|  |
| --- |
|  |
| Project 1: Write-up |
| ThunderCat |
| Daniel Tompkins & Gregory Kniaziuk |

|  |
| --- |
|  |

Table of Contents

Description 2

GUI Storyboards 2

User Narratives

Formal Use Cases

UML Modeling Diagrams

Class

Use Cass

Design patterns used

Pros/Cons

Design considerations

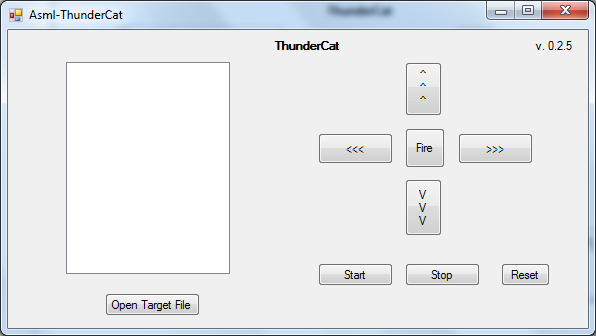
Issues

Description

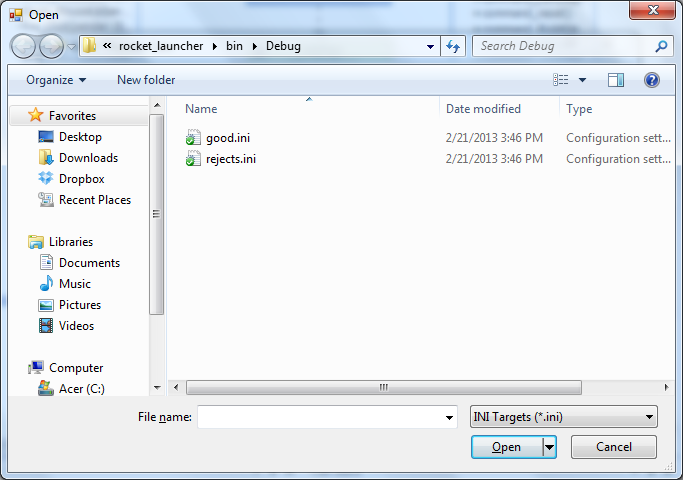
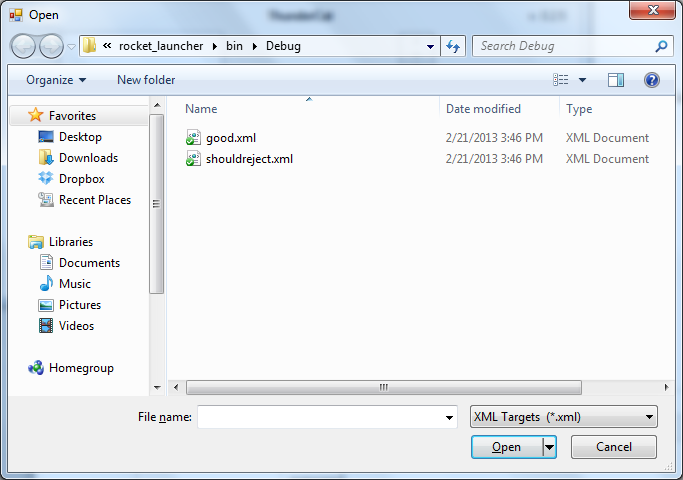
This document contains the details surrounding our Graphical User Interface (GUI) for controlling the turret that we will use in the final project for CPTS 323. The information here was used in the creation of the GUI for this project, and will continue to be used as more parts are added. The GUI storyboards will cover the possible states of the interface to make sure button function makes sense and doesn’t allow the user to end up in an invalid state. The User Narratives and Use Cases will be used for testing and design later on. These will cover the normal and potential use of the program so that we can be sure to include the correct functionality and test for potential problems. UML diagrams and design patterns will describe the objects within the program and how they are working together. The design patters will focus on the general uses that we apply, and the diagrams will show how they were implemented in the code. Design Considerations will be an elaboration on the previous section that discusses other considerations we will be making as we continue on the project. The final section, issues, covers out issue tracking during the making of this project.

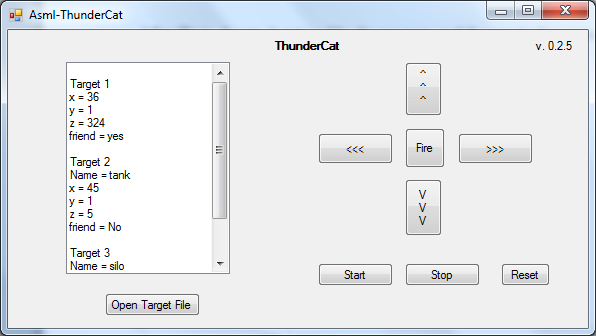
GUI Storyboards

Group name is displayed in the top-center and the version is displayed in the top-right corner of the GUI. On the left side of the GUI manual controls of the turret, Start button, Stop button, and Reset button are located. Directional controls are left, right, up, and down, when one of the buttons is held, the turret moves in that direction. Button in the center of directional controls is the fire button, when pressed the turret should fire a missile in the current direction. Start button starts the Search and Destroy mode. Stop button should stop the Search and Destroy mode. Reset button calibrates the position of the turret by setting the gun at 0, 0, 0.



On the right side of the GUI the Target list box and Open Target File button are located. When Open Target File button is pressed, file explorer window opens and lets you choose a target file of one of the formats, xml or ini. Examples for both formats are below:

After file is chosen, it gets red by the program and the acquired targets are displayed in the Target list box.



User Narratives

Formal Use Cases

|  |  |
| --- | --- |
| **ID:** | **UC-1** |
| **Title:** | Fire Missiles from idle |
| **Description:** | The AMSL should fire missiles when the fire button is pressed. |
| **Primary Actor:** | Judge |
| **Preconditions:** | Software has started in idle mode |
| **Postconditions:** | Missile is fired |
| **Main**  **Success Scenario:** | 1. User clicks fire button  2. Turret fires missile in current direction  3. Program makes a firing noise |
| **Extensions:** | 1a. Nothing happens  1b. Turret fires more than one missile |
| **Frequency of Use:** | For testing |
| **Owner:** | Daniel Tompkins |
| **Priority:** | 5 – High |
| **Risk** | Retain\_06 Retain\_13 |

|  |  |
| --- | --- |
| **ID:** | **UC-2** |
| **Title:** | Fire Missiles from Search and Destroy |
| **Description:** | The AMSL should fire missiles when a target is acquired. |
| **Primary Actor:** | Judge |
| **Preconditions:** | Software has started in Search and Destroy mode |
| **Postconditions:** | Missile is fired |
| **Main**  **Success Scenario:** | 1. Program identifies a target  2. Turret rotates to point directly as target  3. Turret fires a missile |
| **Extensions:** | 1a. Mis-identifies Friend as Enemy  1b. Wrong coordinates given  2a. Turret position not calibrated correctly  3a. Missile misses target  3b. Missile hits a friendly target |
| **Frequency of Use:** | For testing and final competition |
| **Owner:** | Daniel Tompkins |
| **Priority:** | 5 – High |
| **Risk** | Retain\_06 Retain\_11 |

|  |  |
| --- | --- |
| **ID:** | **UC-3** |
| **Title:** | Search and Destroy |
| **Description:** | The program should search for targets for up to two minutes or until four missiles are fired. |
| **Primary Actor:** | Judge |
| **Preconditions:** | Software has started in idle mode |
| **Postconditions:** | All missiles fired, timer stopped, software in idle |
| **Main**  **Success Scenario:** | 1. User clicks on Search and Destroy 2. Timer starts   3. After all four missiles are fired, or two minutes, the program goes back to idle and timer stops |
| **Extensions:** | 1a. Program fails to start Search and Destroy  2a. Timer doesn’t start  3a. Timer fails to stop  3b. Not all missiles were fired  3c. Timer does not reset |
| **Frequency of Use:** | For testing and final competition |
| **Owner:** | Daniel Tompkins |
| **Priority:** | 2 – Low |
| **Risk** | Retain\_14 Retain\_07 |

|  |  |
| --- | --- |
| **ID:** | **UC-4** |
| **Title:** | Identifying Targets from files |
| **Description:** | The program should import target information from an .ini or .xml file |
| **Primary Actor:** | Judge |
| **Preconditions:** | Software has started in idle mode |
| **Postconditions:** | Target information stored and ready for Search and Destroy |
| **Main**  **Success Scenario:** | 1. User selects a file to import from 2. Program extracts target information |
| **Extensions:** | 1a. File does not exist  1b. File does not have .ini or .xml extension  2a. File is not a valid .ini or .xml  2b. Files does not contain any targets |
| **Frequency of Use:** | For testing and final competition |
| **Owner:** | Daniel Tompkins |
| **Priority:** | 1 – Low |
| **Risk** | Avoid\_02 |

|  |  |
| --- | --- |
| **ID:** | **UC-5** |
| **Title:** | Identifying Targets from visual data |
| **Description:** | The program should identify targets using image data during Search and Destroy |
| **Primary Actor:** | Judge |
| **Preconditions:** | Software is in Search and Destroy |
| **Postconditions:** | Turret is aimed at the target |
| **Main**  **Success Scenario:** | 1. Program identifies a target from visual data 2. Target is outlined on live video feed 3. Turret rotates to given coordinates |
| **Extensions:** | 1a. Fails to identify targets  2a. No target outline appears  3a. Turret rotates to wrong angle |
| **Frequency of Use:** | For testing and final competition |
| **Owner:** | Daniel Tompkins |
| **Priority:** | 2 – Low |
| **Risk** | Retain\_15 Retain\_08 Retain\_10 |

|  |  |
| --- | --- |
| **ID:** | **UC-6** |
| **Title:** | Starting the program |
| **Description:** | When the program is run, the GUI should launch, allowing the user to control the turret |
| **Primary Actor:** | Judge |
| **Preconditions:** | The program is built on the computer |
| **Postconditions:** | GUI is loaded and user can control turret |
| **Main**  **Success Scenario:** | 1. User runs the executable 2. GUI loads |
| **Extensions:** | 2a. GUI fails to load |
| **Frequency of Use:** | For testing and final competition |
| **Owner:** | Daniel Tompkins |
| **Priority:** | 2 – Low |
| **Risk** | Retain\_05 |

|  |  |
| --- | --- |
| **ID:** | **UC-7** |
| **Title:** | Manual turret control |
| **Description:** | The turret should be able to be rotated horizontally and vertically using manual controls |
| **Primary Actor:** | Judge |
| **Preconditions:** | The program is in Idle |
| **Postconditions:** | Turret is rotated from initial position |
| **Main**  **Success Scenario:** | 1. User clicks left, right, up, or down and turret rotates 2. User holds left, right, up, or down and turret rotates at increasing speed |
| **Extensions:** | 1a. Turret doesn’t rotate  1b. Turret doesn’t stop rotating  2a. Turret rotation speed does not increase  2b. Turret doesn’t continue rotating while button held |
| **Frequency of Use:** | For testing and final competition |
| **Owner:** | Daniel Tompkins |
| **Priority:** | 2 – Low |
| **Risk** | Retain\_16 Retain\_09 |

UML Modeling Diagrams



Design Patterns Used

Design Considerations

Issues