MITK-IGT: Eine Navigationskomponente für das Medical Imaging Interaction Toolkit

Jochen Neuhaus, Ingmar Wegner, Johannes Kast, Matthias Baumhauer, Alexander Seitel, Ingmar Gergel, Marco Nolden, Daniel Maleike, Ivo Wolf, H.-P. Meinzer, Lena Maier-Hein

Abteilung für Medizinische und Biologische Informatik, DKFZ Heidelberg
j.neuhaus@dkfz-heidelberg.de

Kurzfassung. MITK-IGT ist eine Erweiterung des Medical Imaging Interaction Toolkits, die es ermöglicht Softwareprogramme im Bereich bildgestützte Therapie zu erstellen. Dieser Beitrag stellt die Architektur und Designprinzipien von MITK-IGT vor und vergleicht sie mit anderen Open Source Lösungen. Neben der Ansteuerung von Trackingsystemen und Visualisierungsmodulen liegt der Fokus von MITK-IGT auf einer Filterarchitektur, die das schrittweise Verarbeiten von Trackingdaten erlaubt. Zur BVM 2009 wird die erste Version von MITK-IGT als Open Source veröffentlicht.

1 Einleitung

Das Medical Imaging Interaction Toolkit (MITK) ist ein Open Source C++ Toolkit für medizinische Bildverarbeitung [1]. Es basiert auf dem Insight Segmentation and Registration Toolkit (ITK) und dem Visualization Toolkit (VTK) und erweitert diese um anwendungsspezifische Aspekte. Viele verschiedene Anwendungen im Bereich computerunterstützte Diagnose und Therapieplanung basieren auf MITK. Dieser Beitrag stellt eine Erweiterung für MITK vor, die es ermöglicht, mit Hilfe von MITK auch Anwendungen im Bereich bildgestützte Therapie (Image Guided Therapy, IGT) zu erstellen. Obwohl medizinische Navigationssysteme für rigide Strukturen seit mehreren Jahren kommerziell erhältlich sind, ist IGT immer noch ein aktives Forschungsgebiet, vor allem im Bereich von Weichgewebeeingriffen [2].

Es existieren mehrere Open Source Softwarebibliotheken, die das Erstellen von IGT-Anwendungen erlauben. Gemeinsam ist allen, dass sie verschiedene im medizinischen Umfeld gebräuchliche Trackingsysteme ansteuern, eine Registrierung der Trackingdaten mit Bilddaten erlauben und Module für eine Visualisierung der registrierten Daten beinhalten. Das Image Guided Surgery Toolkit (IGSTK) [3] legt den Fokus auf Softwarequalität, vor allem Robustheit. Das CISST package (Computer Integrated Surgical Systems and Technology) [4] besteht aus mehreren Low-Level Bibliotheken zur Hardwareabstraktion von medizinischen Sensoren und Robotern. Darauf baut die Surgical Assistant Workstation (SAW) [5] auf. Ihr Fokus liegt auf roboterassistierter Chirurgie und 3D

Benutzerinteraktion mit speziellen Eingabegeräten. Das Image Guided Therapy Toolkit (IGT:Toolkit, www.na-mic.org/Wiki/index.php/IGT:ToolKit) kombiniert die weit verbreitete Software 3D Slicer (www.slicer.org) über das Netzwerkprotokoll OpenIGTLink (www.na-mic.org/Wiki/index.php/OpenIGTLink) mit IGSTK.

Außer dem IGT:Toolkit, das 3D Slicer mit einer Trackingsystemansteuerung verbinden, beschränken sich alle anderen Bibliotheken auf den Image Guided Therapy Bereich. Komplett integrierte Anwendungen, die Bildanalyse, Therapieplanung und Navigation unterstützen, lassen sich damit nicht oder nur umständlich erstellen. Im Bereich intrainterventioneller Bildgebung ist die Kombination aus Bildanalyse und Navigation jedoch essentiell. Deswegen haben wir den Ansatz gewählt, das weit entwickelte MITK Framework um die MITK-IGT Komponente zu erweitern, so dass damit erstellte Navigationsanwendungen von allen Vorteilen und Features von MITK profitieren können.

2 Material und Methoden

Bei dem Entwurf von MITK-IGT konnten wir auf Erfahrungen aus der Entwicklung mehrerer Navigationssysteme zurückgreifen. Eine Analyse existierender IGT-Projekte zeigte viele Bereiche, die als wiederverwendbare Komponenten modelliert werden können. Ein Ziel beim Entwurf war, einen Schritt weiter zu gehen als andere Toolkits. MITK-IGT soll nicht nur grundlegende Komponenten zur Hardwareabstraktion und Visualisierung bereitstellen sondern zusätzlich eine Architektur definieren, mit der sich auch komplexe Navigationsanwendungen durch Kombination einfacherer Verarbeitungsschritte modellieren lassen.

Die MITK-IGT Komponente ist in zwei Bereiche aufgeteilt. Eine Tracking Komponente bietet Klassen zur Interaktion mit Trackingsystemen. Darauf baut eine Navigationskomponente auf, die weitreichende Verarbeitungsmöglichkeiten der Trackingdaten erlaubt.

2.1 Trackingkomponente

Die Trackingkomponente erlaubt die Ansteuerung verbreiteter Trackingsysteme wie Polaris, Aurora (beide NDI Inc., Waterloo, Kanada), MicronTracker (Claron Technology Inc., Toronto, Kanada) aber auch speziellerer Systeme wie beispielsweise des Space Navigators (3DConnexion, Fremont, USA) oder der Wiimote (Nintendo, Kyoto, Japan). Das Auslesen der Daten aus den Trackingsystemen geschieht in einem eigenen Thread, so dass die Trackingdaten ohne weitere Latenz zur Verfügung stehen, wenn sie benötigt werden. Die Trackingkomponente stellt eine einheitliche Schnittstelle für die verschiedenen Trackingsysteme bereit, so dass eine Navigationssoftware mit geringen Anpassungen mit verschiedenen Trackingsystemen verwendet werden kann.

2.2 Navigationskomponente

In allen untersuchten Anwendungsszenarien wurden die Daten der Trackingsysteme mehrstufig verarbeitet. Dies führte zu dem Entschluss, das in der Bildverarbeitung verbreitete Architekturprinzip einer Filterpipeline auf Trackingdaten zu übertragen. Mit diesem Ansatz lassen sich viele Verarbeitungsschritte als wiederverwendbare Filter kapseln. Die MITK-IGT Filterpipeline basiert auf der ITK Data Processing Pipeline [6]. Abbildung 1 zeigt ein Beispiel für eine MITK-IGT Filterpipeline. Jedes Filterobjekt kann mehrere Trackingdaten als Eingabe haben, verarbeitet sie und stellt sie anderen Filterobjekten als seine Ausgabe bereit. Indem man die Ausgabe eines Filters als Eingabe eines anderen Filters verwendet, wird eine Verarbeitungspipeline aufgebaut. Am Anfang einer solchen Pipeline stehen Filter, die Trackingdaten entweder direkt von der Trackingkomponente, aus einer Datei mit aufgezeichneten Trackingdaten oder über eine Netzwerkschnittstelle erhalten. Ein Filterobjekt kann Trackingdaten unterschiedlicher Filterobjekte als Eingabe bekommen und seine Ausgabe auch wieder mehreren Filtern bereitstellen. Die Filterpipeline funktioniert nach dem Pull-Prinzip. Wenn die Ausgabe eines Filters abgefragt wird, veranlasst er zuerst eine Aktualisierung seiner Eingabefilter, bevor er mit deren Ergebnissen seine eigenen Berechnungen durchführt. In der Regel wird eine Filteraktualisierung zeitgesteuert in einer festen Frequenz durchgeführt. Alternativ erzeugen die Quellfilter immer, wenn sie von Trackingsystem, Logdatei oder Netzwerk neue Trackingdaten bekommen, ein Event, das mit der Aktualisierungsmethode der

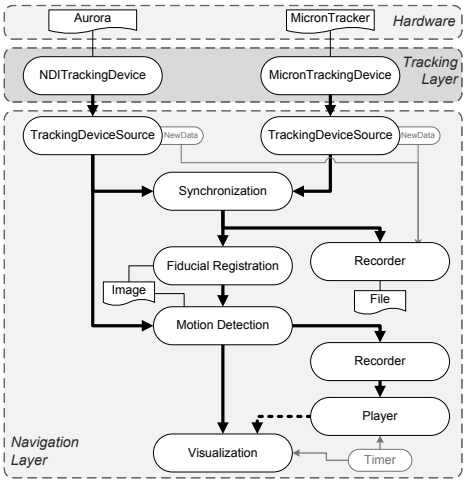

Abb. 1. Beispiel einer MITK-IGT Filterpipeline. Der TrackingDeviceSource - Synchronization - Recorder Teil der Pipeline wird über Events mit der Frequenz der Trackingsysteme betrieben. Der Rest der Pipeline wird von einem Timer mit niedrigerer Frequenz aktualisiert. Die Visualisierung kann zwischen den Trackingsystemen und einem Recorder/Player Paar umgeschaltet werden, um die Visualisierung während einer Intervention „zurück zu spulen“.

Filter gekoppelt werden kann. Dadurch kann mit MITK-IGT neben dem Pull-Mechanismus auch ein eventgesteuerter Push-Mechanismus realisiert werden.

Ein wichtiger Aspekt bei der Pipelineaktualisierung ist, dass nicht alle Teile einer Verarbeitungspipeline in der gleichen Frequenz aktualisiert werden müssen. Da für jeden Filter einer Pipeline getrennt eine Aktualisierung veranlasst werden kann, können einzelne Teile von verschiedenen Threads in verschiedenen Frequenzen aktualisiert werden. Redundante Berechnungen werden mit Hilfe von Zeitstempeln und einer Pufferung der Ausgabedaten vermieden. Mit Hilfe dieser Filterarchitektur können viele Anforderungen erfüllt werden, indem passende Filterobjekte erstellt und miteinander kombiniert werden. Es existieren beispielsweise Filter für folgende Features:

- *Senden und Empfangen von Trackingdaten über ein Netzwerk.* Trackingdaten können über eine Netzwerkschnittstelle gesendet und empfangen werden. Dies erlaubt das Entkoppeln der Trackingsystemansteuerung, der Verarbeitung und der Visualisierung auf mehrere Computer. Beispielsweise können unterschiedliche Visualisierungen einer Intervention parallel auf mehreren Computern erzeugt werden.
- *Speichern und Wiedergabe von Trackingdaten.* Dies erlaubt das Auszeichnen von Interventionen. Bei der Wiedergabe lässt sich die Wiedergabegeschwindigkeit variieren, so dass sowohl Zeitraffer- als auch Zeitlupenfunktionen möglich sind.
- *Synchronisation mehrerer Trackingsysteme.* Ein Synchronisierungsfilter kann Trackingdaten mehrerer Trackingsysteme zeitlich synchronisieren, um unterschiedliche Latenzzeiten auszugleichen. Zusammen mit einem Registrierungsfilter, der die Trackingdaten in ein gemeinsames Koordinatensystem transformiert, können mehrere Trackingsysteme zu einem virtuellen Trackingsystem kombiniert werden.
- *Koordinatensystemtransformation.* Über verschiedene Registrierungsfilter können Trackingdaten in andere Koordinatensysteme transformiert werden.
- *Visualisierungsfilter.* Es existieren Filter für Instrumenten- und Kameravisualisierung, Virtual- und Augmented Reality und spezielle Zielführungsdarstellungen.
- *Particle- und Kalmanfilter.* Diese Filter erlauben die räumliche und zeitliche Filterung von Trackingdaten, um einzelne Messfehler und Aussetzer zu überbrücken.

Die Trackingdaten enthalten neben Position, Orientierung und Zeitstempel auch eine Datenstruktur zur Fehlerpropagierung, so dass sowohl die Trackingsysteme als auch einzelne Filter eine Fehlerabschätzung für die Trackingdaten abgeben können. Da für eine Intervention meist mehrere Objekte getrackt werden müssen, können Toolkonfigurationen zusammen mit allen weiteren benötigten Daten wie Oberflächenmodellen, Geometriebeschreibungen, Tooltypen, Toolaufgaben und anderen Metadaten in einer gemeinsamen Beschreibungsdatei gespeichert und geladen werden.

Für jeden Filter existieren GUI Widgets, die dazu verwendet werden können, die Parameter des Filters einzustellen. Ein Filter kann mehrere GUI Widgets besitzen, etwa ein User-, ein Experten- und ein Debug-Widget.

3 Ergebnisse

Eine interne Version von MITK-IGT wird bereits für mehrere Navigationssysteme verwendet [7]. Eine Open Source Version ist momentan in Entwicklung. Zur BVM 2009 wird eine erste Version auf www.mitk.org unter einer BSD-kompatiblen Lizenz veröffentlicht.

4 Diskussion

Softwaresysteme, die während einer Intervention eingesetzt werden, müssen höchste Qualitätsanforderungen erfüllen. Zuverlässigkeit ist dabei der wichtigste Qualitätsfaktor. IGSTK versucht, Zuverlässigkeit einerseits durch explizite Modellierung und Kontrolle der Objektzustände mit Zustandsmaschinen, andererseits durch Minimierung der Schnittstellen und Features zu erreichen. Die dahinter stehende Idee ist, dass einfache Software weniger Fehler enthält als komplexe Software. Im Forschungsbereich halten wir diesen Ansatz für zu einschränkend. MITK-IGT soll es gerade ermöglichenauch komplexe Navigationssoftware zu entwickeln. Es soll eine Umgebung bieten, die es erlaubt sich auf neue Aspekte zu konzentrieren, in dem es eine erweiterbare Sammlung von wiederverwendbaren Grundfunktionalitäten und ein flexibles Framework, um diese zu einem Gesamtsystem zu kombinieren, bereit stellt.

Literaturverzeichnis

1. Wolf I, Nolden M, Böttger T, et al. The MITK approach. Proc MICCAI Workshop: Open-Source Software. 2005.
2. Baumhauer M, Feuerstein M, Meinzer HP, et al. Navigation in endoscopic soft tissue surgery: Perspectives and limitations. J Endourol. 2008;22(4):751–766.
3. Gary K, Ibáñez L, Aylward S, et al. IGSTK: An open source software toolkit for image-guided surgery. IEEE Computer. 2006;39(4):46–53.
4. Deguet A, Kumar R, Taylor R, et al. The cisst libraries for computer assisted intervention systems. Proc MICCAI Workshop: Systems and Architectures for Computer Assisted Interventions. 2008;Http://hdl.handle.net/10380/1465.
5. Vagvolgyi B, DiMaio SP, Deguet A, et al. The surgical assistant workstation. Proc MICCAI Workshop: Systems and Architectures for Computer Assisted Interventions. 2008;Http://hdl.handle.net/10380/1466.
6. Ibáñez L, Schroeder W, Ng L, et al. The ITK Software Guide. Kitware Inc.; 2003.
7. Meinzer HP, Maier-Hein L, Wegner I, et al. Computer-assisted soft tissue interventions. Proc IEEE ISBI. 2008.