SOUTHERN MACHINERY

Southern Machinery

S-L680/S-H680 Series

A Masterclass in AOI Programming Procedure

For the qualified SMT Engineer and Programmer.

The Path from PCB to Production: A Three-Phase Workflow

Foundation Setup

Create the program, scan the board, and establish high-precision alignment using Mark Points. This is the bedrock of a reliable program.

Core Programming

Enter Editing Mode to register components, build libraries, and configure the specific inspection algorithms that will find defects. This is the heart of the process.

Execution & Refinement

Run the program in Debugging Mode to eliminate false calls, then transition to Formal Inspection for live production monitoring.

This guide follows the complete, end-to-end procedure for transforming a new PCB into a robust, automated inspection program. Each step builds upon the last to ensure accuracy and efficiency.

Step 1: Initiating a New Program and Scanning the Board

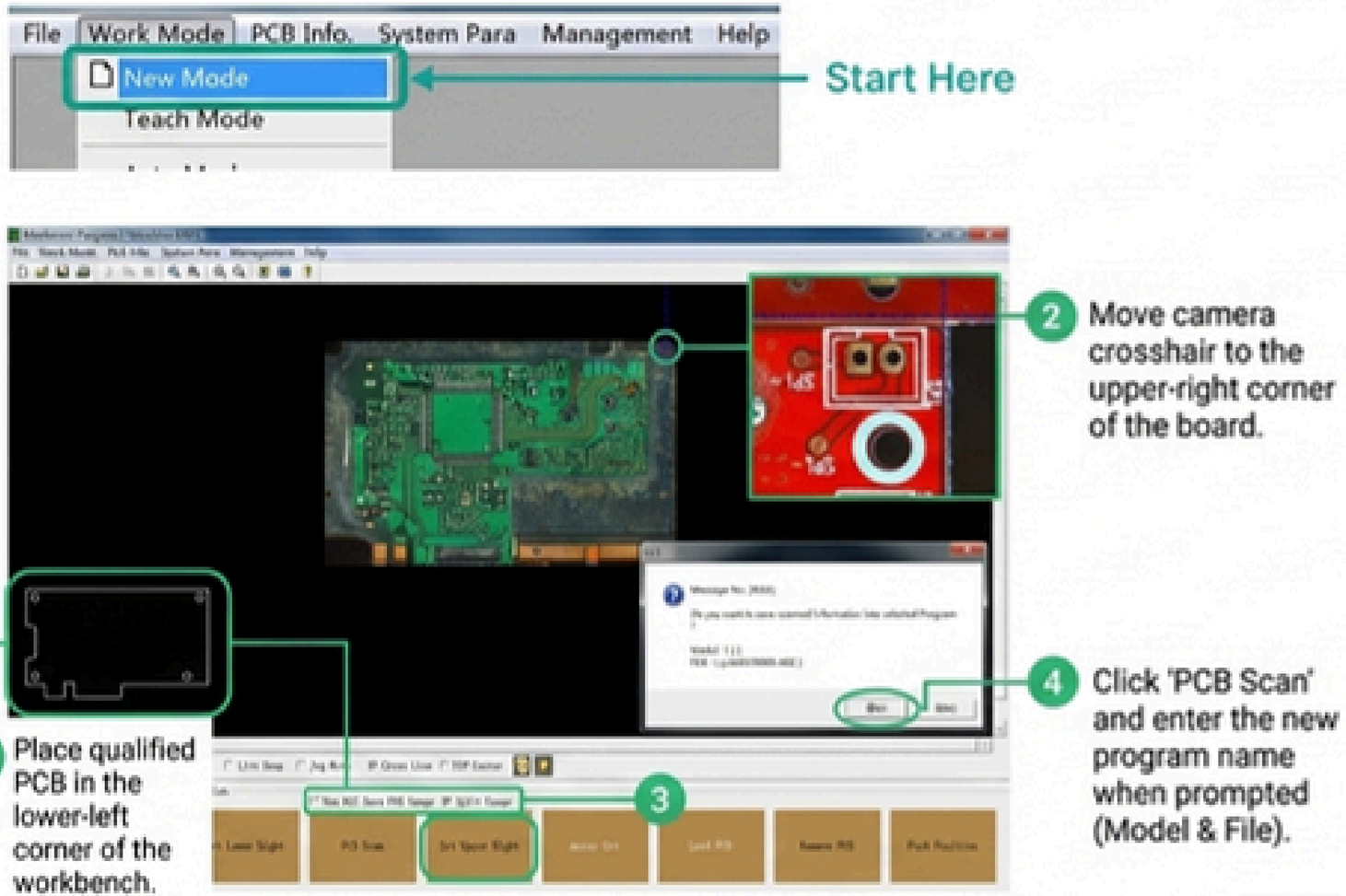
Procedure and Pro-Tip

Procedure:

- Begin by navigating to `Work Mode > New Mode`.
- Physically place and fix a 'golden' (known good) PCB on the workbench.
- Define the board's physical dimensions by setting the upper-right corner coordinate. The lower-left corner acts as the program's zero point.
- Initiate a full PCB scan. The system will stitch together FOV (Field of View) images to create a complete thumbnail for navigation.

Pro-Tip: For offline programming (OLP), check the 'Use OLP, Save PCB Image' option before scanning to save a high-resolution, uncompressed image.

Visual Walkthrough



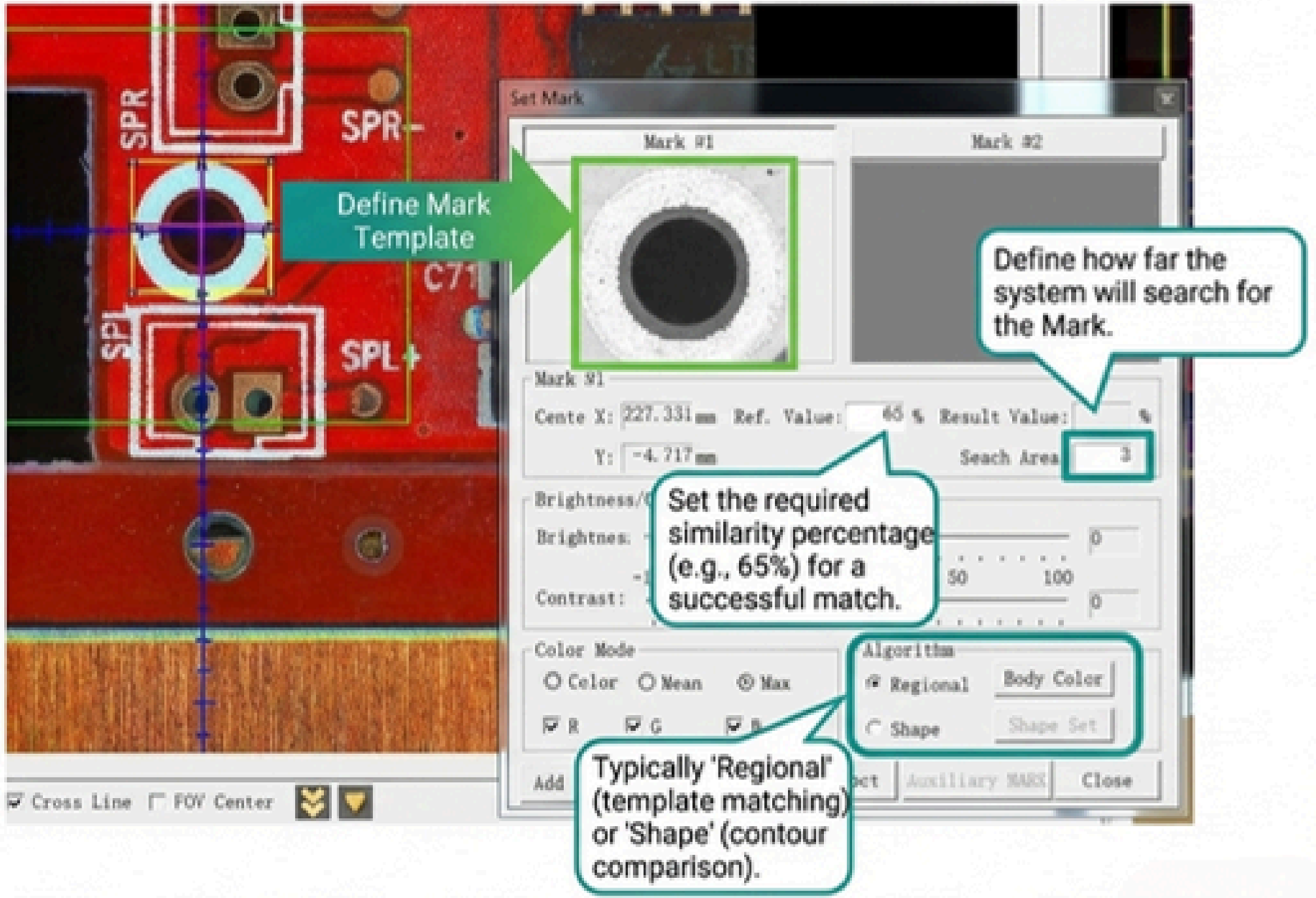
Step 2: Defining Mark Points for Precision Alignment

Objective: To correct for any positional or rotational offsets of the PCB during inspection.

Procedure:

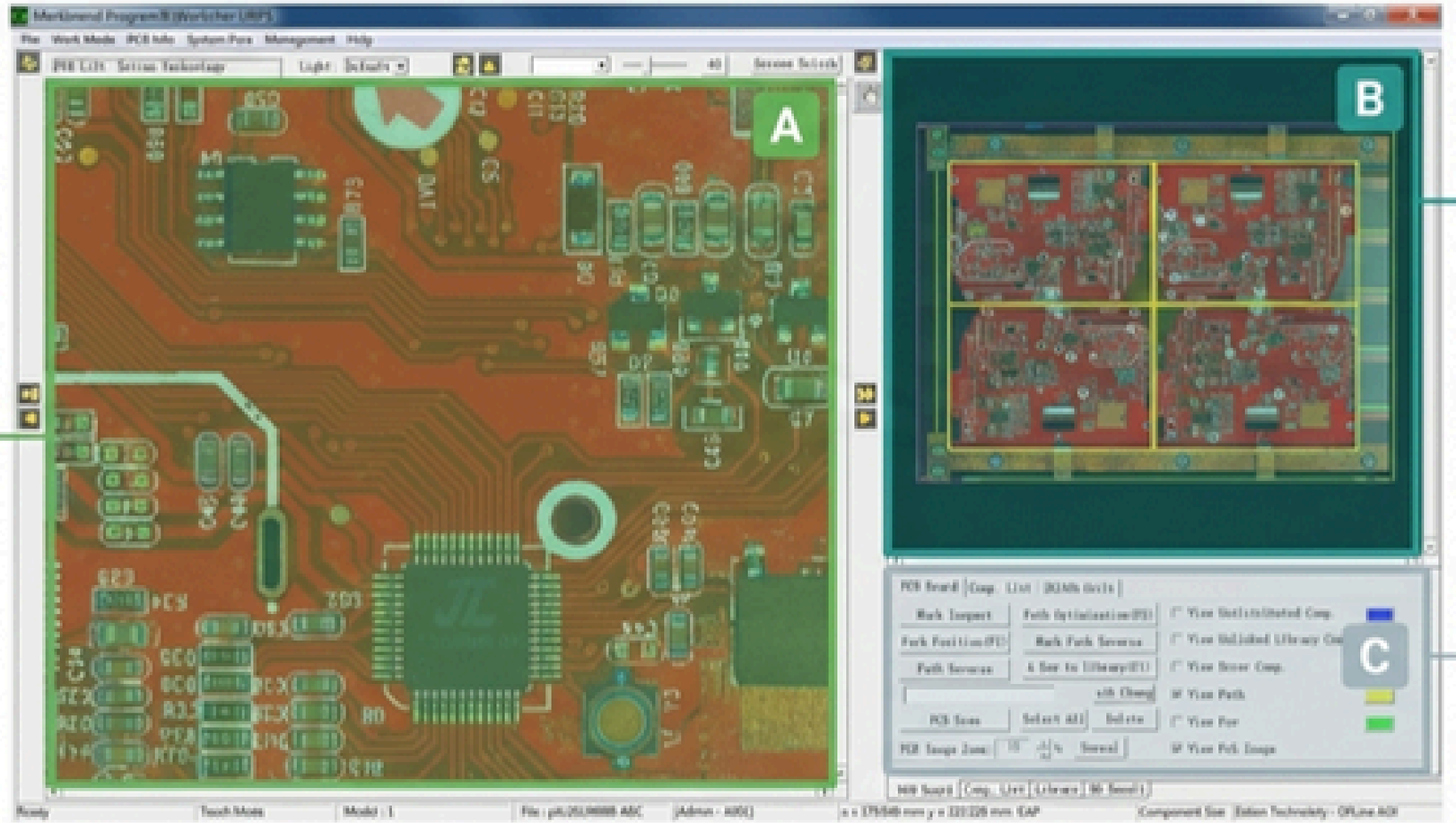
1. After the PCB scan, the system prompts you to set Mark Points.
2. Define at least two Marks, preferably at diagonal corners of the PCB for maximum accuracy.
3. Select a stable, high-contrast feature (e.g., fiducial, unique pad).
4. Draw a frame around the feature to define the Mark's template.
5. Set the matching parameters (Ref. Value, Search Area, Algorithm).
6. Repeat for Mark #2.

Expert Insight: If a Mark point has variable appearances (e.g., due to solder differences), consider adding multiple inspection templates ('grouping Mark points') to improve the matching success rate.



Step 3: Entering Editing Mode - The Programming Hub

Live FOV image for component registration and algorithm setup. All framing and direct interaction happens here.



Full board thumbnail for quick navigation. Click anywhere to move the camera. Also contains FOV path optimization tools.

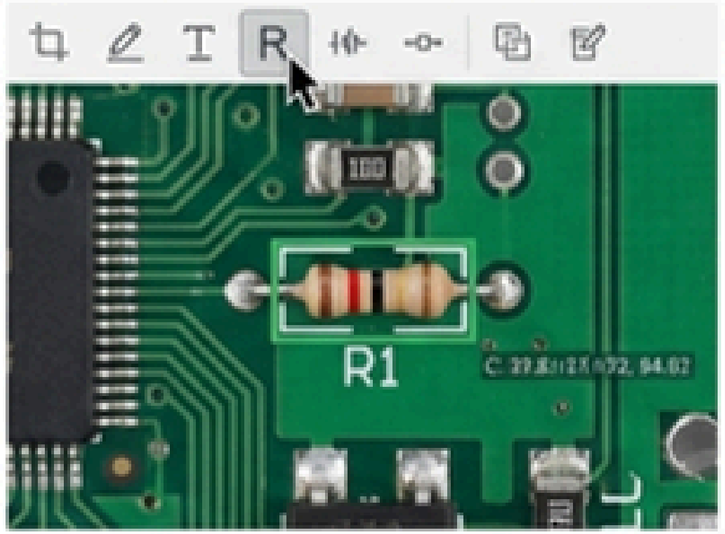
Switch between key data views: `PCB Board` (thumbnail), `Comp. List` (all registered components), `Library` (reusable standards), and `NG Result` (debugging).

Editing Mode is the **most critical interface**, where you define what to inspect and how. All component **registration**, **library** management, and **algorithm configuration** are performed here. The interface is structured for an efficient workflow, separating live image interaction from board-level navigation and data management.

Step 4: Registering Components - The Building Blocks of Inspection

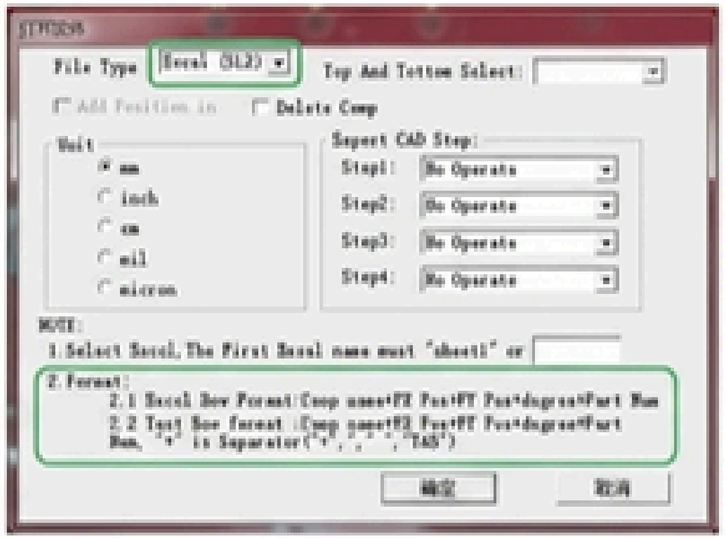
Component registration is the process of defining the coordinate position, size, and angle of every part to be inspected. The system offers multiple methods to suit your workflow.

Manual Frame Registration



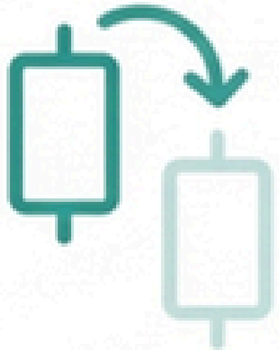
The most direct method. Select a component type from the toolbar and draw a frame around the component in the Image Working Area. Ideal for small numbers of components or parts not in the CAD file.

CAD Import



Highly efficient for complex boards. Import an EXCEL or TXT file with component name, coordinates, angle, and part number. The system automatically places virtual component points on the board, ready for programming.

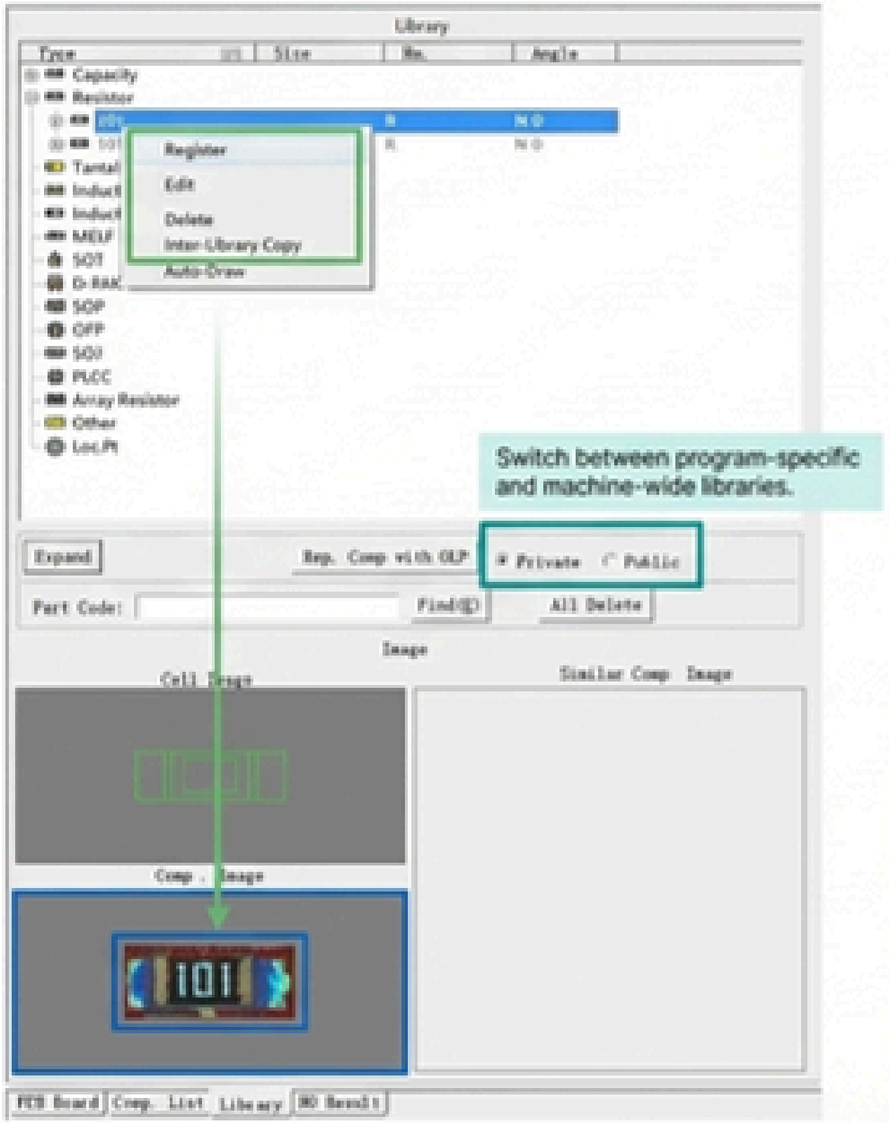
Copy/Cut/Paste Registration (Ctrl+C / Ctrl+X)



For repeating components. Select one or more registered components, copy them, and click on the new location to place a duplicate. If the original was linked to a library, the new one will be too.

Step 5: Mastering Component Libraries for Consistency and Speed

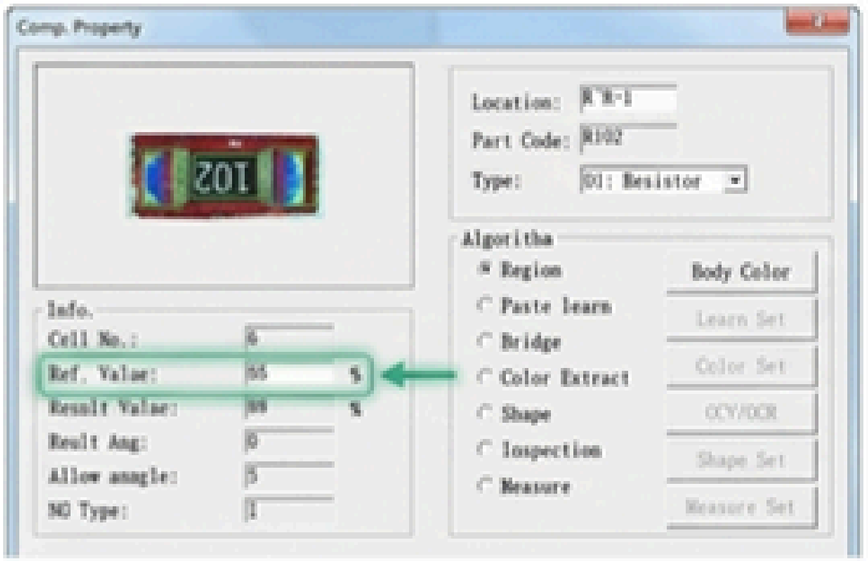
- **The Power of Libraries:** Storing component data (frames, algorithms, standards) in a library allows you to reuse it for every identical component, ensuring uniform inspection standards.
- **Private vs. Public Library:**
  - **Private Library**: Stored with the individual inspection program. Effective only within that program.
  - **Public Library**: Stored separately and shared across ALL inspection programs on the machine. The ideal place for common components (e.g., 0402 resistors, standard SOT-23s).
- **Workflow:**
  1. Register a component for the first time.
  2. Fine-tune its inspection algorithms and parameters.
  3. Register it into the Private or Public Library using a standardized Part Code (e.g., `R-0603-10K-5%`). (Ctrl+Q or right-click `Ins.Comp to Lib`).
  4. Use the library entry to program all other identical components. Any changes to the library entry will automatically update all linked components.



Step 6 (Part 1): Configuring Algorithms - Presence, Position & Polarity

Region Method (Template Matching)

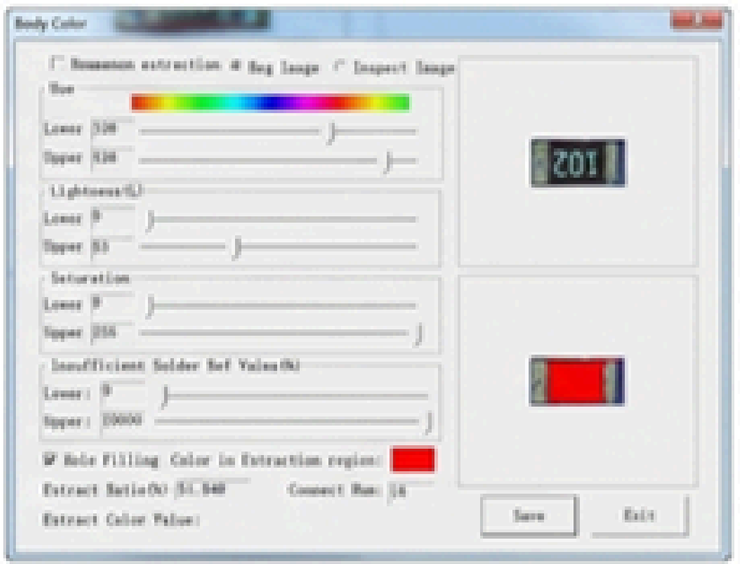
The most common algorithm for component presence and positioning. It compares the live image to a stored template.



- **Use For:** Missing parts, wrong parts, offset, skew, polarity (for asymmetrical parts).
- **Key Parameter:** Ref. Value (%) - The required similarity score for a PASS. A lower value is more tolerant; a higher value is stricter.

Body Color Extraction

A powerful supplement to the Region method. It isolates the component body by its color profile before performing the comparison, ignoring variable text on the component.

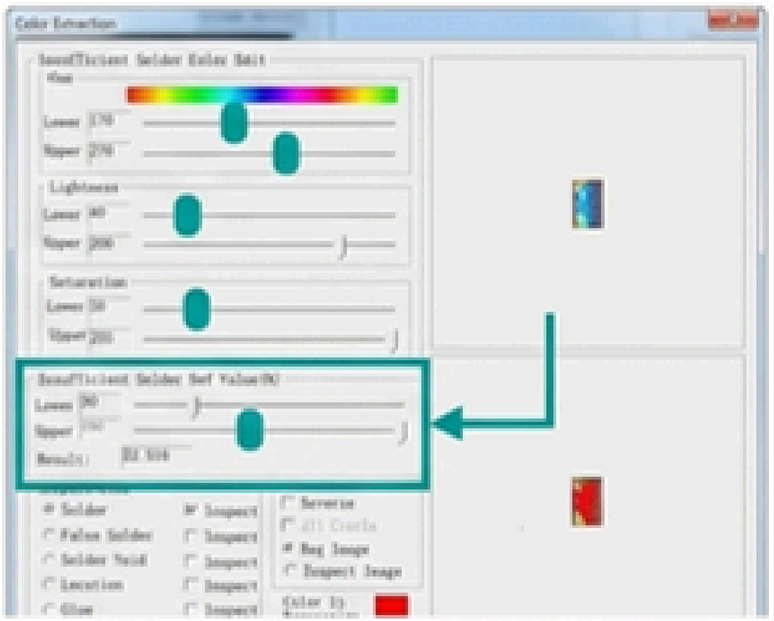


- **Use For:** Resistors or components where silkscreen markings vary but the body color is consistent.
- **Key Parameters:** HSL (Hue, Lightness, Saturation) ranges to define the target color.

Step 6 (Part 2): Configuring Algorithms - Solder Joints & Silkscreen

Color Extraction (for Solder)

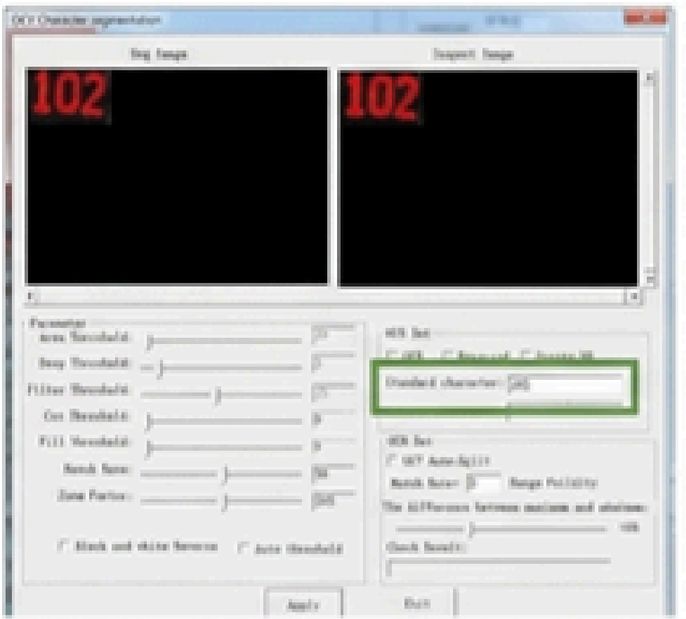
Analyzes the color properties of solder fillets to detect common soldering defects.



- **Use For:** Insufficient solder, excess solder, solder voids.
- **How it Works:** Define the HSL color range of good solder. The algorithm calculates the percentage of the inspection frame filled with that color and compares it to the upper and lower reference values.

OCV/OCR (for Silkscreen)

Verifies component markings to prevent wrong parts.



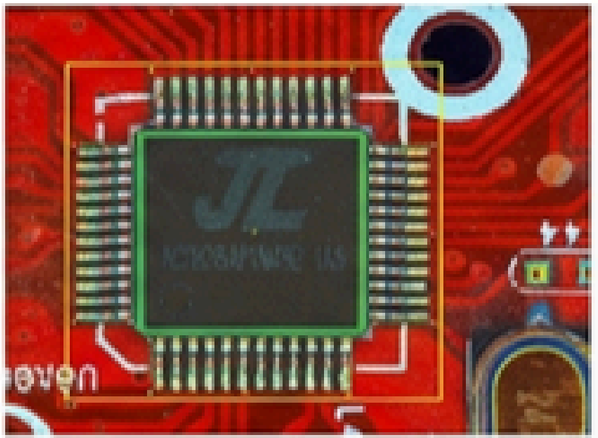
- **OCV (Optical Character Verification):** Compares the image of the text to a stored template image of the correct text.
- **OCR (Optical Character Recognition):** Reads the characters and compares the resulting string to a defined standard string (e.g., '102'). More precise for clear, standardized text.

Step 6 (Part 3): Configuring Algorithms - IC Short Circuits & Leads

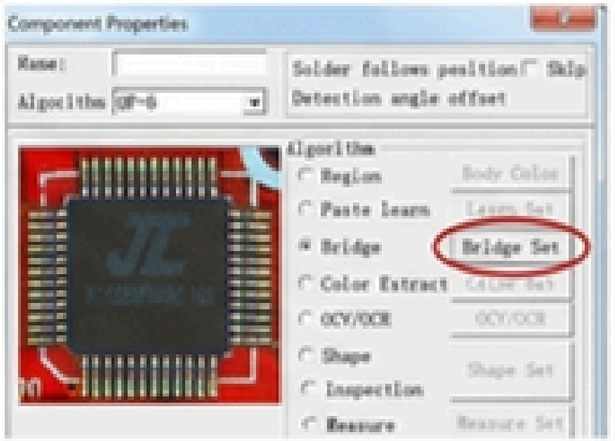
IC Inspection Workflow:

1. Register the IC using a combined frame type (e.g., `**QFP**`, `**SOP**`).
2. Select the `**Bridge**` algorithm and open the `**Bridge Set**` dialog.
3. Adjust the `**Seg. Threshold**` to clearly separate the pins from the background.
4. Click `**Lead Def.**` to define the location and orientation of the pins for one side of the IC. The system automatically creates inspection frames for each pin.
5. The system will now inspect for short circuits between pins and can also use color extraction on the generated pin frames to check for solder defects like insufficient solder or lifted leads.

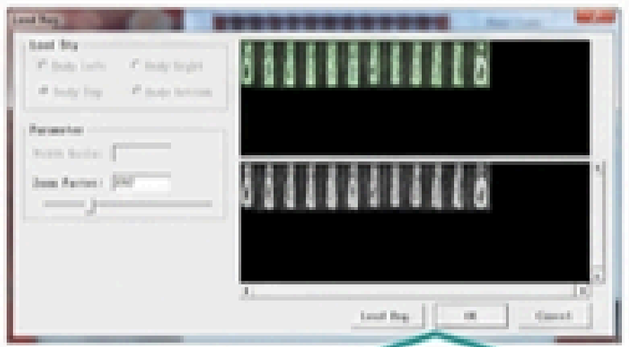
1. Start with a "Bridge" Inspection Frame:



2. Open Bridge Set Dialog:

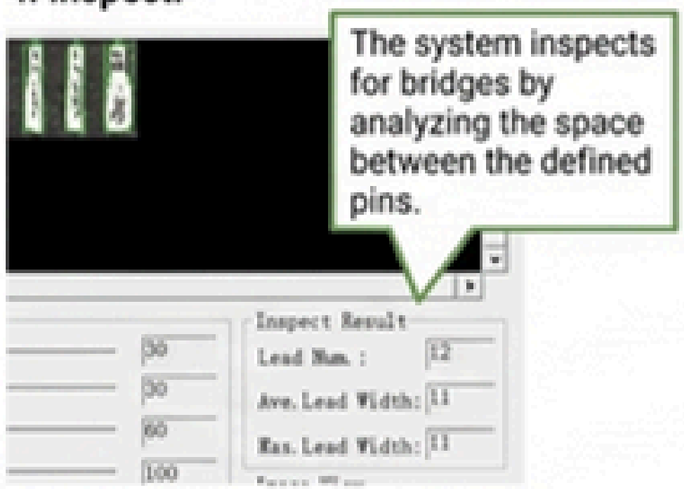


3. Define Leads:

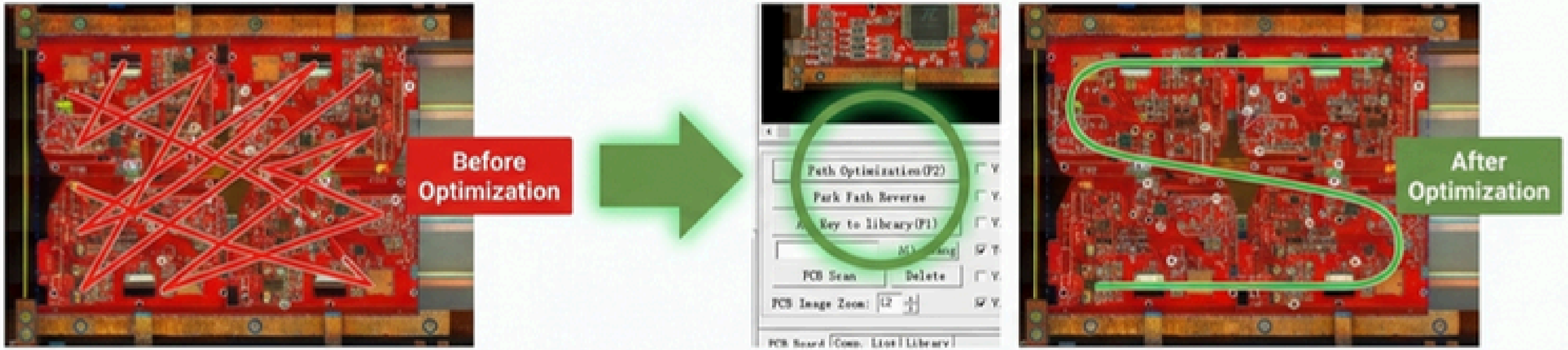


The system automatically segments the individual pins based on the segmentation threshold.

4. Inspect:



Step 7: Optimizing the Inspection Path for Maximum Throughput



Objective
To calculate the most efficient physical path for the camera to travel to capture all programmed components, minimizing inspection time.

Procedure
1. After all components have been registered, navigate to the **PCB Image** page in the right-hand pane.
2. Click the **Path Optimization (F2)** button.
3. The system will automatically calculate the minimum number of FOV movements required and plot the most efficient path.

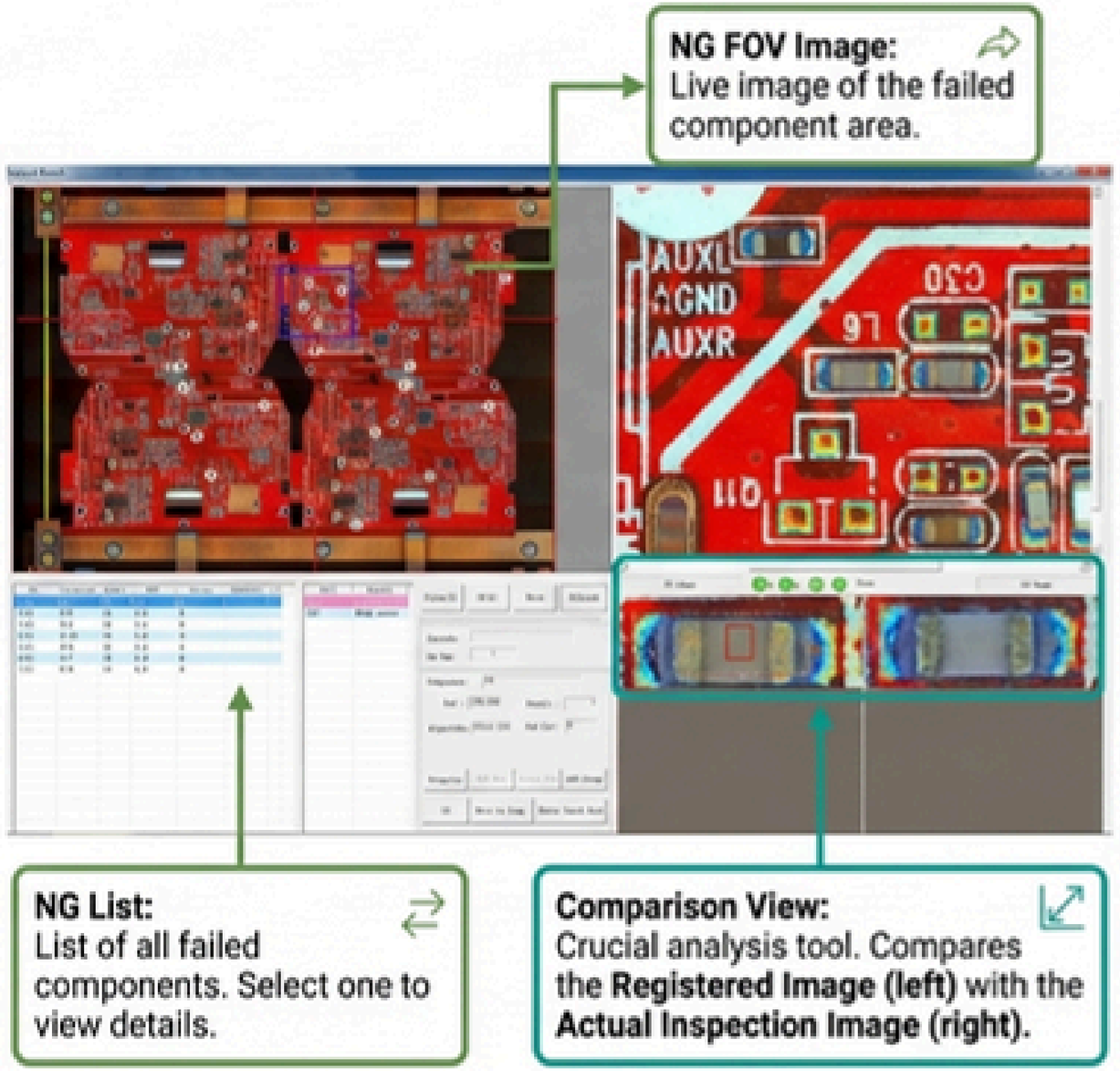
Additional Tools
- **Path Reversal**: Reverses the direction of the calculated path. Can sometimes save time depending on the start/end points.
- **Path Adjustment**: Manually define the FOV inspection order by entering the FOV serial numbers separated by commas for ultimate control.

Step 8: Debugging Mode - Refining Accuracy and Eliminating False Calls

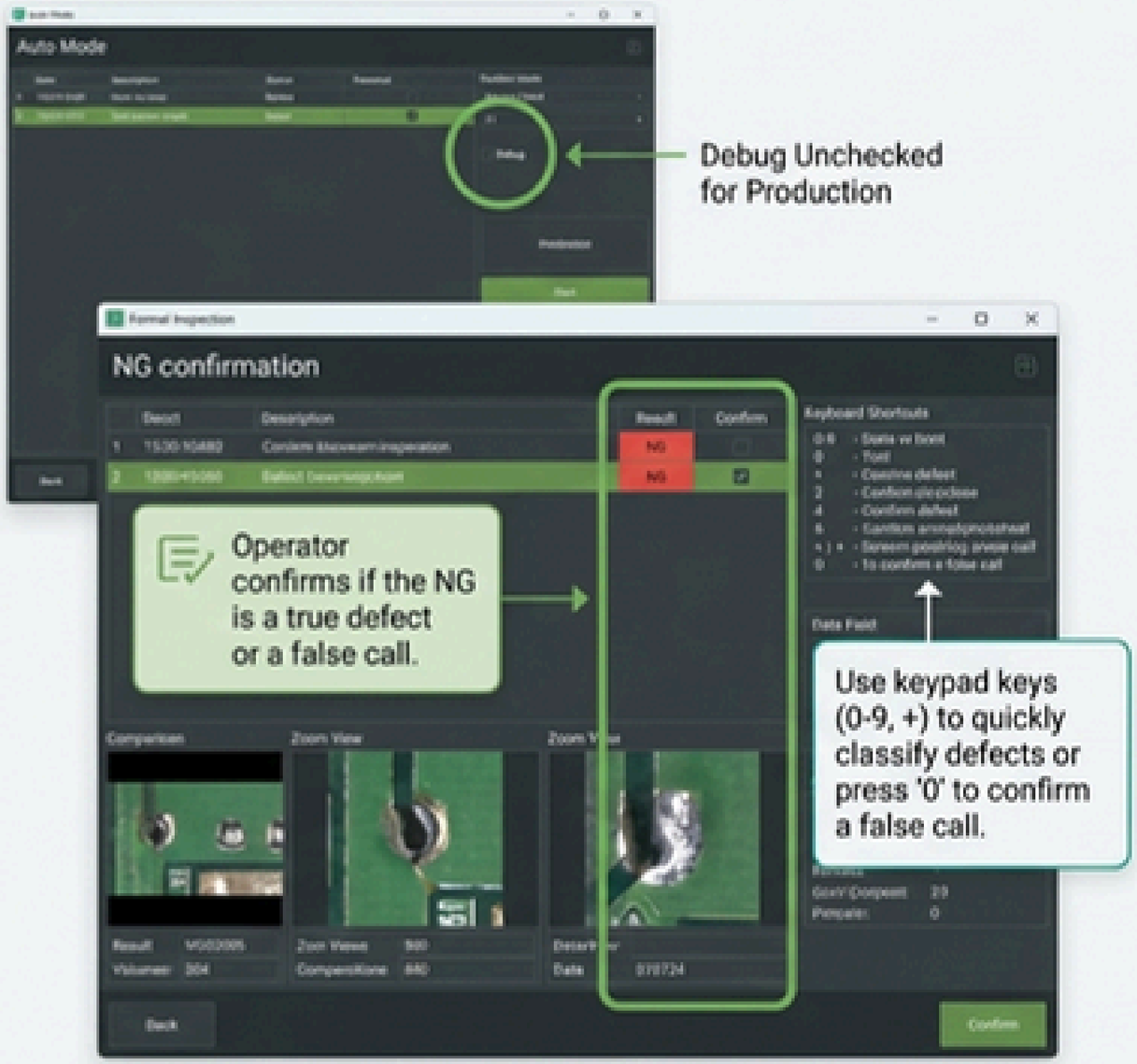
Purpose: To test the program on a real board and adjust parameters to catch all real defects while ignoring acceptable process variations (i.e., reduce false calls).

Common Debugging Actions for False Calls:

- **Problem:** Positional Offset.
- **Solution:** In the result screen, use the small arrow icons on the registered image to slightly adjust its position and click 'Refresh'.
- **Problem:** Minor Appearance Variation (e.g., different reflection).
- **Solution:** For Region/OCV methods, click `Add Group`. This adds the current image as a new acceptable template. The component will PASS if it matches *any* template in the group.
- **Problem:** Parameter Too Strict.
- **Solution:** Click `Property` to open the algorithm settings and adjust the `Ref. Value` (e.g., lower the similarity requirement) or tweak the color extraction ranges.



Step 9: Formal Inspection - Running the Program in Production



How to Start: In the `Auto Mode` screen, ensure the 'Debug' checkbox is **cleared**, then click 'Start.'

Key Differences from Debugging Mode:

No Parameter Changes: The operator cannot adjust algorithms or add groups. The program is locked for production.

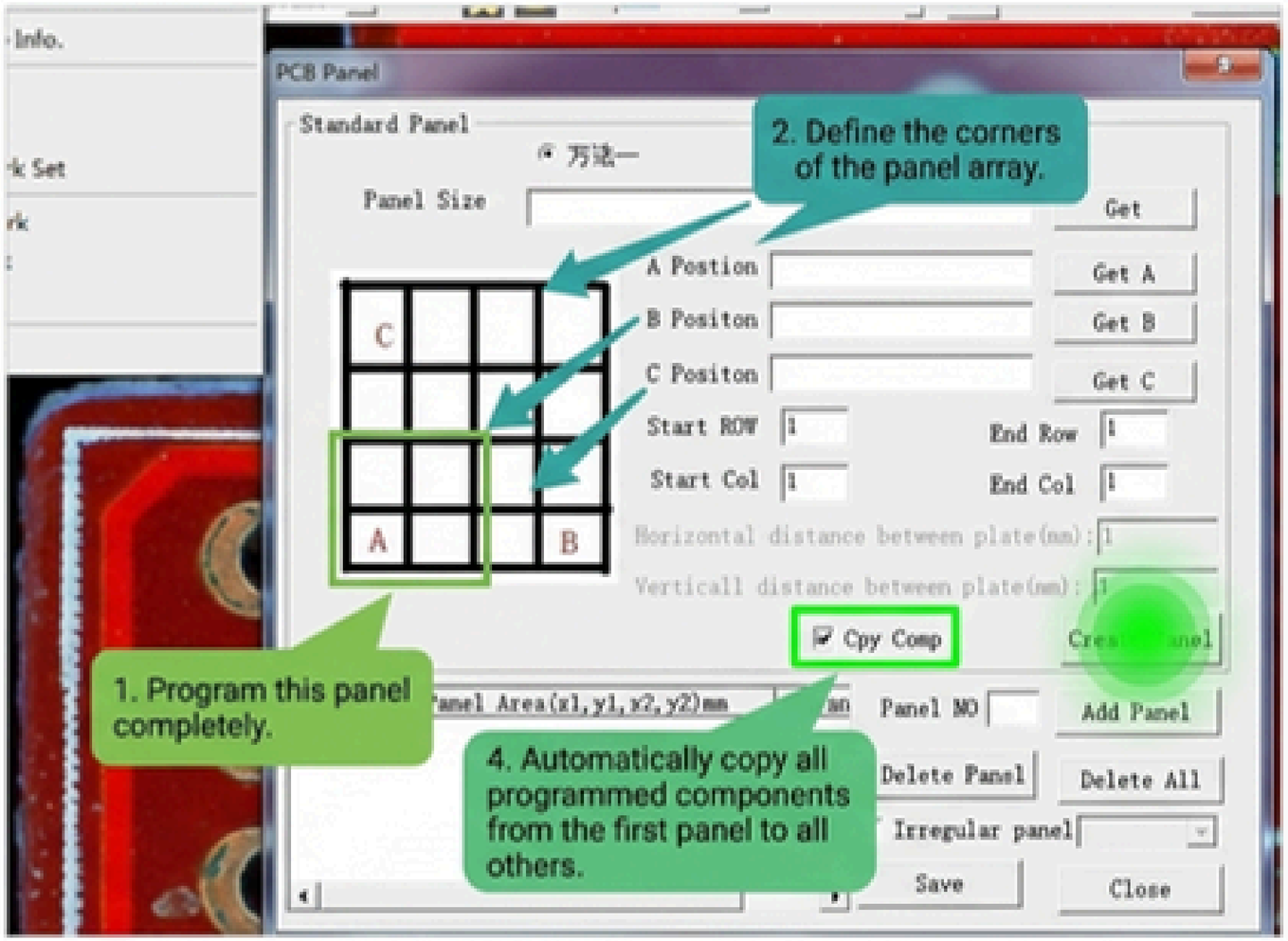
Continuous Operation: The system will automatically inspect board after board without stopping (unless configured otherwise).

Data Logging: If a database is enabled, all confirmed NG results are logged for SPC (Statistical Process Control) analysis.

Operator Workflow:

1. When an NG is found, this confirmation screen appears.
2. The operator uses the comparison images to verify the defect.
3. Confirm the NG type using the keyboard shortcuts.
4. Press `Enter` to complete the confirmation and allow the machine to continue to the next board.

Advanced Workflow: Programming Panelized (Jointed) PCBs



For boards manufactured in a panel (jigsaw), you don't need to program each sub-board individually. Use the dedicated PCB Panel tool for rapid programming.

Standard Panel Workflow:

1. First, fully program all components on a single sub-board (e.g., the bottom-left one).
2. Go to `PCB Info. > PCB Panel`.
3. Define the array by clicking on the same feature on three corner panels (A, B, C).
4. Enter the total number of rows and columns.
5. Check the `Cpy Comp` box.
6. Click `Create Panel`. The system will automatically populate all other sub-boards with the correctly positioned components.

The Southern Machinery AOI Programming Workflow: At a Glance

1. **New Program:** Define board size & scan.
2. **Set Marks:** Establish X, Y, and Theta correction.
3. **Enter Editing Mode:** The main programming environment.
4. **Register Comps:** Use Frame, CAD, or Copy methods.
5. **Utilize Libraries:** Create/use standards for consistency.
6. **Configure Algorithms:** Set parameters for Region, Color, OCV, Bridge, etc.
7. **Optimize Path:** Calculate the fastest inspection route (F2).
8. **Debug Program:** Run on a real board to find and fix false calls.
9. **Formal Inspection:** Go live in production.

Key Best Practices:

- **Start with a 'Golden Board':** Always use a known-good PCB for initial programming.
- **Leverage Public Libraries:** Build and maintain a robust Public Library for common parts to drastically reduce programming time on future projects.
- **Debug Thoroughly:** Invest time in the debugging phase. A few extra minutes adding unit groups or refining parameters can save hours of operator time dealing with false calls in production.