https://doi.org/10.37420/j.csias.2025.006

Research on the Development and Implementation of Image Processing Design Software Based on Dynamic Pixelation and Texture Replacement Technologies

Li Xu[1*];Qin Zou[2]

[1]Maria Curie Sklodowska University,PL

[2]BeiJing Institute Of Economics And Management,CHN

[*]Corresponding author,E-mail:xulidesign@163.com

Abstract

This research designs and develops a web-based interactive image processing design software, focusing on the application of dynamic rasterization and material map replacement technologies for real-time image effect generation and visual reconstruction. The system is implemented using pure front-end technologies, allowing users to adjust parameters for image rasterization, dynamic animation generation, and material map replacement. With the introduction of the "image dark area filling" mechanism and dynamic raster generation algorithm, the system significantly enhances the visual expressiveness of images in art reconstruction, providing an innovative approach for lightweight generative art and web interaction design.The system features real-time preview, dynamic animation rendering, and export capabilities, running entirely on the browser without the need for additional software installation and supporting multiple platforms. Through the "Letter A" dynamic raster art case, the system demonstrates its application in creative expression, image stylization, and interactive design, confirming its value in digital art creation, interactive art, and digital media education. This research offers new forms of visual expression for artistic creation and provides efficient, flexible technical support for interactive art and digital media education, advancing the innovative application of image processing in the field of digital art and interaction design.

Keywords

Dynamic Pixelation;Texture Replacement;Front-End Image Processing;Real-Time Interaction;Generative Art;Image Reconstruction

© By the author(s); licensee Mason Publish Group (MPG), this work for open access publication is under the Creative Commons Attribution International License (CC BY 4.0). (http://creativecommons.org/licenses/by/4.0/)

INTRODUCTION

With the development of digital art and interactive design, the limitations of traditional image processing tools in artistic creation have gradually become apparent. Over the years of artistic practice, the author has extensively used Adobe Photoshop, experimenting with its "Filter → Pixelate → Color Halftone" function to reconstruct images and create stylistic expressions. (Color halftone is an image processing technique that simulates traditional printing technology by breaking an image into a visual structure composed of colored dots, creating unique retro and abstract/halftone art effects.) While the color halftone function can generate effects such as mosaics, dotted textures, and other unique artistic and visual effects, there are a series of limitations in its practical use, especially in terms of artistic creation.

Firstly, the colored dots in Photoshop are automatically generated by the system and cannot be replaced with user-defined shapes or images, severely limiting the creator's visual control over the dot elements. Secondly, Photoshop does not allow dynamic parameter control for these dot elements. Creators cannot achieve animations, or adjust the density, size, or shape variations of the dots in real time, lacking interactivity and real-time feedback.

Based on these issues, the author conceived the idea of developing a simple-to-use, web-based image generator that requires no installation and offers interactivity and real-time feedback. This software would focus on the real-time generation and visual operation of halftone image effects. Users would not only be able to freely upload images and generate halftone effects, but also customize each dot with a substitute graphic (such as patterns, symbols, or custom material images). Additionally, users could control dynamic attributes such as motion, size, density, and deformation of these elements through parameters. The software would also support material texture mapping and visual parameter animation, expanding the expressive potential of images in the direction of "dynamic halftoning." By integrating image processing techniques with interactive artistic expression, the aim is to explore new possibilities for visual image reconstruction, bringing richer and more dynamic forms of expression to digital art and design, and showcasing great innovative potential in the field of visual arts.

1.DESIGN CONCEPT

The author aims to create an image generation and creative platform that offers a highly personalized experience by integrating image processing with user customization and interactivity. Users can upload their own images, converting them into pixelated dot matrix images, and further customize each pixel with substitute graphics. This allows users to imbue the image with greater individuality and artistic expression while providing the freedom to adjust the effects based on personal preferences.

The goal is to break free from the limitations of traditional image editing tools, giving users more creative freedom. Through built-in dynamic effects, such as amplitude, pixel size, shape, density, and other parameters, users can not only adjust the appearance of the image but also control its dynamic evolution, exploring different visual rhythms and variations. This design, based on algorithms and user interaction, not only produces novel artistic effects but also allows each user to enjoy the fun of interacting with the image during the creative process.

The author also pays particular attention to a pain point in traditional image processing workflows: the inability to preview filter effects in real time. In traditional image software like Adobe Photoshop, users typically have to wait for the filter to finish processing before seeing the final result. This "what you see is not what you get" interaction model severely impacts creation efficiency and the overall editing experience. To address this, the author envisions introducing a real-time preview mechanism in the software system. After uploading an image, any changes—whether they involve substituting materials, switching geometric shapes, or adjusting parameters like amplitude or density—will be immediately reflected on the canvas, greatly enhancing interactivity and intuitiveness during the creative process. This design not only lowers the usage threshold but also allows users to experiment and adjust image styles efficiently, achieving more precise artistic control.

© By the author(s); licensee Mason Publish Group (MPG), this work for open access publication is under the Creative Commons Attribution International License (CC BY 4.0). (http://creativecommons.org/licenses/by/4.0/)

Furthermore, considering that traditional image editing software like Adobe Photoshop typically requires users to download and install the program, resulting in a high usage barrier, the author proposes developing the software as a fully HTML5- and JavaScript-based front-end image art generation tool. As a web application, it requires no installation, and users can simply open a webpage to start using it. All operations are executed locally in the browser, with no need for backend support. This ensures fast loading times and enables cross-platform compatibility with Windows, Mac, Linux, and mobile devices, greatly enhancing the convenience and accessibility of the creative process.

To bring this design concept and interactive experience to life in the software system, the author has carefully planned the implementation, covering aspects such as interface design, image processing workflows, core technology selection, effect realization, and material texture replacement technology.

1.1. Interface Design

The author first conceptualized the overall layout of the software's interface. The interface adopts a split-screen structure, with the control panel on the left and the image display canvas on the right. The control panel, arranged from top to bottom, includes the following elements: a "Upload Image" button, an "Upload Material Replacement Image" button, and various parameter adjustment sliders, covering amplitude, shape type (such as rectangle, circle, square), shape size, density, jitter, line style, and more. This sequence of elements is designed to guide users through the operation flow of "Upload → Replace Material → Adjust Parameters → Export," which enhances the intuitiveness and coherence of the user experience.

A zoom control slider is placed at the bottom-right corner of the canvas, allowing users to flexibly adjust the canvas proportion according to their needs, achieving both an overall structure preview and the ability to zoom in on pixel-level details. Given the rich detail of images after the pixelation process, this design significantly enhances user flexibility and precise control during the operation. At the bottom of the control panel, there is an "Export Image" button for users to easily save their final creations. Overall, the interface design emphasizes a clear and straightforward operational path to enhance system usability and interactivity.

1.2. Image Processing Workflow and Technology Selection

After completing the interface design, the author further planned the complete image processing workflow, covering key steps such as image upload and loading, pixel sampling, halftone decomposition, graphic dot replacement, dynamic bouncing control, and animation rendering and export. To achieve efficient and user-friendly functionality, the system is developed using front-end technologies. The page structure is built with HTML5, the layout and styling are handled by CSS, and image-level processing and dynamic effect rendering are implemented using JavaScript and the HTML5 Canvas API.

HTML is responsible for constructing the basic page structure, CSS handles the visual style and layout beautification, and JavaScript implements the core functionality and interaction logic. These three technologies work together to form a complete and efficient front-end framework, providing users with a stable, responsive, and feature-rich image processing experience.

1.3. Pixelation and Dynamic Effects Implementation

The core function of the software design is to automatically pixelate the uploaded image and activate dynamic bouncing effects. In terms of implementation, the software uses a "skip sampling" strategy, where it samples every certain number of pixels (the default interval is 2 pixels, controlled by the skip variable). This approach avoids pixel-by-pixel processing of the entire image, significantly reducing the computational load and improving performance. For example, for a 10×10 pixel image, the program samples pixels from columns and rows 1, 3, 5, 7, and 9, skipping the rest, thereby creating a sparse, regular sampling dot matrix.

After sampling, the system evaluates pixel color based on brightness and transparency. Only pixels that are darker or have higher transparency are converted into bouncing geometric shapes for rendering, while

© By the author(s); licensee Mason Publish Group (MPG), this work for open access publication is under the Creative Commons Attribution International License (CC BY 4.0). (http://creativecommons.org/licenses/by/4.0/)

brighter and opaque areas are ignored to avoid visual clutter and emphasize the focal points of the image. The density and size of the shapes are adjustable by the user through sliders, with the default setting placing one shape every 10 pixels. The size, shape (circle, rectangle, etc.), and position of each shape are determined by the sampling results.

To create a natural and rhythmic dynamic visual effect, the system assigns different bouncing parameters (amplitude, frequency, phase) to each shape, making their movements inconsistent and giving the overall scene a breathing-like dynamic. This halftone bouncing mechanism is implemented using JavaScript and the Canvas API, with requestAnimationFrame used to continuously refresh the canvas and dynamically update the position or scale of the shapes. Users can adjust all parameters in real-time via the sliders on the left panel, catering to their personalized needs.

1.4. Material Texture Replacement Technology Design

Building upon the basic pixel dot matrix processing, the author further introduces the material texture replacement feature to enhance the image's expressiveness and visual depth. Using the Canvas API to retrieve image pixel data (via getImageData), the system selects dark pixels based on their brightness levels, and replaces them with user-uploaded texture tiles at these key positions.

Additionally, users can adjust the size, position, bouncing amplitude, and density of the texture tiles using sliders. The system employs sine and cosine functions to apply subtle bouncing offsets and scaling variations to the texture tiles, creating dynamic effects that are both lively and rhythmic. This technology not only enhances the visual quality and detail of the dot matrix image but also significantly improves the overall artistic expression and dynamic aesthetics, offering users a more expansive creative space.

2. TECHNICAL SOLUTION

Based on the detailed design concepts for the system interface layout, image processing workflow, and functional modules, this section further elaborates on the software's specific development and implementation. To efficiently execute image sampling, halftone reconstruction, and dynamic animation rendering on the browser side, the software is built entirely with a front-end architecture. The following will explain the technical implementation, presenting the software's architecture design and implementation path through a system module function comparison table, an image processing flowchart, and a front-end technology stack diagram.

This software is developed using front-end technologies and employs HTML5, CSS3, and JavaScript for image pixel sampling, graphic reconstruction, and dynamic effect presentation. Core image processing relies on the Canvas 2D rendering engine, which uses the getImageData() method to extract the image's pixel matrix. Based on user-defined geometric shapes (rectangles, circles, squares) or custom texture images, the system performs replacement drawing to create dynamic halftone effects.

The software's processing workflow includes modules for image loading, pixel sampling and processing, dynamic visualization, and user interaction control. Users can adjust parameters such as amplitude, shape ratio, and density to generate and modify image effects in real time. The dynamic effects are updated through the requestAnimationFrame() loop, ensuring smooth image rendering.

In terms of system design, users first upload the original image and optional texture images. The image is then loaded and sampled into pixel data. The software system generates a dynamic representation of the image through reconstruction and halftone replacement. During each frame render, the software calculates the dynamic generation of geometric shapes or texture patterns based on user-set parameters (such as amplitude, frequency, shape ratio) and the pixel data.

The software system also supports exporting the current image as a JPEG file and recording the dynamic effects as an MP4 video. All image processing is completed entirely on the client-side within the browser, avoiding complex server-side calculations and enhancing the user interaction experience. This software is

© By the author(s); licensee Mason Publish Group (MPG), this work for open access publication is under the Creative Commons Attribution International License (CC BY 4.0). (http://creativecommons.org/licenses/by/4.0/)

suitable for interactive art creation, image style experiments, and teaching demonstrations, offering high real-time responsiveness and interactivity.

2.1 System Module Function Comparison Table

The system module function comparison table details the core modules of the entire application and their responsibilities, including the user interface control module, image import module, image processing module, animation rendering module, and image and video export module. Each module is assigned a specific function: the user interface module handles input control and parameter adjustments; the image import module facilitates local file loading and preprocessing; the image processing module performs pixel sampling and dynamic halftone generation; the animation rendering module is responsible for real-time drawing and frame animation updates; and the export module supports saving static images and video formats. The modules collaborate through data transfer and event-driven mechanisms, achieving high cohesion and low coupling, ensuring the system's flexibility and maintainability. (Figure 1)

Module Name	Function Description	Key Code Location or Critical Functions
User Interface Control Module	Manages various UI controls (sliders, buttons, file inputs) and their event bindings	`.controls` section in HTML, event listeners like `addEventListener`
Image Import Module	Handles the import of raw images and texture materials, loading them onto the Canvas	Functions `handleImage()` and `handleMaterial()`
Image Processing Module	Reads pixel data of the image, generates pixelated graphics based on parameters, dynamically calculates shape and position	Pixel sampling and drawing logic in the `draw()` function based on `imageData`
Animation Rendering Module	Implements time-frame based dynamic updates, integrating jitter, amplitude, and shape changes	`frame` variable controlling dynamic effects inside the `draw()` function
Image and Video Export Module	Supports saving the current canvas as an image and recording/exporting dynamic videos	Functions `saveImage()` and `exportVideo()`
Zoom Control Module	Allows users to dynamically adjust the zoom ratio of the Canvas display	Event listeners on `zoomSlider`, modifying `canvas.style.transform`

Figure 1: System Module Function Mapping Table

2.2 Image Processing Flowchart (Image → Halftone → Replacement → Animation)

The image processing flowchart clearly demonstrates the complete process from image input to dynamic presentation. The flow begins with the user uploading the original image and optional material images. After pixel data sampling, the image is divided into halftone units. Based on user-defined parameters (such as amplitude, shape ratio, density, jitter, etc.), the software system calculates the geometric form and dynamic offset for each halftone, and, combined with color information, determines the drawing style for each shape. The animation is achieved by updating the "frame" variable on a per-frame basis to create periodic dynamic changes. The system also supports the mixed drawing of lines and shapes to present rich visual effects. This flow ensures the real-time and interactive nature of image processing, effectively supporting immediate feedback based on user-defined parameters. (Figure 2)

© By the author(s); licensee Mason Publish Group (MPG), this work for open access publication is under the Creative Commons Attribution International License (CC BY 4.0). (http://creativecommons.org/licenses/by/4.0/)

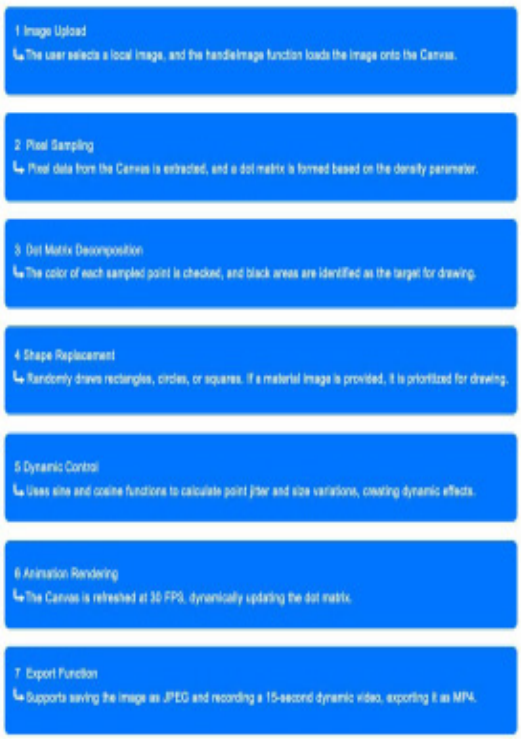


Figure 2: Image Processing Flowchart

2.3 Front-End Technology Stack Selection

The front-end technology stack selection diagram illustrates the technical path used by the system to implement complex image processing and animation rendering within a browser environment. The core technology is based on the HTML5 Canvas 2D API for graphic rendering, the FileReader API for local file reading, and the MediaRecorder API for recording dynamically generated video from the canvas. DOM manipulation handles user interaction control. The system adopts a pure front-end architecture, requiring no back-end support, which enhances the application's cross-platform compatibility and responsiveness. The technology selection focuses on standard web technologies and lightweight implementation, leaving room for future performance optimizations (such as WebGL acceleration or WebAssembly integration), ensuring the system has strong potential for technical evolution. (Figure 3)

Technology/API	Function Description	Usage Scenario
HTML5 Canvas 2D API	Responsible for all graphics drawing and dynamic rendering	Drawing operations on the `canvas` element and its `ctx` context
FileReader API	Reads user-uploaded image files and converts them into usable Image objects	Used in `handleImage()` and `handleMaterial()` to read local files
Image Object	Loads and caches image resources for drawing and processing	Asynchronously loaded and passed to the Canvas for drawing
MediaRecorder API	Records dynamic streams from the Canvas and exports video files	Used in the `exportVideo()` function to record dynamic animation videos
DOM Event Listener	Handles user interactions such as parameter slider adjustments, button clicks, and file uploads	Event binding on various controls using `addEventListener`
CSS3 Transform	Implements Canvas element zoom and enhances user interface interaction	Zoom control `zoomSlider` modifies `transform` for zooming effects

Figure 3: Frontend Technology Development and Technical Selection

© By the author(s); licensee Mason Publish Group (MPG), this work for open access publication is under the Creative Commons Attribution International License (CC BY 4.0). (http://creativecommons.org/licenses/by/4.0/)

3. SYSTEM IMPLEMENTATION AND INTERFACE DESIGN

This software system is built within a browser front-end environment, with no backend dependencies. It implements image pixel analysis, visual reconstruction, and dynamic rendering functionalities, primarily written using HTML5, CSS3, and native JavaScript, with the Canvas API serving as the core for graphic processing.

3.1 System Architecture

The architecture diagram illustrates how the front-end modules of the software system work together. The diagram clearly defines the relationships between different modules (such as image loading, pixel processing, dynamic rendering, and user interaction) and shows how the Canvas rendering engine generates the final visual effects through dynamic drawing and animation refresh. The responsibilities of each module and how they are interconnected help build the complete image processing workflow. (Figure 4)

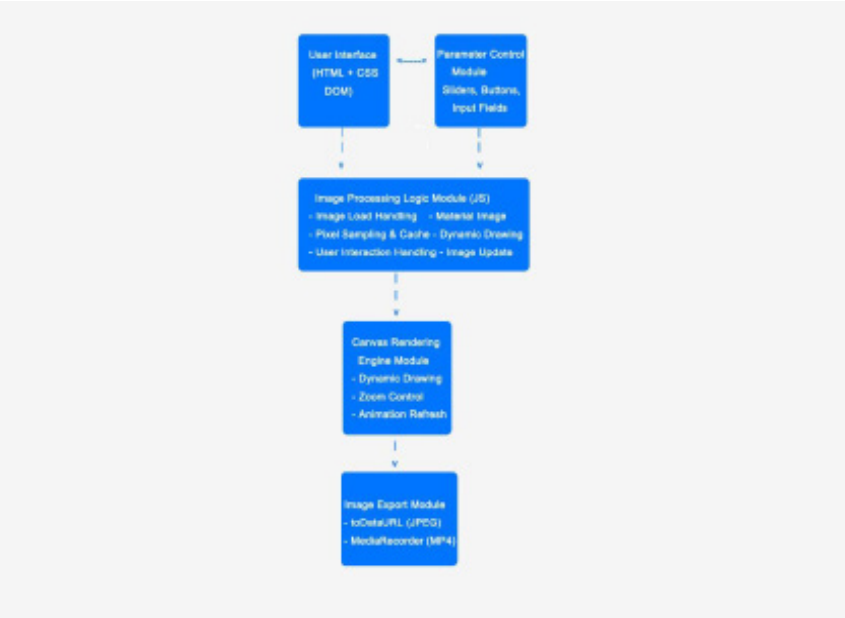


Figure 4: System Architecture Diagram

3.2 Interface Design

The software interface adopts a left-right split layout, with the left side serving as the control panel and the right side as the image display canvas. The control panel area is organized from top to bottom, starting with the “Import Image” and “Import Material Replacement Image” buttons, allowing users to import materials in sequence. Below these buttons are several parameter adjustment controls, including amplitude, shape type, shape size, density, jitter, and line style. Users can adjust the halftone effect and dynamic behavior flexibly using sliders and dropdown menus.

The right-side canvas area serves as the main space for image display and real-time preview, allowing users to visually check the halftone image effect based on the current settings. A zoom control is located at the bottom-right corner of the canvas, making it easy for users to zoom in on details or view the overall effect, enhancing the flexibility and accuracy of operations. At the bottom of the interface is the “Export Image” button, allowing users to save their creative results.

After development, the software interface is clean and intuitive, with a well-organized module layout and a clear, smooth operation flow, ensuring ease of use and an optimal interactive experience. Users can easily import images and texture images, adjust parameters such as amplitude, shape size, and density from the left control panel, and all changes will be reflected in real-time on the right canvas. The canvas supports zooming

© By the author(s); licensee Mason Publish Group (MPG), this work for open access publication is under the Creative Commons Attribution International License (CC BY 4.0). (http://creativecommons.org/licenses/by/4.0/)

and dynamic preview, allowing users to closely observe every detail of the halftone image and its dynamic changes. Overall, the interface design emphasizes user experience, balancing aesthetics and practicality, and provides creators with an efficient and inspiring digital art platform.

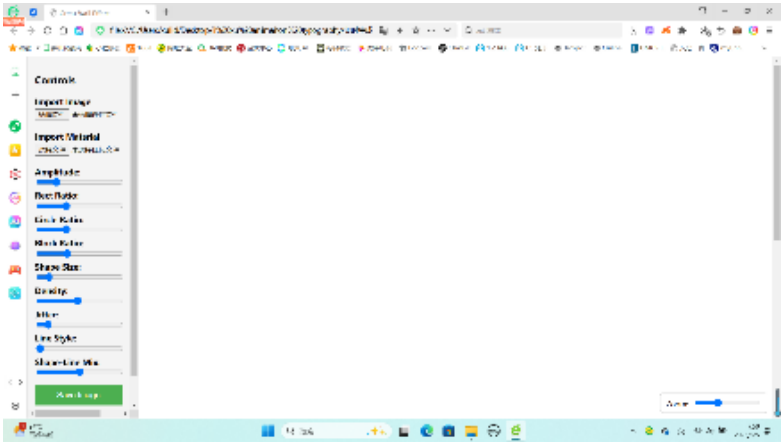


Figure 5: Interface Display Diagram

4. RUNNING TEST EXPERIMENTS AND VISUAL EFFECT GENERATION EXPERIMENTS

4.1 Software Running Test Experiment

After the software development is completed, it undergoes running tests to ensure its stability and reliability. Below is a textual table of the running test experiment for the image processing software, which primarily describes the software's performance in different environments, including functionality verification, performance testing, boundary condition testing, and exception handling. The goal is to comprehensively assess whether the software meets the expected functional requirements and provide a detailed analysis of its performance in real-world usage.

Based on the results of the test experiments, the image processing software performed well in all functionality, performance, and stability tests, meeting the expected requirements. It runs stably and adapts to various environments, demonstrating the software's practicality and reliability. (Figure 6)

Experiment No.	Experiment Item	Experiment Steps	Expected Result	Actual Result
1	Image Import and Display	Import an image and check if it displays on the canvas	The image is successfully imported and displayed	The image was successfully imported and displayed
2	Texture Replacement Import	Import a texture replacement image and check the effect	The replacement texture is successfully applied	The texture was applied correctly, and the image updated
3	Parameter Adjustment Test	Adjust parameters like amplitude, shape, and density, and check the image changes	Changes in parameters are reflected in real-time	The adjustments worked as expected, with real-time image changes
4	Dots Effect Test	Activate the dot effect and check how the image changes	The dot effect is successfully generated and shown	The dot effect was generated correctly, and the image matched the settings
5	Dynamic Effect and Jitter Test	Adjust dynamic effects and jitter, then check the image's response	The dynamic and jitter effects are applied smoothly	The dynamic and jitter effects responded well
6	Zoom Control Test	Use the zoom control to zoom in or out and observe the effect	The image can be zoomed in/out smoothly, with clear details	Zoom operation was smooth, and the image was clear
7	Image Export Function Test	Click "Export Image" and save the image to a specified path	The image is successfully exported, with correct format and location	The image was exported successfully, with the right format
8	Performance Test	Test processing time and performance for images of different sizes	The software processes different sizes efficiently	Fast processing with no lag or delays
9	System Stability Test	Run the software for an extended period, performing regular tasks	The software runs continuously without crashes or lag	The software ran smoothly and stably
10	Multi-Platform Compatibility Test	Run the software on different operating systems and test functionality compatibility	The software works normally on different platforms	The software runs smoothly across platforms with consistent functionality

Figure 6: Software Operation Testing Experiment Table

© By the author(s); licensee Mason Publish Group (MPG), this work for open access publication is under the Creative Commons Attribution International License (CC BY 4.0). (http://creativecommons.org/licenses/by/4.0/)

4.2 Visual Effect Generation Experiment

The author demonstrates the functionality and the rich variety of visual effects that this software system can generate through specific application scenarios, further showcasing its potential in art experimentation and visual expression. To present the software's functionality and artistic capabilities more intuitively, the following example illustrates how dynamic halftone and texture replacement techniques can transform a simple letter graphic into a completely new visual experience. The experiment focuses on the letter "A" and emphasizes the innovative effects of dynamic halftoning and texture replacement on visual expression. The following is a detailed operational process of the visual effect generation experiment:

4.2.1 Initial State: Static Letter "A"

Background color: Black

Initially, the letter "A" is in a static state with a black background. The letter itself is composed of a solid black area without any pixelated dots, giving it a complete, traditional letter appearance. (Figure 7)

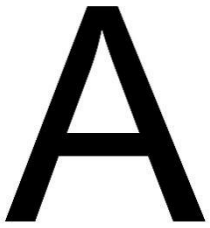


Figure 7: Letter A

4.2.2 Convert to Halftone Effect

After clicking the "Import Image" button at the top of the control panel on the left side and importing the letter "A" image, the right image display area shows the halftone effect of the letter "A" being generated. As the dynamic effect unfolds, the letter "A" gradually appears by displaying small particle points (pixels or particles) one by one. Each particle can be a small dot or a more creative shape, and they form the outline of the letter "A." These points can gradually appear, creating a dynamic effect of the letter being constructed over time. (Figure 8)

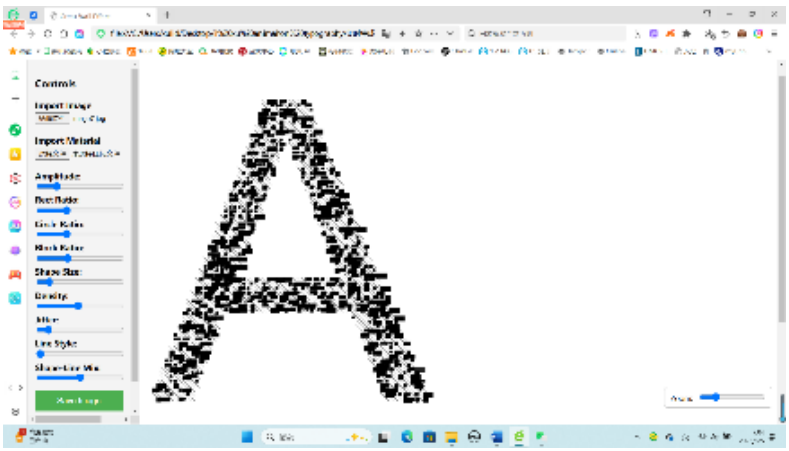


Figure 8: Dotted Grid Effect

© By the author(s); licensee Mason Publish Group (MPG), this work for open access publication is under the Creative Commons Attribution International License (CC BY 4.0). (http://creativecommons.org/licenses/by/4.0/)

4.2.3 Rich Visual Expression of the Halftone Image

To further enhance the visual effect, users can adjust all parameters in real-time through the sliders on the left panel. By assigning different bouncing parameters (amplitude, frequency, phase), they can create inconsistencies in the bouncing motion, which also leads to variations in the letter "A" effect. This allows for the display of diverse shapes or textures, generating unique and expressive visual effects that meet personalized needs. (Figures 9-11)

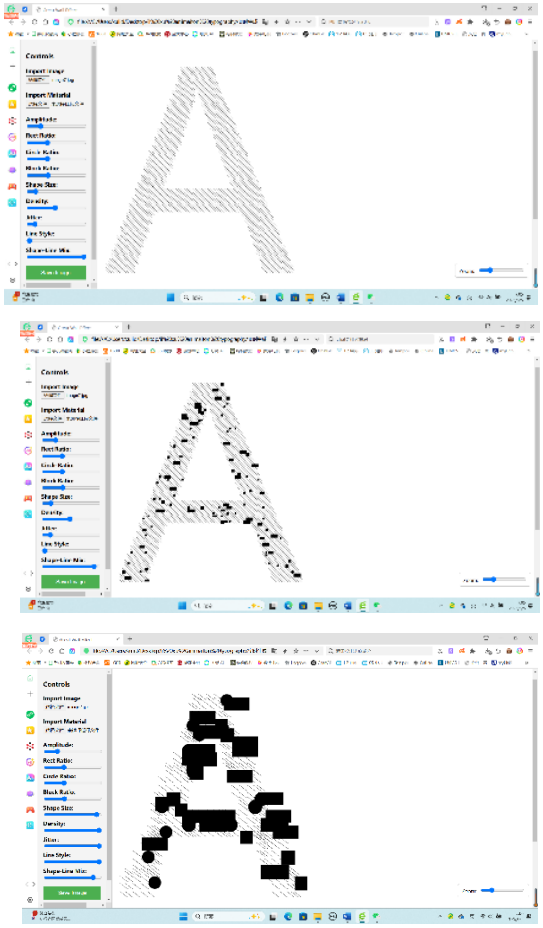


Figure 9-11: Diversified Texture Effects

4.2.4 Dynamic Texture Replacement

Loading the Texture Image:

Once the letter "A" is displayed in a halftone pattern, users can choose an image as a texture to replace these particle points. The system will automatically recognize the pixel information in the selected texture image and replace each point in the halftone grid with a corresponding pixel or texture from the chosen material.

Dynamic Texture Replacement in the Halftone Grid:

After clicking the "Import Texture Image" button at the top of the control panel on the left side, the original black halftone letter "A" will transform into an image composed of a new texture—such as a portrait image. (Figure 12) The entire letter "A" will be redrawn with the new texture, creating a fresh visual effect. This process can either be a smooth transition or a dynamic process where each point is gradually replaced.

© By the author(s); licensee Mason Publish Group (MPG), this work for open access publication is under the Creative Commons Attribution International License (CC BY 4.0). (http://creativecommons.org/licenses/by/4.0/)

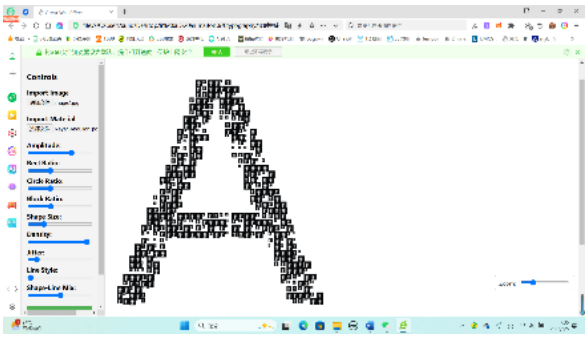


Figure 12: Portrait Replacement Effect

4.2.5 Interactivity of the Texture Image

If the user uploads different texture images, the appearance of the letter "A" can change instantly, producing entirely different visual effects. For example, replacing the texture with an image featuring floral patterns (Figure 13) or animal motifs like a cat (Figure 14) can create more complex visual layers and styles.

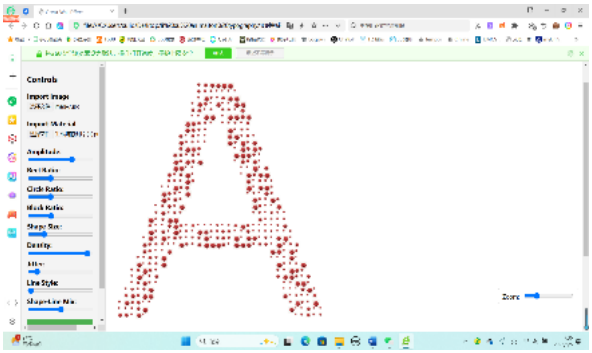


Figure 13: Floral Replacement Effect

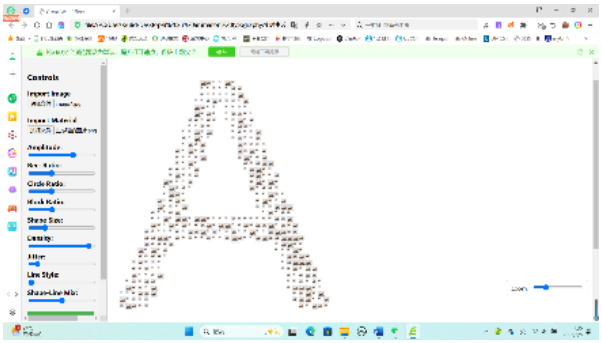


Figure 14: Animal Cat Replacement Effect

© By the author(s); licensee Mason Publish Group (MPG), this work for open access publication is under the Creative Commons Attribution International License (CC BY 4.0). (http://creativecommons.org/licenses/by/4.0/)

4.2.6 Save and Export

When the user clicks the save button on the left control panel, a dialog box will pop up, allowing them to enter a file name and specify the download location. After clicking the download button, the image will be saved in JPG format. (Figure 15)

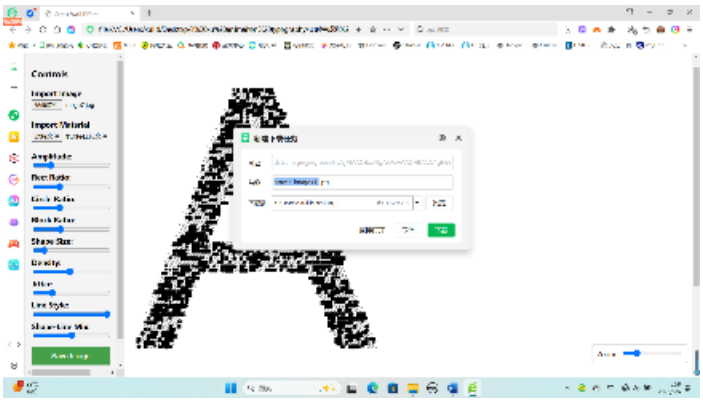


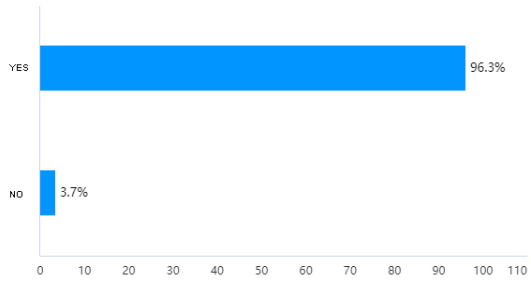
Figure 15: Save and Export

This experiment comprehensively validated the technical implementation of the system by demonstrating the gradual transition from a static letter “A” to dynamic halftone effects, real-time parameter adjustments, dynamic texture replacements, and the interactive display of texture images. It showcased the innovative expression of digital halftone images in visual arts. This process breaks through the limitations of traditional image processing, granting creators greater freedom and flexibility, and expanding the possibilities for digital art creation.

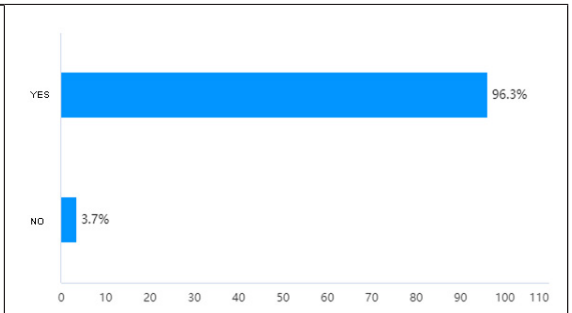
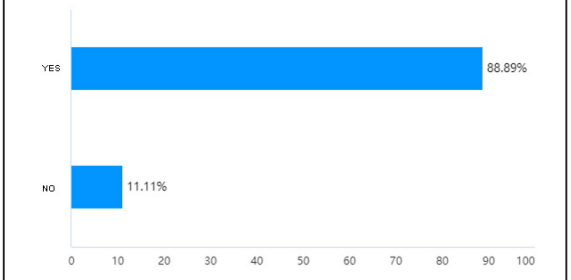
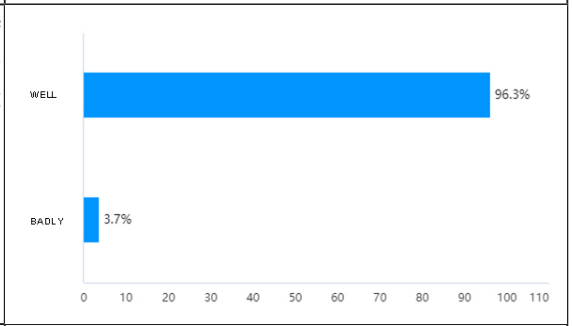
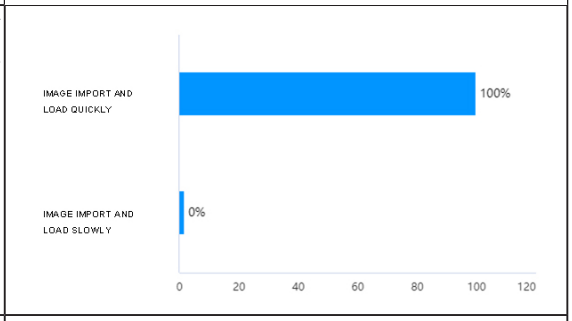
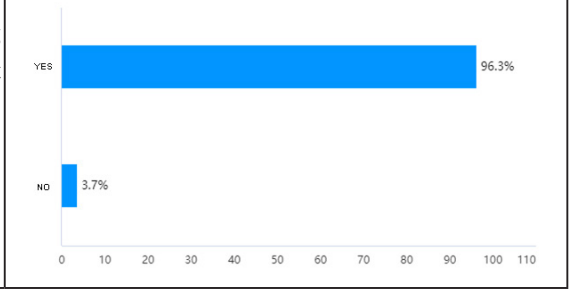
5. SURVEY AND SYSTEM PERFORMANCE EVALUATION

To further validate the system’s operation, I designed and conducted a survey to gather feedback from participants regarding the system’s performance in generating dynamic halftone effects, texture replacement functionality, and user experience aspects.

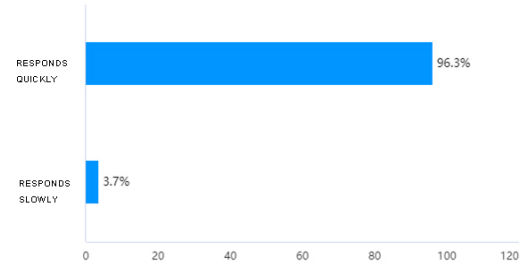
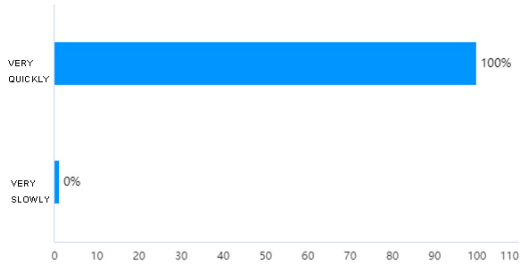
The purpose of this survey is to assess the performance and user experience of the image processing design software system, based on dynamic halftone and texture replacement technologies. The survey covers dimensions such as halftone dynamic effects, texture replacement, image visual effects comparison, performance testing, and overall user experience.

1	After importing the letter "A" image, did it transform into a pixelated effect?	

© By the author(s); licensee Mason Publish Group (MPG), this work for open access publication is under the Creative Commons Attribution International License (CC BY 4.0). (http://creativecommons.org/licenses/by/4.0/)

2	After importing the letter "A" image, did it start moving?	
3	After importing another material image, was the pixelated effect on the screen replaced by the corresponding material image?	
4	How is the overall experience of the process from importing the image, processing it, achieving the effect, to saving the image?	
5	How is the speed of the image display on the right side after clicking the "Import Image" button?	
6	After importing the image and sliding the left slider, does the image on the right side change?	

© By the author(s); licensee Mason Publish Group (MPG), this work for open access publication is under the Creative Commons Attribution International License (CC BY 4.0). (http://creativecommons.org/licenses/by/4.0/)

7	After importing the image and sliding the left slider, how is the visual effect change and the speed of the display on the right side?	
8	How many seconds does it take to save the image after clicking the export/save button?	

Through the analysis of the data, the performance of the image processing design software was evaluated across multiple dimensions. The survey results show:

In terms of basic image processing functionality, the survey indicated that the system's image import speed and save/export functions were highly praised by users. 100% of participants reported that image import was very fast, and the export/save operations were also very smooth, with short response times and almost no delays. This demonstrates that the system performs excellently and responds efficiently when handling large-scale image data.

In generating dynamic halftone effects, the vast majority of users (96.3%) felt that the system responded quickly during user interactions, and the dynamic effects on the screen were clearly noticeable. When adjusting the left-side slider, the right-side screen responded quickly and displayed significant visual effects. This highlights the system's real-time performance and smoothness in dynamic halftone processing. Although a small number of users (approximately 3.7%) reported slower responses in some cases, this did not affect the overall experience, and the efficiency of the system was acknowledged by most users.

Regarding the texture replacement function, the majority of participants (88.9%) indicated that the feature worked smoothly and quickly replaced the texture images, further confirming the system's high stability and excellent interactivity.

In terms of user experience, the overall feedback was very positive. Participants generally found the system's interface user-friendly, the operation simple, highly interactive, and capable of quickly responding to user needs, providing immediate feedback. This smooth interaction experience allowed users to efficiently complete image design tasks.

In summary, the survey confirmed the software's excellence in dynamic halftone effect generation, texture replacement technology, performance, and user experience. Most participants gave high ratings to the software's performance and interactive features, indicating that the image processing design software excels in meeting user needs, improving work efficiency, and providing a smooth operation experience.

6.Conclusion and Outlook

This study focuses on image halftoning and texture replacement techniques, designing and implementing a dynamic image processing tool based on a web platform. The system adopts a purely frontend architecture, leveraging HTML5, CSS3, and JavaScript technologies, combined with the Canvas API to achieve pixel-level

© By the author(s); licensee Mason Publish Group (MPG), this work for open access publication is under the Creative Commons Attribution International License (CC BY 4.0). (http://creativecommons.org/licenses/by/4.0/)

image sampling, graphic reconstruction, and dynamic animation rendering. Users can interact with the graphical interface to customize halftone styles and substitute textures, adjusting animation parameters in real time to achieve dynamic visual reconstruction and artistic stylization of images. Experimental results show that the system offers advantages such as ease of use, efficient responsiveness, and rich visual effects, demonstrating broad application potential in fields like generative art, interactive design, and digital media creation.

In future development, the software will continue to be upgraded functionally, supporting the extension of various shapes and animation effects. Additionally, by incorporating features like the fusion and layering of multiple texture maps, and integrating deep learning techniques, the system aims to push image processing towards more intelligent, personalized, and artistically layered directions.

References:

Xu, L. (2022). Hidden graphics. Wuhan University of Technology Press.

Adobe. (2022). Create timeline animations in Adobe Photoshop. https://helpx.adobe.com/ae_en/photoshop/using/creating-timeline-animations.html

Adobe. (2023). Halftone dots pattern in Photoshop. https://www.adobe.com/products/photoshop/halftone-effects.html

Radke, R. J. (2013). Computer vision for visual effects. Cambridge University Press.

Sýkorová, L. (2021). Drawing as a non-verbal communication tool. PUNO Press.

© By the author(s); licensee Mason Publish Group (MPG), this work for open access publication is under the Creative Commons Attribution International License (CC BY 4.0). (http://creativecommons.org/licenses/by/4.0/)