ColorFaintGray GUI

User Manual and Reference Guide

Yogesh Wadadekar

Development Year: 2025

August 6, 2025

Abstract

ColorFaintGray GUI is a PyQt6-based graphical interface for GNU Astronomy Utilities' `astscript-color-faint-gray` script. This application enables astronomers to create high-quality color astronomical images from separate R, G, and B channel FITS files using an intuitive interface with advanced parameter controls, image caching, comparison tools, and comprehensive workflow management. The interface addresses the challenge of visualizing the full dynamic range of astronomical data by implementing the asinh transformation technique with grayscale backgrounds for faint features.

Contents

1 Introduction

1.1 Overview

Astronomical images span an enormous dynamic range, from bright stellar cores with high signal-to-noise ratios to faint diffuse structures barely above the noise level. Traditional color imaging techniques often saturate bright regions or lose faint details in black backgrounds. ColorFaintGray GUI addresses this fundamental challenge by providing an intuitive interface to the powerful `astscript-color-faint-gray` algorithm.

The application implements the asinh transformation technique described by Lupton et al. (2004)[1] for bright regions while using inverse grayscale for noisy background areas. This approach, detailed in Infante-Sainz & Akhlaghi (2024)[2], enables simultaneous visualization of both high and low surface brightness features within the same image.

[Screenshot: Main application interface showing parameter panel, tabbed workspace, and generated color image]

Figure 1: Main ColorFaintGray GUI interface with parameter controls on the left and tabbed workspace on the right.

1.2 Key Features

- **Complete Parameter Coverage**: Access to all astscript-color-faint-gray parameters including input HDUs, weight factors, minimum values, and zeropoint adjustments
- **Intuitive Parameter Control**: Real-time adjustment of asinh transformation parameters with immediate visual feedback
- **Advanced Input Control**: Individual HDU specification, weight scaling, minimum clipping, and zeropoint calibration for each channel
- **Robust Preset System**: Save, load, and manage parameter combinations with complete metadata and automatic validation
- **Automatic Image Caching**: All generated images cached with metadata for easy comparison and workflow management
- **Multi-Image Comparison**: Side-by-side comparison of up to 6 different parameter combinations

[1] Lupton, R., Blanton, M. R., Fekete, G., et al. 2004, PASP, 116, 133
[2] Infante-Sainz, R. and Akhlaghi, M. 2024, Research Notes of the AAS, 8, 10

- **Command Integration**: View and copy exact command-line equivalents for reproducible workflows with full parameter sets
- **Advanced Visualization**: Support for both traditional color and innovative grayscale background modes
- **Error Handling**: Comprehensive validation and user-friendly error reporting for all operations

2 Installation and Setup

2.1 System Requirements

- Python 3.12 or later
- PyQt6 $\geq$ 6.5.0
- GNU Astronomy Utilities with `astscript-color-faint-gray`
- Scientific Python packages: numpy, astropy, matplotlib, Pillow
- Sufficient disk space for image caching (recommended: 1GB+)

2.2 Installation Process

1. **Verify GNU Astronomy Utilities**:

```
which astscript-color-faint-gray
astscript-color-faint-gray --help
```

2. **Install Python Dependencies**:

```
pip install -r requirements.txt
```

3. **Launch Application**:

```
python main.py
```

2.3 Configuration

The application creates a configuration directory at `~/.config/astscript-color-faint-gray/` containing:

- `config.json`: Application settings and defaults
- `presets/`: Saved parameter presets
- `command_history.json`: Command execution history

3 Interface Overview

3.1 Main Components

The ColorFaintGray GUI features a modern tabbed interface with five primary areas:

3.1.1 Parameter Panel (Left Side)

The parameter panel contains all image processing controls organized into collapsible sections:

- **Input Settings**: File selection, HDU specification, weight factors, minimum values, and zeropoint adjustments
- **Basic Parameters**: Core transformation controls (qbright, stretch, contrast, gamma)
- **Advanced Parameters**: Color/grayscale thresholds, quality settings, and transformation options
- **Output Settings**: File format, destination options, and quality control
- **Command Display**: Real-time command preview with all parameters
- **Action Buttons**: Generate, Reset, comprehensive Preset management

[Screenshot: Parameter panel showing basic and advanced controls]

Figure 2: Parameter panel with basic and advanced control sections.

3.1.2 Tabbed Workspace (Center)

The main workspace consists of four specialized tabs:

Image Loader Tab Handles input file selection and validation:

- Individual file browsers for R, G, B channels
- Automatic FITS validation with status indicators
- File information display with dimensions and statistics
- Optional thumbnail previews for each channel

Preview Tab Interactive display of generated images:

- High-quality rendering with smooth zooming

- Mouse wheel zoom with cursor-centered scaling
- Click-and-drag panning for large images
- Image metadata overlay and statistics

Cache Grid Tab Thumbnail browser for cached images:

- Grid layout of automatically generated thumbnails
- Click thumbnails to restore parameters and view full images
- Search and filter capabilities
- Cache management and export tools

Compare Tab Multi-image comparison interface:

- Side-by-side display of up to 6 images
- Synchronized zooming and panning
- Parameter difference analysis
- Export comparison data

[Screenshot: Four-tab interface showing Image Loader, Preview, Cache Grid, and Compare tabs]

Figure 3: Main tabbed workspace showing all four interface tabs.

4 Basic Workflow

4.1 Loading Images

1. Navigate to the **Image Loader** tab (active by default)
2. Click **Browse** buttons to select FITS files for each channel:
   - **R Channel**: Red wavelength data (e.g., r-band, I-band)
   - **G Channel**: Green wavelength data (e.g., g-band, V-band)
   - **B Channel**: Blue wavelength data (e.g., u-band, B-band)
3. Verify successful loading via green checkmark indicators

4. Review file information in the details panel

[Screenshot: Image Loader tab with three loaded FITS files showing checkmarks and file information]

Figure 4: Image Loader tab with successfully loaded R, G, B channel files.

Important Considerations:

- All images must have identical dimensions
- Compatible world coordinate systems are recommended
- Properly calibrated data (flat-fielded, dark-subtracted) yields best results

4.2 Parameter Adjustment

The ColorFaintGray GUI provides comprehensive control over all astscript-color-faint-gray parameters, organized into logical sections for efficient workflow management.

4.2.1 Input Settings

Input Files and HDUs Each channel (R, G, B) can specify individual HDU numbers for multi- extension FITS files:

- **HDU Specification**: Individual HDU number for each channel (default: 0)
- **Multi-Extension Support**: Access any image extension within FITS files
- **Flexible Input**: Mix different HDUs from the same or different files

Weight Factors (0.001–1000, default: 1.0) Channel-specific scaling factors applied before processing:

- **Photometric Calibration**: Compensate for different filter sensitivities
- **Color Balance**: Adjust relative channel contributions
- **Exposure Scaling**: Account for different exposure times

Minimum Values (auto or manual) Pixel value thresholds applied to each channel:

- **Noise Clipping**: Remove negative noise artifacts

- **Background Subtraction**: Set appropriate zero levels
- **Dynamic Range Control**: Optimize for specific brightness ranges

Zeropoint Adjustments (default: 0.0) Magnitude zeropoint corrections for photometric accuracy:

- **Photometric Calibration**: Apply instrumental magnitude corrections
- **Filter Response**: Compensate for effective wavelength differences
- **Atmospheric Extinction**: Include extinction corrections

4.2.2 Basic Parameters

qbright (0–100, default: 1.0) Controls enhancement of bright features. Higher values make bright regions (star cores, galaxy centers) more prominent while preserving color balance.

stretch (0–100, default: 1.0) Linear stretching parameter for faint features. Increasing this value reveals more faint background details and diffuse structures.

contrast (0–100, default: 3.0) Linear contrast adjustment applied globally. Higher values increase overall image contrast without affecting the asinh transformation.

gamma (0.1–10.0, default: 0.8) Nonlinear brightness adjustment using gamma correction. Values > 1 brighten the image, < 1 darken it. Applied after linear processing.

4.2.3 Advanced Parameters

colorval (auto-estimated) Threshold separating color and black regions. Pixels above this value appear in full color using RGB data.

grayval (auto-estimated) Threshold separating black and grayscale regions. Pixels below this value appear in inverse grayscale.

coloronly (checkbox, default: false) When enabled, eliminates grayscale regions entirely. Background appears either in color or pure black.

forcecolor (checkbox, default: false) Forces color mode for all pixels above minimum thresholds, useful for ensuring consistent color representation across the image.

quality (1–100, default: 95) Output image quality setting affecting compression and file size for formats that support quality adjustment.

[Figure: Grid showing same galaxy with different parameter values demonstrating effects of qbright, stretch, gamma adjustments]

Figure 5: Parameter effects demonstration: same input with varying qbright, stretch, and gamma values.

4.3 Image Generation

1. Ensure all three channel files are loaded and validated
2. Configure input settings if using non-default HDUs, weights, minimums, or zeropoints
3. Adjust basic parameters (qbright, stretch, contrast, gamma) as desired using the Parameter Panel
4. Set advanced parameters (colorval, grayval, coloronly, etc.) if specific output characteristics are required
5. Specify output path and format in the output settings
6. Click the large green **Generate Image** button
7. Monitor progress in the progress dialog with real-time status updates
8. Review results in the automatically opened Preview tab

Important Notes:

- Output path specification is now mandatory and automatically validated
- All parameters are preserved for reproducibility and preset creation
- Command generation includes complete parameter set for external use
- Error handling provides detailed feedback for any processing issues

Alternative generation methods:

- Keyboard: F5 or Ctrl+G
- Menu: Tools → Generate Image

4.4 Viewing and Evaluation

The Preview tab provides comprehensive image viewing capabilities:

- **Zoom**: Mouse wheel for cursor-centered scaling
- **Pan**: Left-click drag to navigate large images
- **Fit**: Right-click menu option to fit image to window
- **Information**: Pixel values and coordinates on mouse hover

[Screenshot: Preview tab showing generated color image with zoom controls and information overlay]

Figure 6: Preview tab displaying generated color image with interactive viewing controls.

5 Advanced Features

5.1 Image Caching System

All generated images are automatically cached with complete metadata:

- **Automatic Storage**: Images saved as high-quality TIFF files
- **Thumbnail Generation**: PNG thumbnails for quick browsing
- **Parameter Preservation**: Exact parameter values stored
- **Command History**: Complete command-line equivalents saved
- **Configurable Limits**: Default 25 images (adjustable in settings)

5.1.1 Cache Grid Navigation

The Cache Grid tab displays all cached images as interactive thumbnails:

1. Click any thumbnail to view full-size image in Preview tab
2. Parameters are automatically restored to the Parameter Panel
3. Right-click thumbnails for context menu options:
    - Add to Comparison
    - Copy Command

- Show Information
- Delete Entry

[Screenshot: Cache Grid tab showing thumbnails of multiple cached images with timestamps and parameter summaries]

Figure 7: Cache Grid tab displaying thumbnails of automatically cached images.

5.2 Multi-Image Comparison

The Compare tab enables systematic evaluation of different parameter combinations:

5.2.1 Adding Images to Comparison

- **From Current**: Tools → Add Current to Comparison (Ctrl+M)
- **From Cache**: Right-click thumbnail → Add to Comparison
- **Direct Generation**: Generate with different parameters and add each result

5.2.2 Comparison Features

- **Layout Options**: Grid (2×3), horizontal, or vertical arrangements
- **Synchronized Viewing**: Optional synchronized zoom and pan across all images
- **Parameter Analysis**: Automatic detection and highlighting of parameter differences
- **Export Options**: Save comparison images and analysis data

[Screenshot: Compare tab showing 4 images side-by-side with parameter difference analysis panel]

Figure 8: Multi-image comparison interface with parameter difference analysis.

5.3 Preset Management

The ColorFaintGray GUI includes a comprehensive preset management system that enables efficient reuse of successful parameter combinations with complete validation and metadata tracking.

5.3.1 Creating Presets

1. Configure all desired parameters in the Parameter Panel (including input settings, basic parameters, advanced parameters, and output options)
2. Click **Save Preset** button or use Presets → Save Current as Preset (Ctrl+Shift+S)
3. Enter descriptive name and optional description in the save dialog
4. Preset automatically saved as JSON file with complete parameter set and metadata
5. Validation ensures all required parameters are captured

5.3.2 Loading and Managing Presets

- **Quick Load**: Use preset dropdown menu in Parameter Panel for immediate access
- **Load Preset Button**: Browse and load preset files with automatic validation
- **Management Interface**: Access comprehensive preset manager via Presets → Manage Presets (Ctrl+P)
- **Automatic Validation**: All loaded presets validated for parameter completeness and compatibility
- **Error Recovery**: Robust handling of corrupted or incomplete preset files

5.3.3 Preset Features

- **Complete Parameter Coverage**: All parameters saved including input settings, HDU specifications, weights, and advanced options
- **Metadata Preservation**: Creation date, description, and parameter summaries automatically stored
- **Cross-Platform Compatibility**: JSON format ensures presets work across different systems
- **Import/Export**: Share preset collections between users and installations
- **Backup and Recovery**: Automatic backup of preset directory during operations

5.3.4 Recommended Preset Categories

- **Galaxies - Elliptical**: Optimized for smooth elliptical galaxy structure with emphasis on outer isophotes
- **Galaxies - Spiral**: Balanced parameters for spiral arm structure and nuclear regions
- **Star Fields**: Stellar color preservation with minimal artifacts in dense fields
- **Nebulae - Emission**: Enhanced contrast for Hα and other emission line regions
- **Nebulae - Reflection**: Optimized for blue reflection nebulae and dust lanes
- **Deep Field - Survey**: Maximum faint feature enhancement for survey-depth observations
- **Deep Field - Ultra-deep**: Specialized parameters for ultra-deep field observations
- **Custom - High Resolution**: Optimized for high spatial resolution observations (HST, JWST)

6 Parameter Reference

6.1 Mathematical Foundation

The asinh transformation implements the algorithm described by Lupton et al. (2004):

$$I_{rgb} = \frac{R + G + B}{3} \tag{1}$$

$$f = \frac{\operatorname{asinh}(\alpha \cdot q \cdot I_{rgb})}{\alpha} \tag{2}$$

where α is the stretch parameter and q is the qbright parameter.

Each channel is then scaled by:

$$\text{Channel}_{out} = \text{Channel}_{in} \times \frac{f}{I_{rgb}} \tag{3}$$

6.2 Parameter Effects and Ranges

Parameter	Range	Default	Primary Effect
qbright	0.001–100	1.0	Bright feature enhancement
stretch	0.001–100	1.0	Faint feature visibility
contrast	0.1–10	3.0	Linear contrast scaling
gamma	0.1–10	0.8	Nonlinear brightness curve
R/G/B HDU	0–999	0	FITS extension selection
R/G/B Weight	0.001–1000	1.0	Channel scaling factor
R/G/B Minimum	auto/manual	auto	Noise clipping threshold
R/G/B Zeropoint	±50	0.0	Magnitude calibration
colorval	auto/manual	auto	Color/black threshold
grayval	auto/manual	auto	Black/gray threshold
coloronly	boolean	false	Disable grayscale regions
forcecolor	boolean	false	Force color mode
quality	1–100	95	Output image quality

Table 1: Complete parameter reference with ranges and effects.

6.3 Object-Specific Recommendations

6.3.1 Galaxies and Extended Objects

Optimal for revealing faint outer structure and tidal streams:

- qbright: 1.0–2.0 (moderate bright enhancement)
- stretch: 1.5–3.0 (strong faint feature enhancement)
- gamma: 0.7–0.9 (slight darkening for better contrast)
- coloronly: false (preserve grayscale backgrounds)

6.3.2 Star Fields and Clusters

Emphasizes stellar colors while maintaining photometric accuracy:

- qbright: 0.5–1.0 (avoid oversaturation)
- stretch: 0.8–1.5 (moderate faint enhancement)
- gamma: 0.8–1.0 (preserve stellar brightness relationships)
- contrast: 2.0–3.0 (maintain color accuracy)

6.3.3 Nebulae and Emission Regions

Balances emission lines with continuum features:

- qbright: 1.0–1.5 (reveal internal structure)
- stretch: 2.0–4.0 (show faint emission)
- gamma: 0.6–0.8 (darken to enhance contrast)
- colorval: 0.1–0.5 (adjust color/gray balance)

7 Command-Line Integration

7.1 Enhanced Command Display and History

The Parameter Panel continuously displays the equivalent command-line invocation with complete parameter coverage:

```
astscript-color-faint-gray -g 0 \
  --rhdu 0 --ghdu 0 --bhdu 0 \
  --rweight 1.0 --gweight 1.0 --bweight 1.0 \
  --rmin auto --gmin auto --bmin auto \
  --rzero 0.0 --gzero 0.0 --bzero 0.0 \
  input_r.fits input_g.fits input_b.fits \
  --qbright 1.5 --stretch 2.0 --contrast 3.0 \
  --gamma 0.8 --quality 95 \
  --output color_image.pdf
```

7.1.1 Command Features

- **Complete Parameter Set**: All GUI parameters reflected in command line
- **Automatic Output Path**: Mandatory output specification prevents execution errors
- **Real-time Updates**: Command updates immediately as parameters change
- **Copy Functionality**: One-click copying for external use
- **Validation**: Commands guaranteed to be syntactically correct and complete

7.1.2 Command History Access

- **View History**: Tools → Command History (Ctrl+H)
- **Copy Current**: Tools → Copy Current Command (Ctrl+Shift+C)
- **Export History**: Save complete command log for documentation
- **Parameter Reconstruction**: Load previous commands to restore exact parameter states

7.2 Batch Processing Integration

Commands copied from the GUI can be used for automated processing with complete parameter specifications:

```
#!/bin/bash
# Batch process multiple image sets using GUI-optimized parameters

# Parameters optimized in GUI for galaxy fields
BASIC_PARAMS="--qbright 1.5 --stretch 2.2 --gamma 0.8 --contrast 3.0"
INPUT_PARAMS="--rhdu 0 --ghdu 0 --bhdu 0 --rweight 1.0 --gweight 1.0 --
   bweight 1.0"
ADVANCED_PARAMS="--quality 95"

# Process multiple fields with consistent parameters
for field in field1 field2 field3; do
    echo "Processing ${field}..."
    astscript-color-faint-gray -g 0 \
        $INPUT_PARAMS \
        ${field}_r.fits ${field}_g.fits ${field}_b.fits \
        $BASIC_PARAMS $ADVANCED_PARAMS \
```

```
        --output ${field}_color.pdf
done
```

7.2.1 Advanced Batch Processing

For complex workflows with varying parameters:

```
#!/bin/bash
# Process with different parameters for different object types

# Define parameter sets exported from GUI presets
declare -A PRESETS
PRESETS[galaxy]="--qbright 1.5 --stretch 2.2 --gamma 0.8"
PRESETS[starfield]="--qbright 0.8 --stretch 1.0 --gamma 0.9"
PRESETS[nebula]="--qbright 1.2 --stretch 2.8 --gamma 0.7"

# Apply appropriate preset based on object type
for object_type in galaxy starfield nebula; do
    for target in ${object_type}_*; do
        [[ -f ${target}_r.fits ]] || continue
        astscript-color-faint-gray -g 0 \
            --rhdu 0 --ghdu 0 --bhdu 0 \
            ${target}_r.fits ${target}_g.fits ${target}_b.fits \
            ${PRESETS[$object_type]} \
            --output ${target}_color.pdf
    done
done
```

8 Color Science and Visualization

8.1 Dynamic Range Challenge

Astronomical images typically exhibit pixel value distributions spanning 4-6 orders of magnitude. The ColorFaintGray approach addresses this by:

1. **Bright Regions**: Asinh transformation preserves colors while preventing saturation
2. **Intermediate Regions**: Smooth transition to pure black
3. **Faint Regions**: Inverse grayscale reveals low surface brightness features

[Figure: Same astronomical field shown with traditional black background vs. ColorFaintGray technique, highlighting revealed faint structures]

Figure 9: Dynamic range comparison: traditional color (left) vs. ColorFaintGray technique (right) showing revealed faint structures.

8.2 Color Accuracy Considerations

8.2.1 Photometric Calibration

For scientifically accurate colors:

- Use properly calibrated input images with consistent zero points
- Apply appropriate weight values for different filter sensitivities
- Consider atmospheric extinction corrections
- Validate colors against known stellar standards

8.2.2 Perceptual Adjustments

The gamma parameter enables perceptual optimization:

- Gamma = 1.0: Linear relationship (photometrically accurate)
- Gamma $<$ 1.0: Emphasizes faint features (better for extended objects)
- Gamma $>$ 1.0: Enhances bright regions (suitable for stellar fields)

9 Troubleshooting

9.1 Common Issues and Solutions

9.1.1 Installation Problems

"astscript-color-faint-gray not found"

- Verify GNU Astronomy Utilities installation
- Check PATH environment variable
- Test manual execution: `astscript-color-faint-gray -help`

Python dependency errors

- Ensure Python 3.12+ is installed
- Verify PyQt6 installation: `python -c "import PyQt6"`
- Reinstall requirements: `pip install -r requirements.txt -upgrade`

9.1.2 File Loading Issues

"Invalid FITS File" errors

- Verify file integrity with `astropy.io.fits`
- Check for proper FITS headers and image extensions
- Ensure files contain image data (not tables or spectra)

"Dimension Mismatch" errors

- All three input files must have identical pixel dimensions
- Use `astinfo` to check image sizes
- Resample images to common pixel grid if necessary

9.1.3 Generation Failures

Parameter validation errors

- Ensure qbright and stretch are not zero
- Check that all parameters are within valid ranges
- Reset to defaults and adjust incrementally

Memory or performance issues

- Use smaller test images for parameter optimization
- Ensure adequate free disk space for temporary files
- Close other memory-intensive applications

9.2 Performance Optimization

9.2.1 Image Size Considerations

Processing time scales approximately with the square of linear image dimension:

Image Size	Typical Processing Time	Memory Usage
1000×1000	10–30 seconds	100–200 MB
2000×2000	1–2 minutes	400–800 MB
4000×4000	5–10 minutes	1.5–3 GB
8000×8000	20–40 minutes	6–12 GB

Table 2: Performance scaling with image size.

9.2.2 Workflow Optimization

1. **Parameter Development**: Use heavily binned or cropped images for initial parameter exploration

2. **Cache Utilization**: Leverage automatic caching to avoid regenerating identical images
3. **Batch Processing**: Export optimized commands for processing multiple similar datasets

10 Best Practices and Workflow

10.1 Recommended Workflow

1. **Preparation Phase**:
   - Ensure input images are properly calibrated
   - Verify consistent photometric calibration across channels
   - Create working copies of original data
2. **Initial Parameter Exploration**:
   - Start with default parameters
   - Generate initial image to assess overall appearance
   - Use Cache Grid to track parameter experiments
3. **Systematic Optimization**:
   - Adjust qbright first to optimize bright feature appearance
   - Fine-tune stretch to reveal desired faint structure level
   - Use gamma for overall brightness balance
   - Adjust color/gray thresholds if needed
4. **Comparison and Validation**:
   - Use Compare tab to evaluate different approaches
   - Generate images with slight parameter variations
   - Validate colors against known standards
5. **Final Production**:
   - Generate high-quality image with optimized parameters
   - Save successful parameter set as named preset
   - Export command for documentation and reproducibility

10.2 Quality Control Guidelines

10.2.1 Scientific Accuracy

- Verify that stellar colors match expected values for known star types
- Check that galaxy colors are consistent with morphological types
- Ensure that color gradients are astrophysically meaningful
- Validate against published color-magnitude diagrams when possible

10.2.2 Aesthetic Considerations

- Balance between scientific accuracy and visual appeal
- Avoid over-stretching that introduces noise artifacts
- Ensure smooth transitions between color, black, and gray regions
- Consider the intended audience (scientific vs. outreach)

10.3 Documentation and Reproducibility

10.3.1 Parameter Recording

Always document successful parameter combinations:

- Save parameters as named presets with descriptive names
- Copy and save the exact command used for each final image
- Record the rationale for parameter choices
- Note any special considerations for the dataset

10.3.2 Data Provenance

Maintain complete records of the image processing chain:

- Original data sources and observation details
- Calibration procedures applied before ColorFaintGray processing
- Complete ColorFaintGray command with all parameters
- Software versions (Gnuastro, ColorFaintGray GUI, Python, PyQt6)

11 Keyboard Shortcuts

Action	Shortcut
Generate Image	F5, Ctrl+G
Open Images	Ctrl+O
Save Image	Ctrl+S
Reset Parameters	Ctrl+R
Manage Presets	Ctrl+P
Save Current as Preset	Ctrl+Shift+S
Command History	Ctrl+H
Copy Current Command	Ctrl+Shift+C
Add to Comparison	Ctrl+M
Switch to Image Loader	Ctrl+1
Switch to Preview	Ctrl+2
Switch to Cache Grid	Ctrl+3
Switch to Compare	Ctrl+4
Toggle Parameter Panel	F9
Application Help	F1

Table 3: Complete keyboard shortcut reference.

12 Technical Implementation

12.1 Software Architecture

ColorFaintGray GUI is built using a modular architecture:

- **Core Module**: Parameter management, command building, image caching
- **GUI Module**: PyQt6-based interface components
- **Utils Module**: File handling, error management, configuration

12.2 File Formats and Standards

12.2.1 Input Formats

- **FITS**: Primary format, single or multi-extension
- **HDU Support**: Flexible HDU specification for complex files
- **World Coordinates**: Preserves astrometric information when present

12.2.2 Output Formats

- **Cache**: High-quality TIFF files for internal storage
- **Export**: PDF (default), PNG, JPEG, TIFF options
- **Metadata**: JSON format for parameter and command storage

12.3 Performance Characteristics

12.3.1 Memory Usage

- Base application: 50–100 MB
- Per image processing: $\sim 3\times$ input file size
- Cache storage: Configurable, default 25 images
- Peak memory: Depends on largest processed image

12.3.2 Processing Pipeline

1. Input validation and file loading
2. Optional minimum value clipping
3. Weight and zero-point scaling
4. RGB mean calculation (stacking)
5. Asinh transformation application
6. Individual channel scaling
7. Maximum normalization
8. Enhancement (gamma/contrast/bias)
9. Color/gray region separation
10. Final image assembly

13 Recent Improvements and Version History

13.1 Version 1.1 (Current) - Enhanced Parameter Coverage

The latest version represents a significant expansion of the ColorFaintGray GUI capabilities, addressing user feedback and implementing comprehensive parameter support:

13.1.1 Major Enhancements

- **Complete Parameter Coverage**: All astscript-color-faint-gray parameters now accessible through the GUI interface
- **Enhanced Input Control**: Individual HDU specification, weight factors, minimum value settings, and zeropoint adjustments for each channel
- **Robust Preset System**: Completely functional preset save/load system with comprehensive validation and error handling
- **Improved Command Generation**: All commands now include mandatory output paths with automatic validation
- **Advanced Error Handling**: User-friendly error messages and automatic recovery for common issues

13.1.2 Bug Fixes and Stability Improvements

- **Preset Loading Crash Fix**: Resolved application crash when loading presets due to missing output file specifications
- **Command Validation**: Enhanced command builder ensures all required parameters are always included
- **Parameter Synchronization**: Fixed issues where GUI state and command generation could become inconsistent
- **File Handling**: Improved robustness for various FITS file formats and structures

13.1.3 User Interface Improvements

- **Organized Parameter Layout**: Logical grouping of input settings, basic parameters, and advanced options
- **Real-time Validation**: Immediate feedback on parameter validity and conflicts
- **Enhanced Tooltips**: Comprehensive help text for all parameters
- **Improved Workflow**: Streamlined parameter adjustment and preset management processes

13.2 Development Priorities

Future development focuses on:

- **Performance Optimization**: Enhanced processing speed for large images
- **Batch Processing**: GUI support for processing multiple image sets
- **Advanced Visualization**: Additional image analysis and comparison tools
- **Extended Format Support**: Support for additional input and output formats

14 Acknowledgments and References

14.1 Acknowledgments

ColorFaintGray GUI development is built upon the excellent work of:

- GNU Astronomy Utilities development team
- The astscript-color-faint-gray algorithm by Raúl Infante-Sainz and Mohammad Akhlaghi
- PyQt6 and the Qt framework for the graphical interface
- The Python scientific computing ecosystem

14.2 Key References

1. Lupton, R., Blanton, M. R., Fekete, G., et al. 2004, "Preparing Red- Green-Blue Images from CCD Data," PASP, 116, 133
2. Infante-Sainz, R. & Akhlaghi, M. 2024, "Gnuastro: Visualizing the Full Dynamic Range in Color Images," Research Notes of the AAS, 8, 10
3. Akhlaghi, M. & Ichikawa, T. 2015, "Noise-based Detection and Segmentation of Nebulous Objects," ApJS, 220, 1

14.3 Software Citation

When using ColorFaintGray GUI in research, please cite:

> This work made use of ColorFaintGray GUI v1.0, a Python/PyQt6-based interface for GNU Astronomy Utilities' astscript-color-faint-gray (Infante- Sainz & Akhlaghi 2024). The underlying color image algorithm implements the asinh transformation technique described by Lupton et al. (2004).

15 Appendix

15.1 Example Parameter Sets

Object Type	qbright	stretch	gamma	Notes
Elliptical Galaxies	1.0	2.0	0.8	Emphasize outer structure
Spiral Galaxies	1.2	1.8	0.9	Balance arms and nucleus
Star Forming Regions	0.8	2.5	0.7	Reveal emission structure
Globular Clusters	0.6	1.2	0.9	Preserve stellar colors
Deep Survey Fields	1.5	3.0	0.7	Maximum faint enhancement
Ultra-deep Fields	2.0	3.5	0.6	Extreme faint sensitivity
Planetary Nebulae	1.0	2.0	0.8	Balance central star
Reflection Nebulae	0.9	2.2	0.8	Preserve blue colors
Emission Nebulae	1.1	2.6	0.7	Enhance line emission
High-res (HST/JWST)	0.8	1.5	0.9	Preserve fine details

Table 4: Recommended parameter starting points for different astronomical objects, with enhanced coverage for modern datasets.

15.2 Troubleshooting Checklist

15.2.1 Before Seeking Help

1. Verify all input files are valid FITS format
2. Check that astscript-color-faint-gray executes independently
3. Try with default parameters first
4. Review error messages in dialogs and console output
5. Check available system memory and disk space
6. Ensure Python and PyQt6 versions meet requirements

15.2.2 Information to Include When Reporting Issues

- Operating system and version
- Python version (`python -version`)
- PyQt6 version (`pip show PyQt6`)
- GNU Astronomy Utilities version (`astscript-color-faint-gray -version`)
- Input file characteristics (dimensions, format, size)
- Complete error messages
- Steps to reproduce the problem

15.3 Configuration File Reference

The main configuration file (`config.json`) contains:

```
{
  "app": {
    "last_input_dir": "/path/to/input/directory",
    "last_output_dir": "/path/to/output/directory",
    "default_output_format": "PDF",
    "cache_size": 25,
    "cache_dir": "cache/",
    "astscript_path": "astscript-color-faint-gray"
  },
  "ui": {
    "parameter_panel_width": 300,
    "grid_thumbnail_size": 150,
    "window_geometry": "saved window position"
  },
  "parameters": {
    "qbright": 1.0,
    "stretch": 1.0,
    "contrast": 3.0,
    "gamma": 0.8
  }
}
```

—

For the most current information and updates, please visit the project repository and consult the GNU Astronomy Utilities documentation at https://www.gnu.org/software/gnuastro/.