Smart Anomaly Detection Assistant (SADA) v0.9 Development Report

1. Introduction

This document provides a detailed report on the development of version 0.9 of the Smart Anomaly Detection Assistant (SADA).
The development process focused on improving the selection, zoom, brightness, and contrast adjustment functionalities within the GUI application.
The project aimed to enhance the user experience by ensuring accurate image adjustments and seamless zoom operations.

2. Development Timeline

The development of SADA v0.9 spanned over multiple sessions, addressing various issues and incorporating new features.
Below is a summary of the key development phases:

1. **Initial Setup and Image Selection Implementation**: Set up the basic framework and implemented image selection features, including polygon and rectangular selections.

2. **Zoom Functionality**: Developed zoom-in, zoom-out, and zoom-to-selection functionalities, ensuring that selected areas could be magnified correctly.

3. **Brightness and Contrast Adjustments**: Integrated sliders to adjust brightness and contrast. Initial implementations had issues with adjusting only the selected areas.

4. **Debugging and Issue Resolution**: Significant time was spent debugging issues where brightness and contrast adjustments were applied incorrectly, particularly after zoom operations.

5. **Final Integration and Testing**: Ensured all components worked together seamlessly, with special attention to edge cases and user interactions.

Overall, the development of version 0.9 took approximately 25-30 hours, including coding, debugging, and testing.

3. Key Challenges and Solutions

Several critical challenges were encountered during the development of SADA v0.9. The primary issues and their resolutions are detailed below:

1. **Incorrect Area Adjustments**: Initially, adjustments to brightness and contrast after zooming into a selection were not applied correctly. The solution involved accurately calculating the coordinates and dimensions of the selected areas and ensuring they were handled consistently across different operations.

2. **Maintaining Selection States**: Ensuring that the selection state was preserved correctly between different operations (e.g., selecting an area, zooming in, then adjusting brightness) was challenging. The solution was to implement logic to differentiate between different user actions and apply adjustments appropriately.

3. **GUI Synchronization**: Keeping the GUI elements in sync with the underlying image data required careful management of state and event handling. This was achieved by implementing robust event callbacks and updating the canvas appropriately after each operation.

These challenges required in-depth debugging, adding logging statements, and iteratively refining the logic to achieve the desired functionality.

4. Detailed Function Implementations

The following sections detail the implementations of critical functions within SADA v0.9.

Image Selection and Zoom Functions:

- `enable_selection`: Allows users to select an area using polygon points.
- `enable_standard_selection`: Allows users to select a rectangular area.
- `zoom_to_selection`: Zooms into the selected area, resizing it to fit the canvas.

Brightness and Contrast Adjustments:

- `adjust_brightness`: Adjusts the brightness of the selected area or the entire image if no selection is made.
- `adjust_contrast`: Adjusts the contrast of the selected area or the entire image if no selection is made.

These functions involved calculating offsets, managing selections, and applying image enhancements correctly.

Code Snippets:

```python
def adjust_brightness(self, value):
    if self.original_image:
        factor = float(value) / 100
        base_image = self.original_image.copy() if self.analyzed_image is None else self.analyzed_image.copy()
        enhancer = ImageEnhance.Brightness(base_image)
        enhanced_image = enhancer.enhance(factor)

        if self.zoomed_selection_coords:
            left, top, right, bottom = self.zoomed_selection_coords
            left = max(int(left), 0)
            right = min(int(right), self.original_image.width)
            top = max(int(top), 0)
            bottom = min(int(bottom), self.original_image.height)

            cropped_image = enhanced_image.crop((left, top, right, bottom))
            zoomed_image = cropped_image.resize(self.image.size, Image.LANCZOS)
            self.image = zoomed_image
        else:
            self.image = enhanced_image

        self.image_processing_gui.display_image_on_canvas(self.image)
        self.update_status("Brightness adjusted")
```

This function showcases the logic for applying brightness adjustments to a zoomed selection.

5. Conclusion and Future Work

The development of SADA v0.9 has successfully improved the image selection, zoom, and adjustment functionalities, addressing key issues from previous versions.

The application now provides a more robust and user-friendly experience, ensuring that adjustments are applied accurately and consistently.

Future Work:

Future versions of SADA will focus on further enhancing the anomaly detection capabilities, improving performance for larger images, and adding more image processing features such as color correction and advanced filtering options.

Additionally, extensive user testing will be conducted to gather feedback and refine the user interface and experience.

This report provides a comprehensive overview of the development process, challenges faced, and solutions implemented, serving as a valuable reference for future development efforts.