Visualizing the Volumetric Accuracy of Medical Tracking Solutions

December 17, 2015
Intermediate Presentation

Stefan Matl
matl@in.tum.de

Supervisor: Benjamin Busam
Project Partner: FRAMOS
Outline

- Project Outline
  - Motivation & Problem Statement
  - Requirements & Specifications
  - Scoping & Planning
- Software Design
  - Software Architecture & Design Patterns
- Progress
- Project Outlook
  - Tasks and Subtasks & Challenges
Project Outline

- Motivation
  - Position and orientation information in real-time
  - Different systems
  - Spatial accuracy often unknown

- Problem Statement
  - Visualizing the volumetric accuracy
  - Interactive GUI

* Figures of US probe and tracking setup kindly provided by FRAMOS GmbH
Project Outline

- Requirements & Specifications
  - Visualization
    - Differentiate tracking error and system accuracy
    - Additional reference points or medical devices
    - Different visualizations
  - GUI
    - Interactive
    - Real-time
  - Input
    - Pose & accuracy via TCP / IP
  - Output
    - Rendering
  - Environment
    - Windows, Visual Studio, C++, OpenGL, OpenIGTLink, GitLab
Project Outline

- Scoping and Planning - Original Project Plan

Initial Meeting

Project Requirements Presentation

Data Collection
- Camera Calibration
- Image Acquisition
- Accuracy Evaluation

Data Visualization
- 3D Plot / Map
- Alternatives
- GUI

Intermediate Presentation

Integration
- Analyse FRAMOS OTS
- Verification

Final
Software Design  UML Use Case Diagram

Visualization / GUI

- Rotate OpenGL Camera << includes >>
- Start / Stop Fade-Out << includes >>
- Change Fade-Out Time << extends >> << includes >>
- Change Measurement Size << includes >>
- Change Visualization Method << includes >>
- Load / Save Data << includes >>
- Reset Scene << includes >>

FRAMOS Employee
Software Design  UML Sequence Diagram

Camera  Server  Main  Scene  DataTree

Loop

getNewData

sendKeysData

<<return>>

sendKeysData

sendKeysData

Alt

[there are new measurements]

addNewMeasurements

Loop

[for each measurements]

addVertex

<<return>>

<<return>>

<<return>>

render

<<return>>
Software Design

• Strategies
  • GUI / rendering: bottom-up implementation
  • Server-Client: top-down

• Server-Client Communication
  • OpenIGTLink protocol, using specification for Position Message
  • Separate thread

• Design Patterns
  • Creational Patterns: RAIIL
  • Concurrency Patterns: Monitor Object (for server)
Progress

- Initial Approach
  - Show all positions
  - Use color for accuracy

- Drawback
  - Cluttered
  - Overloaded
  - Some structure

- Questions
  - How useful is old data?
  - What if paths cross?
Progress

- Questions
  - How useful is old data?
    - Postprocessing
  - What if paths cross?
    - Show latest data
- Current Approach
  - Show recent positions
  - Use color for accuracy
  - Use size for accuracy
  - Show reference frame
  - Show additional devices
  - Let measurements fade out
Progress
Project Outlook

- Completed Tasks and Subtasks
  - 5 different visualizations
  - Showing reference frame
  - Showing US probe mesh
  - Server implementation
  - OpenIGTLink integration
  - Customization of visualization
  - Connection with FRAMOS framework
    (directional communication via OpenIGTLink packages)

- Challenges so far
  - Integration of OpenIGTLink took > 7 h
Project Outlook

- Upcoming Tasks
  - Finish integration on client side
    (framework @ FRAMOS)
  - Camera calibration
    risk: might be challenging
  - Finish GUI
  - More tests / evaluation
  - Heatmap for postprocessing (optional)
  - Documentation / report
Project Outlook

- Updated Project Plan

|--------|-------------|------------|------------|------------|------------|------------|------------|

⚠️ Intermediate Presentation

Finish Integration → Camera Calibration → Finish GUI → More Tests → Heatmap → Documentation / Report → Final Presentation
• Questions

• Comments

• Ideas