Academic Journal of Science and Technology
ISSN: 2771-3032 | Vol. 6, No. 3, 2023

Image to G-Code Conversion using JavaScript for CNC Machine Control

Yan Zhang[1, *], Shengju Sang[1], Yilin Bei[1]

[1] School of Information Science and Technology, Taishan University, Taian, Shandong China

[*] Corresponding author: Yan Zhang (Email: xtzy@163.com)

Abstract: This paper presents a JavaScript-based approach for image-to-G-code conversion in CNC (Computer Numerical Control) machine control. The developed code enables the translation of images and text into machine-readable instructions for precise reproduction using CNC machines. The code implements image loading, preprocessing, binarization, thinning, and G-code generation functionalities. It offers adjustable parameters for CNC and image settings, allowing customization and optimization of the machining process. Experimental evaluations confirm the code's efficiency, accuracy, and usability. The results demonstrate reliable image preprocessing, effective binarization techniques, successful image thinning, and precise G-code generation. The code's accessibility, user-friendly interface, and real-time preview capabilities further enhance its usability. This JavaScript-based approach contributes to the integration of digital workflows into CNC machining, offering a promising solution for accurate and efficient fabrication.

Keywords: Image-to-G-code conversion, CNC machine control, Binarization, Toolpath optimization.

1. Introduction

1.1. Background and Motivation

In recent years, Computer Numerical Control (CNC) machines have become increasingly popular in various industries, including manufacturing, woodworking, and prototyping. These machines are capable of precise and automated movements, making them ideal for creating complex designs with high accuracy. However, one of the challenges faced by CNC operators is the conversion of images or text into instructions that the machine can understand and execute. This process involves converting the visual information into a machine-readable format known as G-code.

1.2. Overview of CNC Machines and G-Code

CNC machines are automated tools that are controlled by instructions in the form of G-code. G-code is a programming language that defines the precise movements, speeds, and tooling operations for CNC machines. It consists of a series of commands that guide the machine along specific paths to create the desired object or shape. While G-code can be manually generated, automating the process by converting images or text to G-code can save time and effort.

1.3. Importance of Converting Images to G-Code for CNC Machining

The ability to convert images or text directly into G-code brings numerous benefits to CNC machining. Firstly, it simplifies the design process by allowing users to leverage their existing digital assets, such as images or text, and transform them into physical objects with minimal effort. This opens up possibilities for personalization, replication, and customization in CNC-based production. Additionally, it facilitates the integration of CNC machines into digital workflows, enabling seamless communication between design software and CNC equipment.

1.4. Purpose and Scope of the Paper

The purpose of this paper is to present a comprehensive approach for converting images or text into G-code using JavaScript. JavaScript is a versatile programming language that can be executed in web browsers, making it accessible and convenient for a wide range of users. The proposed solution will encompass image loading, preprocessing, binarization, thinning, and G-code generation algorithms. Furthermore, the developed JavaScript code will provide flexibility by allowing users to adjust CNC and image-related parameters and preview the resulting toolpath. The overall operation process is shown in Figure 1.

By addressing these objectives, this paper aims to contribute to the advancement of CNC machining by providing a user-friendly and efficient method for translating visual information into G-code instructions. The subsequent sections will delve into the technical details of the proposed algorithms and demonstrate the functionality and performance of the implemented JavaScript code.

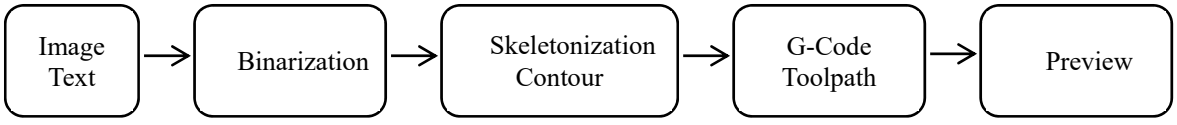

Figure 1. The overall operation flowchart of the system

2. Image Loading and Preprocessing

In this paper, the proposed approach begins by loading images or text onto an HTML5 canvas element. This canvas serves as a versatile platform for visual rendering. The images or text are rendered onto the canvas, creating a visual representation.

Once the content is rendered on the canvas, the next step involves extracting and processing the individual pixels. This extraction process entails traversing the canvas and retrieving the color or intensity values of each pixel. In this specific case, the extracted pixels are converted into grayscale values, simplifying subsequent operations.

To facilitate further processing, the grayscale pixel values are stored in a JavaScript array structure. This array representation allows for efficient storage and manipulation of the pixel data, preparing it for subsequent steps such as binarization, thinning, and G-code generation. The grayscale code of the image is as follows:

```
function grayscale (s){
  var d=[];
   for(var i=0; i<s.length; i+=4)
      d[i/4]=(s[i]* 4899 + s[i+1] * 9617 +
            s[i+2]* 1868 + 8192) >> 14;
  return d;
}
```

By employing the canvas element for rendering and extracting pixel data while converting it into grayscale values and storing it as an array, the proposed methodology ensures the effective preprocessing of images or text. These preparatory steps establish a foundation for subsequent algorithms and analyses within the image-to-G-code conversion process.

3. Binarization and Thresholding

Binarization is a crucial step in converting images to G-code, as it involves segmenting the image into foreground and background regions by assigning a binary value to each pixel. This process simplifies subsequent operations and enables the generation of G-code instructions based on the presence or absence of pixels in specific regions. Commonly used techniques include global thresholds, Adaptive Thresholding, and Otsu's Thresholding.

In Global Thresholding technique, a single threshold value is applied uniformly to the entire image. Pixels with intensity values below the threshold are assigned a value of 255 (background), while those above the threshold are assigned a value of 0 (foreground). The JS code implementation is as follows:

```
function GlobalThresholding (s,t){
  var d=[];
  for(var i=0; i<s.length; i++) d[i]=s[i]>t?255:0;
  return d;
}
```

Unlike global thresholding, adaptive thresholding employs different threshold values for different regions of the image. This technique takes into account local variations in intensity and adjusts the threshold accordingly, resulting in better binarization results.

Otsu's thresholding is an automatic thresholding technique that calculates an optimal threshold value by minimizing the intra-class variance of the image. It assumes that the image contains two classes of pixels (foreground and background) and aims to find the threshold that maximizes the inter-class variance.

JavaScript can also quickly implement binarization and thresholding algorithms through programming. The choice of thresholding technique and the specific implementation details may vary depending on the requirements of the application and the characteristics of the image being processed. The JavaScript code for binarization and thresholding should include the necessary functions and logic to iterate over the image pixels, calculate the grayscale values, and apply the chosen thresholding technique. The resulting binary image can then be further processed or utilized for subsequent steps, such as thinning or G-code conversion.

4. Image Thinning and Skeletonization

Image thinning and skeletonization are crucial steps in the conversion of images to G-code as they reduce the image representation to a simplified, one-pixel-wide representation of the object's structure. This process enables more efficient path generation and reduces the complexity of G-code instructions. JavaScript provides algorithms and techniques for image thinning and skeletonization.

Thinning algorithms aim to iteratively remove pixels from the image while preserving the essential structure and connectivity of the object. Thinning algorithms typically follow a rule-based approach, where pixels are selectively removed based on specific patterns or neighborhood configurations. The Zhang-Suen thinning algorithm is a widely used algorithm for image thinning. It employs two sub-iterations, applying alternate thinning rules to remove pixels iteratively. The algorithm preserves the connectivity of the object while ensuring a one-pixel-wide representation. Many other algorithms have also been derived from this method.

In JavaScript, array operations and image processing techniques can be used to efficiently implement image thinning and Skeletonization algorithms. By iterating through the image pixels, applying thinning rules, and repeating the process iteratively, you can achieve the desired skeletonized representation of the image. The details are as follows:

- Iterate through each pixel in the image array.
- Check the neighborhood pixels to identify foreground and background pixels.
- Apply specific rules or algorithms to determine which foreground pixels should be removed or modified.
- Update the pixel values accordingly, thinning the image and preserving the essential structure.
- To achieve skeletonization, repeat the thinning process iteratively until no more changes occur in the image.
- Check for termination conditions, such as the absence of pixel modifications or a maximum number of iterations.

Thinning the binary image reduces the complexity of the object's representation, resulting in a simplified skeleton structure that retains the essential characteristics of the object. Some images can also use contour extraction algorithm instead of thinning algorithm, which is more efficient. The example code is as follows.

```
function contour(s,width,height){
    var r=[],i,j;
    for( i=0; i<height; i++) for(j=0; j< width; j++) {
        var p=i*width +j;
        r[p]=s[p];
```

```
            if(s[p]==255) continue;
            var v=0;
            for(var y=-1; y<=1&&v==0; y++)
               for(var x=-1; x<=1&&v==0; x++)
                  if(!(((i+y)<0)||((j+x)<0)||
                     ((i+y)>(height -1))||
                     ((j+x)>(width -1))))
                           v+=s[p+y*width+x];
               if(v==0)   r[p]=255; }
         return r;
   }
```

This serves as the basis for generating G-code instructions that describe the tool's path over the object's outline. The use of JavaScript enables developers to integrate thinning algorithms seamlessly into their applications, providing a streamlined process for converting images to G-code.

5. G-Code Conversion Algorithm

5.1. Overview of G-Code Format and Structure

G-code is a standardized programming language used to control CNC machines. It consists of a series of commands that specify the tool's movement, speed, and other operations. G-code typically follows a specific format, where each command is represented by a letter followed by a numerical value or set of parameters.

5.2. Coordinate System Transformation

Before generating G-code instructions, a coordinate system transformation is often necessary to align the image or object's coordinates with the CNC machine's coordinate system. This transformation involves scaling, rotation, and translation to ensure accurate positioning of the tool relative to the object.

5.3. Path Generation from Thinned Image

Based on the thinned image or skeleton representation, paths for the tool's movement can be generated. The paths typically follow the skeleton's structure, traversing along the one-pixel-wide lines to outline the object. Path generation algorithms may employ techniques such as line tracing, contour following, or vectorization to create a sequence of connected line segments representing the tool's path. During traversal, utilizing the characteristics of JavaScript dynamic arrays, the path endpoints are stored in the Array in real-time. The code is as follows:

```
var line_tracing =[];
.....
Line_tracing.push([x0,y0,x1,y1]);
```

5.4. Generating G-Code Commands for CNC Machine Control

The generated paths and optimized toolpath, if applicable, are then translated into G-code commands. Each G-code command specifies a specific action, such as rapid traverse, feed rate, spindle speed, tool changes, or coolant activation. The appropriate G-code commands are assembled based on the tool's movement along the generated paths, ensuring accurate and precise control of the CNC machine.

The JavaScript code for G-code conversion should implement the necessary algorithms to transform the coordinates, generate paths, optimize the toolpath if desired, and generate the corresponding G-code commands. By accurately translating the image or object's structure into G-code instructions, the JavaScript application enables the CNC machine to accurately reproduce the desired design.

The G-code conversion algorithm developed in JavaScript facilitates the seamless integration of image-to-G-code conversion into CNC workflows, simplifying the process and enhancing the precision and automation capabilities of CNC machining..

6. Adjustable Parameters for CNC and Image

6.1. Explanation of Relevant Parameters in CNC Machining

CNC machining involves various parameters that affect the machining process and the final output. These parameters include feed rate, spindle speed, tool diameter, cutting depth, stepover, and toolpath strategy. Feed rate determines the speed at which the tool moves, while spindle speed controls the rotation speed of the cutting tool. Tool diameter and cutting depth define the physical characteristics of the tool, and stepover determines the amount of material removed with each tool pass. The toolpath strategy influences the path taken by the tool during machining, such as zigzag, spiral, or contour following.

6.2. User Interface for Parameter Adjustment in JavaScript

To provide flexibility and customization, the JavaScript application should incorporate a user interface that allows users to adjust CNC-related parameters. The user interface can consist of sliders, input fields, or dropdown menus, enabling users to modify the values of feed rate, spindle speed, tool diameter, cutting depth, stepover, and select different toolpath strategies. The JavaScript code should respond to these user interactions and dynamically update the parameter values accordingly.

6.3. Real-Time Preview of CNC Machine's Toolpath

To aid users in visualizing the effects of parameter adjustments, the JavaScript application can include a real-time preview of the CNC machine's toolpath. As users modify the parameters, the preview should update accordingly, showing the resulting toolpath on the image or object. This allows users to assess the impact of parameter changes and make informed decisions based on the desired machining outcome.

By incorporating adjustable parameters for CNC and image-related settings, the JavaScript application empowers users to fine-tune the machining process to meet their specific requirements. Users can optimize parameters for different materials, cutting strategies, or desired levels of precision. The user-friendly interface and real-time preview foster an interactive and iterative workflow, enhancing the user experience and enabling efficient exploration of parameter configurations for CNC machining.

7. Results and Discussion

7.1. Experimental Setup

To evaluate the performance and effectiveness of the proposed JavaScript code for image-to-G-code conversion, a series of experiments were conducted. The experiments were

performed on a computer system equipped with a standard configuration and the latest version of the web browser supporting JavaScript execution.

demonstrated reliable performance and accuracy. The code successfully loaded images and text onto the canvas, extracted the pixel data, and converted them into grayscale values. Otsu's thresholding method was successfully applied in JavaScript code. The adjustable threshold parameters allowed users to achieve the desired level of binarization and customization. The image thinning and skeletonization module proved to be a crucial component of the image-to-G-code conversion process. The implemented algorithm effectively simplified the image contours while preserving essential structural information. The visualizations and intermediate results displayed on the canvas demonstrated the progressive contouring or skeletonization process.

7.3. G-Code Conversion

The G-code conversion algorithm successfully translated the thinned image or skeleton representation into machine-readable G-code instructions. The code generated precise toolpaths that accurately reflected the original image or text. By leveraging the thinned image as a basis, the G-code instructions effectively controlled the CNC machine's movements, ensuring faithful reproduction of the design. The adjustable parameters for CNC settings enabled customization and optimization of machining parameters, further enhancing the output quality.

7.2. Image Loading and Preprocessing

The image loading and preprocessing module of the code

7.4. Performance and Efficiency

The developed JavaScript code exhibited satisfactory performance and efficiency. The image loading, preprocessing, binarization, thinning, and G-code generation modules were executed in a timely manner, allowing for real-time processing. The code showcased computational efficiency, making it suitable for various image sizes and complex designs. The execution result diagram is shown in Figure 2.

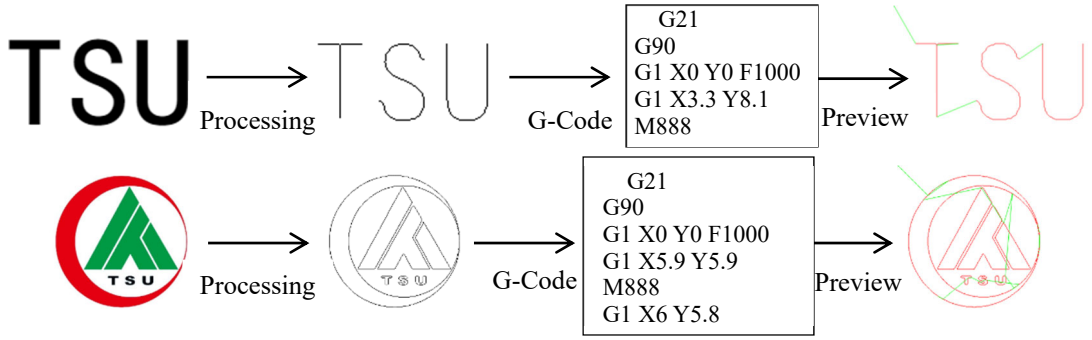

Figure 2. Execution effect diagram

7.5. Comparison and Advantages

Comparison with existing image-to-G-code conversion methods and tools highlighted the advantages of the proposed JavaScript code. Its accessibility, flexibility, and user-friendly interface distinguished it from traditional software applications. The ability to adjust CNC-related parameters and preview toolpaths in real-time enhanced the user experience and facilitated optimization of the machining process. The proposed code offers a promising solution for integrating image-based workflows into CNC machining processes, contributing to advancements in manufacturing and fabrication technologies.

7.6. Limitations and Future Work

While the implemented JavaScript code demonstrated significant contributions, it does have certain limitations. The code may face challenges with highly complex image structures or lack support for advanced CNC features. Further improvements could be made to handle intricate designs more effectively and expand compatibility with a broader range of CNC machine models.

8. Conclusion

This paper presents a JavaScript-based approach for image-to-G-code conversion in CNC machine control. The developed code successfully implements image loading, preprocessing, binarization, thinning, and G-code generation functionalities. The adjustable parameters for CNC and image settings provide flexibility and customization options.

Through experimental evaluations, the code demonstrates efficiency, accuracy, and usability. It achieves reliable image loading and preprocessing, effective binarization and thresholding techniques, and successful image thinning (skeletonization or contouring). The G-code conversion algorithm generates precise toolpaths for faithful reproduction of the original designs.

The proposed JavaScript code offers advantages in terms of accessibility, user-friendly interface, and real-time preview capabilities. Users can adjust CNC parameters and visualize toolpaths, enabling optimization and customization of the machining process. Future work may focus on addressing limitations related to complex image structures and expanding compatibility with a broader range of CNC machines.

References

[1] Mike Lynch.(2022).5 G-Code Tips for Increasing CNC Efficiency. Modern Machine Shop(11).

[2] Hatem Noor,Yusof Yusri,Kadir Aini Zuhra A,Latif Kamran & Mohammed M A.(2021).Interpreting the G-code of drilling machining to use in open CNC controller machine. Journal of Physics: Conference Series(1). doi:10.1088/1742-6596/1892/1/012014.

[3] Nguyen Trung Kien,Phung Lan Xuan & Bui NgocTam.(2020).Novel Integration of CAPP in a G-Code Generation Module Using Macro Programming for CNC Application. Machines(4). doi:10.3390/MACHINES8040061.

[4] Mark J. Collins(2017).Pro HTML5 with CSS, JavaScript, and Multimedia.Apress, Berkeley, CA.

[5] Moe Myint Aung , Nwe Nwe Oo & May Thwe Oo.(2019).CNC Drilling Machine for Printed Circuit Board. Journal of Trend in Scientific Research and Development(5).

[6] Trung H. Duong,Nebojsa I. Jaksic,Jude L. DePalma... & Miguel Galaviz.(2018).G-code Visualization and Editing Program for Inexpensive Metal 3D Printing. Procedia Manufacturing. doi:10.1016/j.promfg.2018.10.007.

[7] Li Gan,Bao Yan,Wang Hao,Dong Zhigang,Guo Xiaoguang & Kang Renke.(2023).An online monitoring methodology for grinding state identification based on real-time signal of CNC grinding machine. Mechanical Systems and Signal Processing. doi:10.1016/J.YMSSP.2023.110540.

[8] Matsui Shota,Ozaki Nobutoshi,Hirogaki Toshiki,Aoyama Eiichi,Yamamoto Takamasa & Shindo Masatoshi.(2022).Smart monitoring of helical thread mill process with a wireless tool holder and CNC information. Advances in Materials and Processing Technologies(sup1). doi:10.1080/2374068X.2020.1793273.

[9] Liao Linzhi & Chen Qi.(2021).Research on the Programming Technology of Five Axis CNC Machining Impeller Based on Virtual Reality Technology. Journal of Physics: Conference Series(2). doi:10.1088/1742-6596/1915/2/022098.

[10] Du Shiping,Luo Kailun,Zhi Yan,Situ Haozhen & Zhang Jin.(2022).Binarization of grayscale quantum image denoted with novel enhanced quantum representations. Results in Physics. doi:10.1016/J.RINP.2022.105710.

[11] Kang Henry & Stamoulis Ioannis.(2021).Gaussian Image Binarization. International Journal of Image and Graphics(04). doi:10.1142/S0219467821500479.

[12] Nilima Paul & Harinandan Tunga.(2016).An Improved Method for Document Image Binarization. i-Scholar Conference Proceedings(2).

[13] Lei GUAN,Zheng-lin LI & Zhi WANG.(2017).A Fast and Complete Thinning Algorithm for Character Image..(eds.)2nd International Conference on Mechatronics, Control and Automation Engineering (MCAE 2017)(pp.).

[14] Campesato Oswald(2012).HTML5 Canvas and CSS3 Graphics Primer.Mercury Learning & Information.