COMP390
2023/24

Computer Vision and AR for Endovascular Intervention

Student Name: [Yulin Huang]

Student ID: [201676465]

Supervisor Name: [Anh Nguyen]

DEPARTMENT OF COMPUTER SCIENCE

University of Liverpool
Liverpool L69 3BX

Acknowledgements

UNIVERSITY OF LIVERPOOL

COMP390 2023/24

Computer Vision and AR for Endovascular Intervention

DEPARTMENT OF COMPUTER SCIENCE

University of Liverpool
Liverpool L69 3BX

Contents

Abstract

This section should contain a concise summary of the document content.

Statement of Ethical Compliance

Data Category: A
Participant Category: 0

I confirm that I have read the ethical guidelines and will follow them during this project. Further details can be found in the relevant sections of this proposal.

1 Introduction & Background

2 Design & Implementation

2.1 Part1: Real-world Model Interaction and Tracking

In this part, I used OpenCV[1] and SciKit-Surgery Augmented Reality[2] libraries for image processing and model tracking, while Qt[3] is used to design the graphical interface, and also VTK[4]. OpenCV and SciKit-Surgery help in processing images and tracking Aruco Marker[5] within these images. Qt allows me to create a user-friendly interface that ensures multiple settings can be made more easily, enhancing the user experience. VTK facilitates the management of overlay and multi-layer video rendering in my project. An ArUco Marker Generator is also included, which enables the user to generate different ArUco Markers and save them. This section will detail the system's design, focusing on System Components and Organization, Data Structures and Algorithms, User Interface Design, and Design Notation and Diagrams.

2.1.1 Design

1. **System Components and Organization**
   The project is structured into three primary components, each responsible for distinct functionalities within the system. Here's a detailed breakdown of these components and their organization:

   (a) **Frontend - User Interface**
   The system's user interface is developed using Qt, a framework that enables the creation of graphically applications. The main class controlling the UI is *Overlay_and_Tracking.py*, which serves as the central hub for user interactions and display functionalities. This class manages the overlay of models on video streams and provides interactive elements for users to control various settings, such as model color changes, video source switching, models uploading and changing, or adjusting ArUco marker types and sizes, and more.

   (b) **Backend - Helper Classes**
   The backend is composed of various helper classes, each set to handle specific tasks:

   - *Image Capture:* The *video_source.py* class handles the acquisition of video streams from various sources, including live cameras and recorded media. It is responsible for configuring camera settings, initializing video capture, and video frame cropping to adapted to screen size. This component ensures the reliability and stability of video feed intake.

- *Model Loading:* Managed by *model_loader.py*, this component is important for the model loading and initializing. It loads ".stl" model files from external files, sets up texture mapping, and prepares the models for real-time overlay. The class also checks for errors in model data to prevent crashes or rendering issues during operation. It also optimizes the structures of model data to enhance rendering efficiency and reduce memory overhead.
- *Model Overlay:* The *overlay_window.py* is central to integrating 3D models with live or recorded video. With the help of VTK, this module could set up a multi-layered rendering environment where each layer can independently handle elements like video backgrounds, 3D models, or some GUI overlays(In the future, maybe.).
- *Transform Management:* The *transform_manager.py* class provides a method for managing 4x4 transformation matrices crucial for spatial adjustments of models in 3D space. It stores and retrieves transformations efficiently. And allow it for dynamic modifications of object orientations and positions.
- *ArUco Marker Tracking:* Functionality provided by *arucotracker.py* includes detecting and decoding ArUco markers from the video stream using OpenCV. This module calculates position and orientation of detected marker, and handle the spatial position data for model tracking.

(c) **Additional Component - ArUco Marker Generator**
An independent component in the system is the ArUco Marker Generator, managed by *Aruco_Generator.py.* This tool allows users to select and visualize different ArUco markers. Users can also save these markers as separate image files.

Organization:
The system's architecture is designed to easy maintenance and scalability. The modular nature of the helper classes allows for isolated development and testing, which enhances the system's robustness and flexibility. This organization simplifies development and testing and enables the integration of additional functionalities in the future with minimal disruption to the existing system.

2. **Data Structures and Algorithms**

   (a) **Frame Storage**
   (b) **Feature Data Structures**
   (c) **Image Processing Algorithms**
   (d) **Marker Detection and Tracking**
   (e) **Model Positioning and Rendering**
   (f) **Optimization Techniques**
   (g) **Algorithm Efficiency and Complexity**

3. **User Interface Design**

   (a) **Screen Mockups, Sketches, and Screenshots**
   (b) **OpenCV-Based Real-world Interaction Component**
   (c) **ArUco Marker Detection and Model Positioning**

4. **Design Notation and Diagrams**

   (a) **Use Case Diagrams**
   (b) **Interaction Diagrams and Class Diagrams**
   (c) **Pseudocode and Data Flow Diagrams**

2.1.2 Implementation

2.2 Part2: Endovascular Intervention Simulation

2.2.1 Design

2.2.2 Implementation

3 Testing & Evaluation

4 Project Ethics

5 Conclusion & Future Work

5.1 Conclusion

5.2 Future Work

6 BCS Criteria & Self-Reflection

6.1 An Ability to Apply Practical and Analytical Skills

6.2 Innovation and/or Creativity

6.3 Synthesis of Information, Ideas, and Practices

6.4 Meeting a Real Need in a Wider Context

6.5 An Ability to Self-Manage a Significant Piece of Work

6.6 Critical Self-Evaluation of the Process

References

[1] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[2] S. Thompson, T. Dowrick, M. Ahmad, *et al.*, "SciKit-Surgery: Compact Libraries for Surgical Navigation," *International journal of computer assisted radiology and surgery*, vol. 15, no. 7, pp. 1075–1084, 2020. DOI: 10.1007/s11548-020-02180-5.

[3] The Qt Company, *Qt - cross-platform software development for embedded and desktop*, [Online; accessed 29-April-2024], 2024. [Online]. Available: https://www.qt.io.

[4] W. Schroeder, K. Martin, and B. Lorensen, *The Visualization Toolkit (4th ed.)* Kitware, 2006, ISBN: 978-1-930934-19-1.

[5] M. Fiala, "Artag, a fiducial marker system using digital techniques," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2, 2005, 590–596 vol. 2. DOI: 10.1109/CVPR.2005.74.

[6] "Aruco markers." (2021), [Online]. Available: https://fab.cba.mit.edu/classes/865.21/people/zach/arucomarkers.html (visited on 04/29/2024).

[7] Wikipedia contributors, *Qt (software) — Wikipedia, the free encyclopedia*, [Online; accessed 29-April-2024], 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Qt_(software)&oldid=1219937369.

[8] Wikipedia contributors, *Opencv — Wikipedia, the free encyclopedia*, [Online; accessed 29-April-2024], 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=OpenCV&oldid=1208982530.

Glossary

Aruco Marker ArUco markers are 2D binary-encoded fiducial patterns designed to be quickly located by computer vision systems. ArUco marker patterns are defined by a binary dictionary in OpenCV, and the various library functions return pattern IDs and pose information from scanned images.[6] 3

OpenCV OpenCV (Open Source Computer Vision Library) is a library of programming functions mainly for real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage, then Itseez (which was later acquired by Intel). The library is cross-platform and licensed as free and open-source software under Apache License 2. Starting in 2011, OpenCV features GPU acceleration for real-time operations.[7] 3

Qt Qt (pronounced "cute" or as an initialism) is cross-platform application development framework for creating graphical user interfaces as well as cross-platform applications that run on various software and hardware platforms such as Linux, Windows, macOS, Android or embedded systems with little or no change in the underlying codebase while still being a native application with native capabilities and speed.[8] 3

VTK The Visualization Toolkit (VTK) is an open-source, freely available software system for 3D computer graphics, modeling, image processing, volume rendering, scientific visualization, and 2D plotting. It supports a wide variety of visualization algorithms and advanced modeling techniques, and it takes advantage of both threaded and distributed memory parallel processing for speed and scalability, respectively.[4] 3