicht, noch ohne die Schnittbilddarstellung
 - Erstellen des Pflichtenhefts
 - Erstellen des Klassenentwurfs unter Verwendung der bereits vorhandenen Klassen
 - Implementierung der Aktionen unter Verwendung eines Aufzählungsdatentyps vergleichbar mit dem Beispiel in `gpi2.gui.actionCommands`
 - Speichern, Laden, Darstellen und Editieren der Daten (ohne Schnittbilder) soll möglich sein.
3. Ab 9. Mai: Implementierung der Schnittbilddarstellung mit JUnit-Test
 - Implementieren Sie die Anzeige der Schnittbilder in allen geforderten Ebenen
 - Schreiben Sie zur Extraktion der Schnittbilder sinnvolle JUnit-Tests
4. Ab 30. Mai: Internationalisierung der Anwendung

- Alle auf der GUI angezeigten Strings sollen in mindestens zwei Sprachen dargestellt werden können
- 5. Ab 6. Juni: Einbauen des Observer-Patterns und der Ausgabe der Befehls- und Aktions-History
 - Modifizieren Sie die vorhandene Logging-Klasse so, dass sie „observable“ ist.
 - Implementieren Sie die Anzeige der Befehls- und Aktions-History so, dass sie ein Observer ist, sich also automatisch aktualisiert, wenn eine neue Meldung erscheinen soll.
- 6. Ab 13. Juni: Einfügen des erweiterten Datumsfelds in die Oberfläche
 - Im Datumsfeld soll jeweils das Datum im aktuellen sprachspezifischen Format dargestellt und bearbeitet werden können. Es erwartet auch in diesem Format die Eingabe. Beispielsweise für Deutsch im Format TT:MM:JJJJ, in englischsprachiger Einstellung jedoch JJJJ/MM/TT

Abgabetermine

Relevant für Abgabetermin 2 am 26. Mai 2008 sind die Schritte 1 und 2.

Relevant für Abgabetermin 3 am 23. Juni 2008 sind alle restlichen Schritte.