

PROBLEM

Polarized light is a powerful tool that has been utilized in fields like geology, chemistry, and in our team’s case, biology. Polarized light can be used to view the structure of the extracellular matrix and is especially responsive to collagen, the protein mostly responsible for holding the body together. This application of polarized light can help view information sensitive to this reaction and can open the door to new discoveries.

CURRENT SYSTEM

The model built around the utilization of Lucid Vision Labs’ polarization camera and LEDs of Arduino circuits requires a specialized GUI in order to have control of both technologies at once. The current system, built upon a MATLAB GUI, runs into issues like crashing spontaneously and intermittent packet loss. This project aims to resolve these issues and offer a cleaner user experience.

REQUIREMENTS

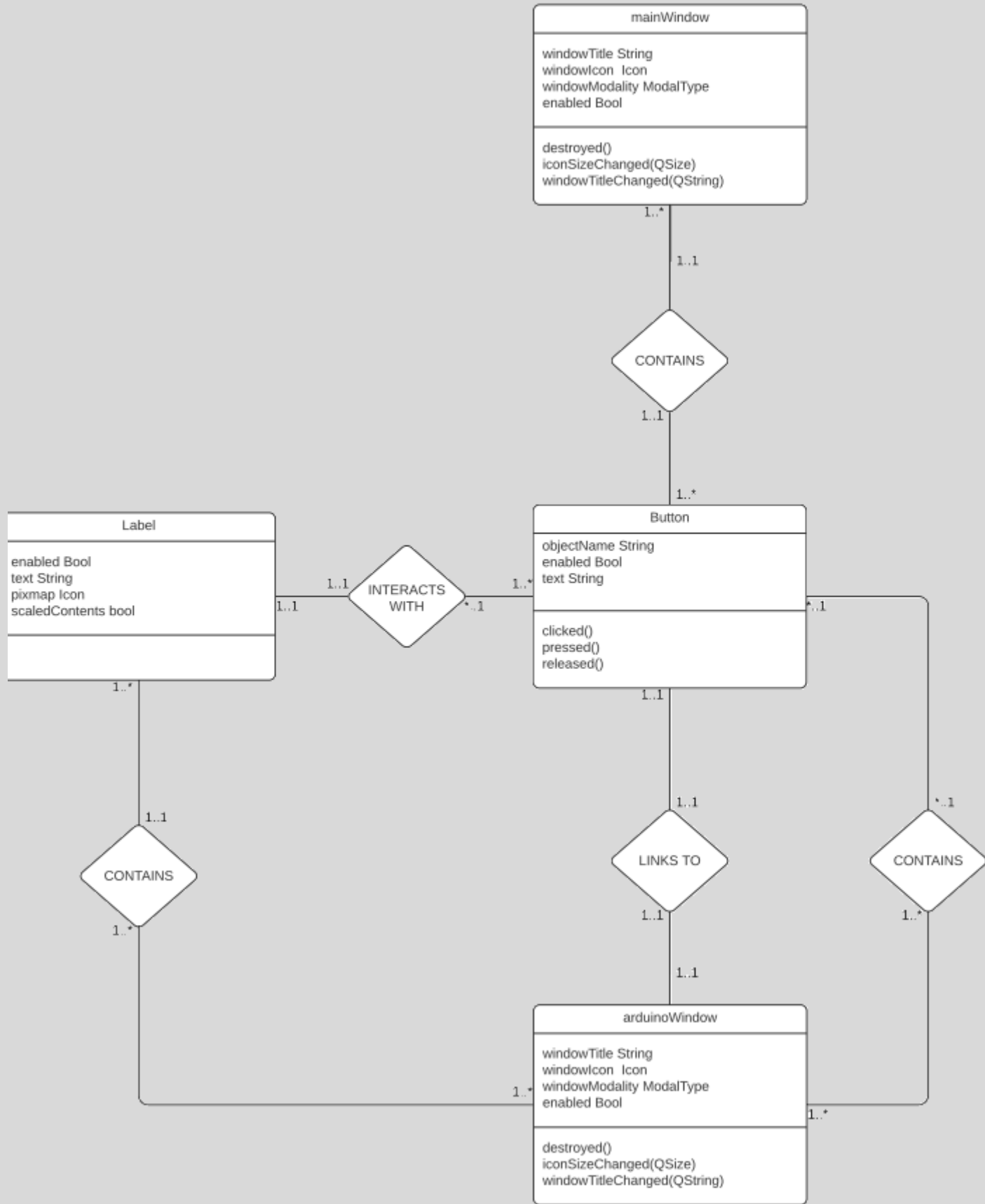
- Users will be able to launch ArenaView through the GUI, allowing for access to camera controls
- Users can access Arduino controls (LED controls) through a special page in the GUI
- Users can preview images that were saved from the camera
- Users can edit camera properties like camera frame rate/exposure time

VERIFICATION

The software underwent inspection by QA testers after new features were implemented, being careful to reach acceptance criteria for every button available to them. Tests fell under these categories:

- **Readability** – Users foresee what a button should do
- **Usability** – Users can properly understand how to use the button with instant feedback
- **Functionality** – Feature works as intended, noting any oddities and bugs that occur

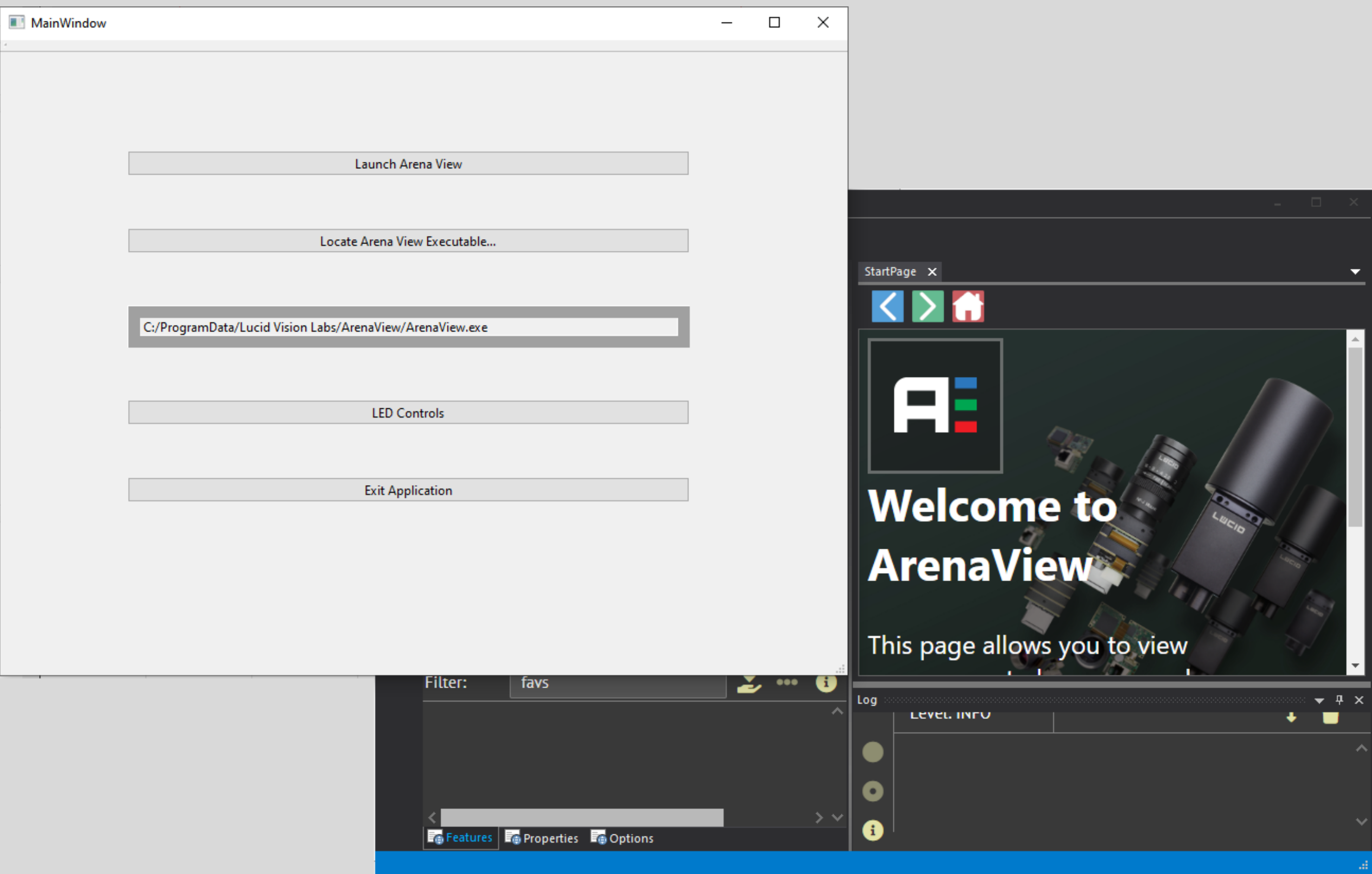
SYSTEM DESIGN



IMPLEMENTATION

- The GUI is written in C++ through the QT framework
- The library for Lucid Labs camera access was provided by Arena SDK, a product of Lucid Vision Labs
- The GUI for camera controls is accessed through Arena View, a program by Lucid Vision Labs
- Arduino controls were provided by Arduino IDE

SCREENSHOTS



SUMMARY

Our program, ArenaAssistant, is a GUI model that has lots of room for expansion and more features. These are supported by its framework, QT, and its relatively simple interface that can allow for features to be added without hindering any previous progress. It is also more lightweight and manageable than the previous implementation through MATLAB, as it can be launched in its own executable.

ACKNOWLEDGMENT

The material presented in this poster is based upon the work done by John Robinson. Special thanks to our product owner and mentor, Jessica Ramella-Roman, and our instructor Masoud Sadjadi.