# SpaceRock GUI Documentation (DRAFT)

## Introduction

The SpaceRock system requires a Graphical User Interface (GUI) in order to allow ground-based users to control and evaluate the system’s performance. This document will describe the layout, design, and internal workings of the SpaceRock GUI simulator.

## Design



Figure 1: Mock-Up of SpaceRock GUI

|  |  |
| --- | --- |
| **UI Element** | **Function** |
| Object Display Canvas | Either displays the greyscale image in Raw mode or displays the detected objects in Processed mode. The greyscale image and detected objects will be scaled according to the current camera settings. The detected objects will be color coded according to their possible danger to the SRS. In Processed mode, objects can be selected and their details can be viewed from the Selected Object Information Panel. |
| Selected Object Information Panel | Displays the current selected object’s camera assigned ID, estimated size, and estimated velocity while in Processed Mode. |
| Camera Zoom | Adjusts the camera’s zoom toward the center of its field of view. |
| Section Size | Adjusts the image processing section size. |
| Section Overlap | Adjusts the amount of overlap each section is allowed to have with its neighbors. |
| Raw/Processed Mode | Raw Mode returns the greyscale image with no processing. Processed Mode returns all objects detected processed and its relative coordinates on the field of view. |

## Technical Information

### Networking

The SpaceRock GUI communicates with a SpaceRock system (real or simulated) via a Transmission Control Protocol (TCP) stream containing serialized Java objects. TCP was chosen for ease-of-development, and experimentation may reveal that the extra overhead it introduces is unacceptable. In that case, switching to User Datagram Protocol (UDP) should not be difficult.

The actual contents of the TCP stream are special purpose Java classes developed in parallel with the GUI. The client sends an “Outgoing” class object, which contains camera parameters such as desired zoom level, and also system parameters such as desired sector overlap. The SpaceRock system or simulator sends back an “Incoming” class object, which contains a list of identified objects (and their position, size, heading, etc.), and optionally a list of images used for the raw image display mode.

### GUI

The GUI is implemented using JavaFX. JavaFX was chosen as it is the modern Java UI framework, and presents very large amounts of flexibility in GUI layout and behavior. Additionally, JavaFX is hardware accelerated, ensuring that even large displays of space objects are not slowed down by the rendering system.