ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika

ISSN(p): 2338-8323 | ISSN(e): 2459-9638 | Vol. 11 | No. 4 | Halaman 892 - 906
DOI: http://dx.doi.org/10.26760/elkomika.v11i4.892 Oktober 2023

Image Processing-based Automatic Printed Circuit Board Printing Machine

NINDY PRATIWI ABUNG, BUDI HARSONO, KEVIN SUTANTO, EDDY WIJANTO

Program Studi Teknik Elektro, Universitas Kristen Krida Wacana, Indonesia
Email: nindy.2016te006@civitas.ukrida.ac.id

Received 11 Juli 2023 **|** *Revised* 2 Agustus 2023 **|** *Accepted* 29 Agustus 2023

ABSTRAK

Mesin Computer Numerical Control (CNC) telah umum digunakan saat ini, salah satunya dalam pembuatan Printed Circuit Board (PCB). Pada paper ini, mesin pencetak PCB berbasis image processing dirancang dengan menggunakan Raspberry Pi, dimana file jalur PCB yang dibuat dengan menggunakan aplikasi Eagle akan di-export menjadi bentuk image grayscale, dan disimpan dengan resolusi 300 dpi. Sistem yang dirancang memiliki kemampuan membaca dpi dan menghitung jumlah step untuk setiap piksel, serta menghasilkan koordinat jalur dan lubang pada image PCB. Selain untuk pengikisan jalur, alat ini juga dapat mengebor lubang pada PCB secara otomatis. Raspberry Pi digunakan untuk mengendalikan motor stepper agar dapat bergerak sesuai dengan koordinat image PCB yang telah diperoleh. Alat ini juga dilengkapi dengan graphical user interface (GUI) untuk mempermudah user dalam mengendalikan mesin CNC. Dari hasil yang didapatkan, data erosi strip PCB menunjukkan bahwa metode kompensasi lebar strip PCB yang direncanakan mencukupi karena lebih besar 0,1-0,2mm dari ukuran strip pada Eagle.

***Kata kunci**: CNC, PCB, Image Processing, Printing and Drilling*, *Raspberry Pi*

ABSTRACT

Computer numerical control (CNC) equipment is commonly used for the production of printed circuit boards (PCBs). In this paper, the PCB path file generated by the Eagle application will be exported as a grayscale image and saved with a resolution of 300 dpi for image processing-based PCB printing equipment. The system has the capacity to produce the path and hole coordinates on the PCB image, read dpi, and determine the number of steps for each pixel. This device has the ability to automatically drill holes on the PCB in addition to stripping the path. The stepper motor is managed by the Raspberry Pi and moved in accordance with the acquired PCB image coordinates. A graphical user interface (GUI) is included to make it simpler for users to operate the CNC machine. From the results, the PCB strip erosion data show that the planned PCB strip width compensation method is sufficient because it more than 0.1-0.2mm greater than the strip size on Eagle.

***Keywords**: CNC, PCB, Image Processing, Printing and Drilling*, *Raspberry Pi*

1. INTRODUCTION

A copper-plated board known as a printed circuit board (PCB) is used to connect electrical components **(Budiprayitno et al., 2014)**. The copper layer that is unnecessary on the PCB board is removed in order to create the path on the PCB. Traditionally, an etching technique employing an iron(III) chloride (FeCl3) to remove the copper layer from the PCB. Iron (III) chloride (FeCl3) is a type of hazardous and toxic waste that can harm human health **(Ou et al., 2014)**. In addition to the etching procedure, a Computer Numerical Control (CNC) machine can be used to manufacture PCBs. A CNC machine is a particular kind of machine whose motion is managed by a computer via a specially created program **(Madekar et al., 2016)**. Some machine tools that can operate on a CNC machine, such as lathes, drilling machines, and etc. The G-code file is extracted from the PCB image file that was going to be printed and inputted into the CNC machine **(Sathyakumar et al., 2017)**. The G-code is then acquired and delivered to Arduino via serial communication using a PC. The CNC machine's stepper motor will be driven by Arduino using the G-Code **(Anggi et al., 2020)**. A camera that is parallel to the spindle is utilized to determine the machine's initial location **(Nayak et al., 2017)**. The camera is positioned so that it faces the PCB in order for the crosshairs to be seen when the images are enlarged. The proposed CNC tool cannot be taken around because it is still attached to a personal computer (PC).

In addition, in order to help the user operate the PCB CNC milling control module **(Megalingam et al., 2018)**, the graphical user interface (GUI) also interprets all user input, converts gerber files to G-code, and delivers data to the FPGA via RS232 **(Ou et al., 2014)**. Every piece of data that is received by an FPGA is translated into a command signal and sent to the rotational speed control and position control modules **(Purnomo et al., 2015)**. In a single setup, the CNC PCB Milling and Wood Engraving Machine combines the design and operation of a CNC machine to produce PCBs. When the cost of testing circuit production is taken into account, PCB creation is quite expensive in the current market. Accordingly, this CNC PCB Milling machine would be a useful tool for cheaply and effectively scratching and penetrating circuits **(Ghodmare et al., 2017)**. This CNC machine was developed to make machines less expensive and more versatile **(Choudhary et al., 2017)**.

Further, image processing techniques were employed in Milling machine control system. MATLAB is the software used to process the image, and the Alf and Vegard's Risc processor (AVR) microcontroller is the mechanical controller hardware used to operate the stepper motors. In order to separate the objects in the printed PCB image file, the PCB image file must first be converted to a binary image and processed using the segmentation method **(Susanto, 2019)**. Further, using the Sobel method **(Asmaidi et al., 2019)**, find edges; the edge detection results will provide pixel coordinates. The [X, Y] coordinate value of an image pixel can be retrieved by utilizing a contour tracking technique or the so-called Moore-Neighborhood Tracing algorithm **(Bharat et al., 2016)**. The microcontroller in this study performs a number of tasks, including limit switches, limiting the end point of each axis movement, and displaying status on the 16x2 LCD **(Munazilin, 2013)**. It also converts data from a PC and processes it to drive stepper motors and servo motors. Drilling holes in the PCB must be done manually, and the CNC tool is still linked to the PC, as it can only be used to scrape PCB paths **(Shi et al., 2017)**.

In this paper, an automatic PCB printing machine was designed based on image processing using a Raspberry Pi. The PCB route is created using the Eagle software, and after it is ready to print, the design is exported as a grayscale image file. Further, a program created to identify the location of PCB routes and drill holes on the PCB. Because this device has a Raspberry Pi within, it can operate without being connected to a personal computer (PC) and can be moved around. The Raspberry Pi is used to control CNC machines that drill holes in PCBs and scrape copper, in addition to processing images. In this system, a GUI is also created to simplify tool operation for users. The contributions of this proposed system are as follow:

1. Design a technique to locate and measure the diameter of drilling holes in PCB images.
2. Design a mechanism for determining the blade's starting position and motion in order to scrape copper off of the PCB.
3. Design a technique for moving the stepper motor in the X, Y, and Z directions in accordance with the coordinates when scraping copper off of the PCB.
4. Design a technique to account for the blade's breadth as a result of copper corrosion on the PCB.
5. Design a strategy for drilling the PCB's hole points.

2. METHODS

2.1 *Hardware Design*

Figure 1 depicts the operation of the route printing system and the drilling of holes on the proposed PCB. Three axes—X, Y, and Z—have been incorporated into the design of this automatic PCB printing system. The movement of each stepper motor utilized by a CNC machine is represented by these three axes. A limit switch is included on each axis to establish the starting point of each axis. Each stepper motor is driven by Raspberry Pi using a stepper motor driver. The circuit of copper removal system and PCB hole drilling is shown in Figure 2.

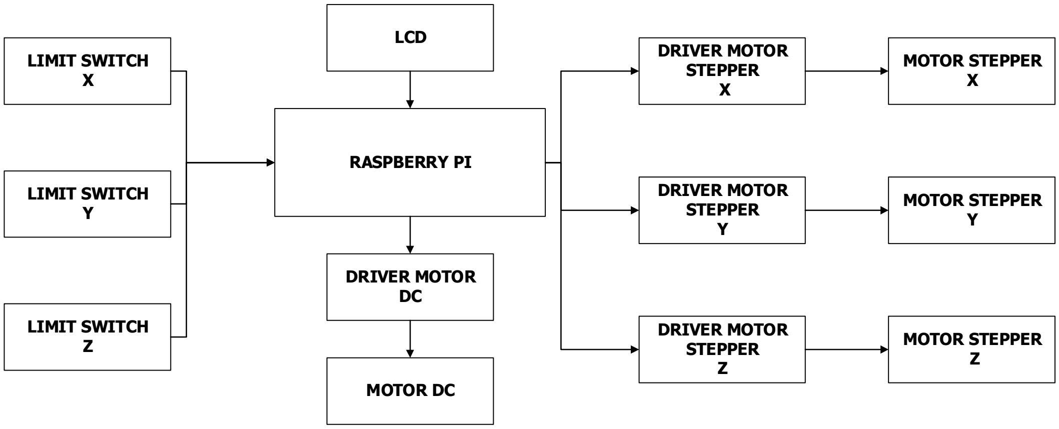


Figure 1. Track Printing and Drilling System on the PCB

Each stepper motor is attached to the Raspberry Pi using the A4988 motor driver, which is turned on by setting the enable pin (ENA) to LOW. The pin direction (DIR) is set to HIGH or LOW to indicate the stepper motor's rotational direction. The stepper motor will revolve clockwise if the DIR pin is given a HIGH state, and counterclockwise if the DIR pin is given a LOW condition. A micro switch connects the A4988 motor driver's pins MS1, MS2, and MS3 together. The stepper motor's step conditions are set using this pin. The X and Y stepper motors employed a 1/16 step condition, where all of the pins are conditioned HIGH. The Z stepper motor uses a full step condition, where all of the pins are conditioned LOW. The X, Y, and Z stepper motors have steps that are separated by 0.00125mm and 0.02mm, respectively.

In this design, limit switch is used to establish the starting positions of each axis. Three pins, designated Common (Com), Normally Close (NC), and Normally Open (NO), make up the limit switch in use. The Com pin is attached to the Raspberry Pi's general purpose input-output (GPIO) pins 21, 22, and 23 as an input, while the NO pin is connected to +5V and the NC pin is connected to GND, resulting in an output that is HIGH when the pin is pressed and LOW when it is not. A series of light emitting diode (LED) indicators are also connected to the Com pin through an 820 ohm current-limiting resistor where it will light up when the limit switch is depressed.

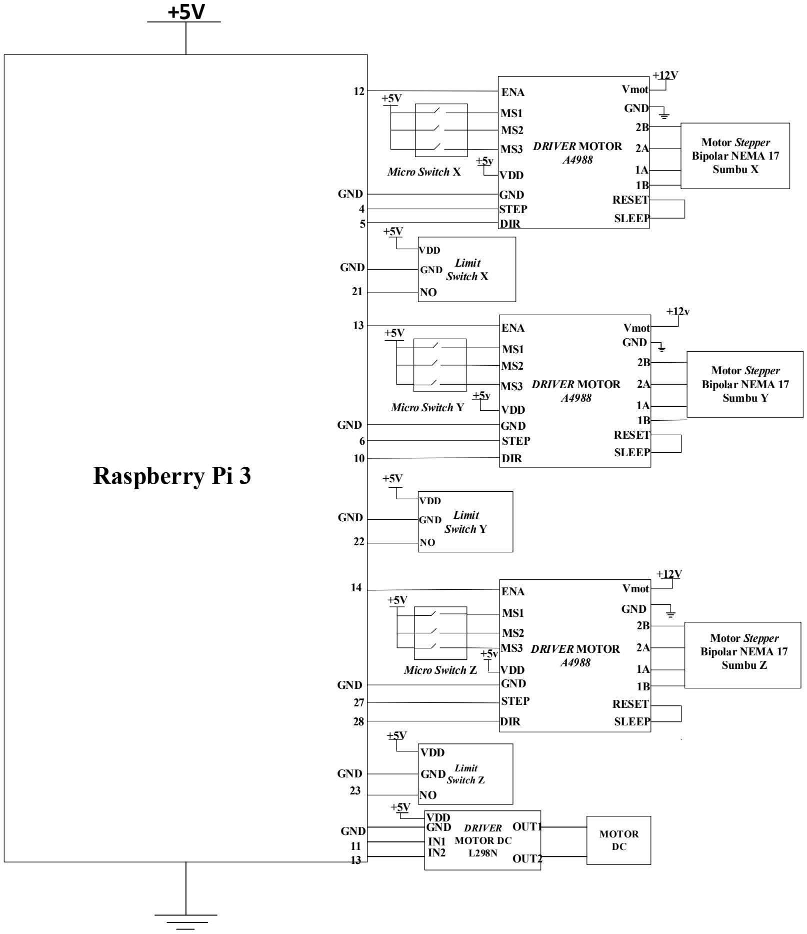


Figure 2. Schematic of The Copper Removal System and Drilling Holes on PCB

The L298N motor driver is used to link the DC motor to the Raspberry Pi. In order to turn on the L298N driver, the ENA pin must be connected to +5V, and to drive the DC motor, one of the input pins (IN1 or IN2) must be set to LOW and the other to HIGH. The DC motor will rotate counterclockwise if IN1 is given a HIGH condition and IN2 is given a LOW condition. On the other hand, the DC motor will rotate clockwise if IN1 is given a LOW condition and IN2 is given a HIGH condition. The direction of DC motor rotation must be changed to clockwise in order to apply PCB drilling and scraping in this system, where IN1 is set to HIGH and IN2 to LOW.

Three axes in the proposed automatic PCB printing machine correspond to the movement of each stepper motor. The X axis represents the forward and backward movement of the bed where the PCB is positioned, the Y axis indicates the right or left movement of the blade/drill bit, and the Z axis represents the up and down movement of the blade/drill bit. At the beginning of each axis, three limit switches are positioned. When the limit switch is depressed

(ON), the Raspberry Pi pin will get a HIGH state. This limit switch serves as an input to the Raspberry Pi. The stepper motor is in its starting axis position while the HIGH condition is present.

2.2 *Software Design*

The software block diagram for the automatic PCB printing machine is displayed in Figure 3. The user must first upload an image before the image file is analyzed to determine the locations of the PCB's paths and holes. The created GUI will allow the Raspberry Pi to wait for and read user commands. The commands that are read by the Raspberry Pi will be utilized to control the device's stepper motor.

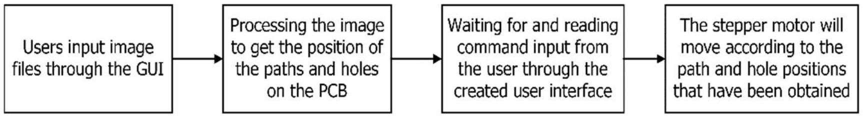

Users input image files through the GUI
Processing the image to get the position of the paths and holes on the PCB
Waiting for and reading command input from the user through the created user interface
The stepper motor will move according to the path and hole positions that have been obtained

Figure 3. Block Diagram of Automatic PCB Printing Machine Software

The location and diameter of the holes on the PCB will further be determined using the PCB file that has been transformed into a grayscale image. Figure 4 shows the steps involved in determining the location and diameter of PCB holes.

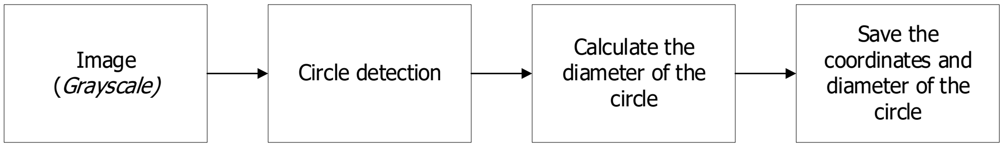


Image (*Grayscale*)
Circle detection
Calculate the diameter of the circle
Save the coordinates and diameter of the circle

Figure 4. Image Processing to Get the PCB Hole's Position and Diameter

The circle on the PCB image is detected with the parameters presented in Table 1.

Table 1. Parameter of PCB Image Circle

No.	Parameter	Value
1.	Detected area	5 pixels
2.	Value of roundness/circularity	More than 0.7
3.	Convexity value	> 0.8
4.	Inertia ratio value	< 0.01

To calculate the diameter of a circle, it is necessary to know the size of the pixels that make up an image. Pixel size in millimeters can be obtained from the image resolution value using Equation (1), while the diameter of a circle can be calculated using Equation (2).

$$pixel\ size = \frac{1}{resolution} \times 25,4mm \tag{1}$$

$$diameter(mm) = pixel\ diameter(pixel) \times pixel\ size \tag{2}$$

The positions and hole diameters are recorded in an array after the circles in the PCB image have been identified and calculated. The hole size on the GUI display will be activated using the position and hole diameter that have been stored in the array.

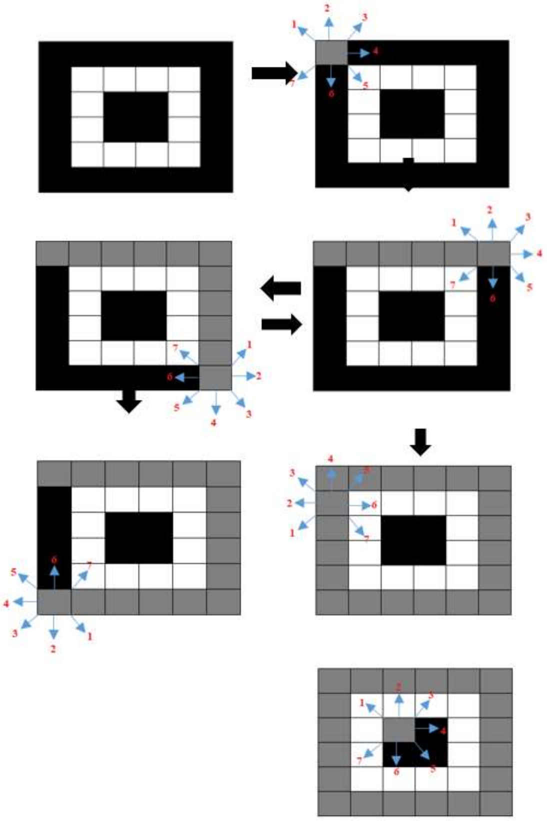


Figure 5. Pixel Tracing Process

The bed is first repositioned to its starting position as a reference point to initiate the copper removal procedure on the PCB. It is necessary to first scan every pixel in order to establish the beginning point for pixel scraping. Starting at position 0.0, the scan operation is carried out line by line. If a pixel in the scanning process has a value of 0, that pixel will be used as the starting point. The stepper motor will move the blade to the location, and the Z-axis stepper motor will turn on the rotation of the blade, which will cause the color of the pixel to change from black to gray in the animation, signifying that the pixel has been erased. The pixel tracing operation will be completed once the scraping starting point has been acquired.

Figure 5 presents the PCB image used to illustrate the pixel tracing process. The pixel in position 0.0 has a value of 0, which indicates that the erasing process will start from this position. The pixel search procedure will further start and continue until it locates a pixel that satisfies the criteria and is located close to the current pixel with a value of 0. In other words, the process of verifying pixels around the present pixel is done clockwise. The first place to be examined is the pixel on the left diagonal, the top of the pixel, the top right diagonal, to the right of the pixel. The blade will erode the pixel to the right of the current pixel because that pixel is 0.

The scan procedure will run once more after the pixel search phase has finished looking for all nearby pixels with 0 values. If the scan procedure does not discover a pixel with a value of 0 in the first line, it will move on to the next line until it found. The route tracing process will continue if the scan process comes across a pixel with a value of 0. The tracing procedure won't end until the final row and column cannot be located with a pixel value of 0, at which point it has finished. The finished path tracing process is depicted in Figure 6. Mechanically, the blade will be lowered and erode the copper from the pixel with a value of 0, and it will be lifted if the pixel has a value of 1.

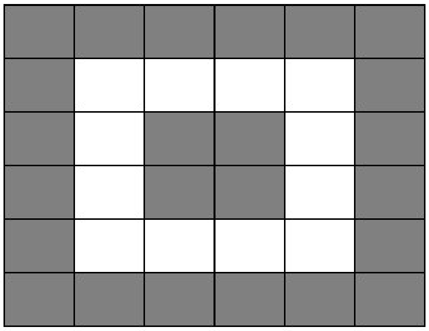


Figure 6. Pixel Search Results

The next step is to establish the direction in which the copper scraper will move after the PCB image has been processed to obtain the route location. Each blade's movement must be adjusted to the observed pixels' row and column coordinates. Here is a method of moving the blade which is designed to move to the starting point of scraping which is where the difference between the current position of the blade and the pixel to be aimed at. The stepper motor's movement direction will be determined by this difference.

To move the blade, there are conditions that must be met, namely if a > 0 and b = 0, the blade will move to the right; if b > 0 and a = 0, the blade will move down; if a < 0 and b = 0, the blade will move to the left; if b < 0 and a = 0, the blade will move up; where a = X destination position – X current position and b = Y destination position – Y current position.

Figure 7 shows the direction of the blade’s movement to the right, down, left, and up. Red color shows the target position while black color presents the present position

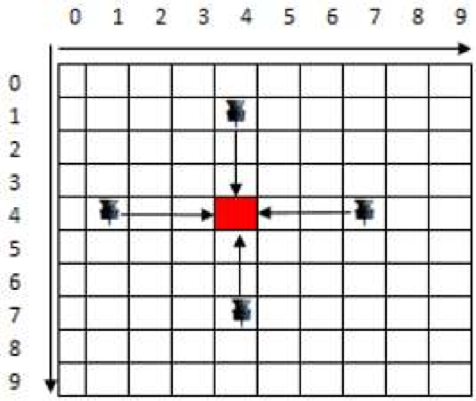


Figure 7. Movement Direction of The Blade

Further, if a > 0 and b > 0, then the blade will move first to the lower right diagonal until the blade is in the same line as the intended position, then the blade will move to the right; if b > 0 and a < 0, then the blade will move first to the upper right diagonal until the blade is in the same line as the intended position, then the blade will move to the right; if a < 0 and b > 0, then the blade will move first to the lower left diagonal until the blade is in the same line as the intended position, then the blade will move to the left; if b < 0 and a < 0, then the blade will move first to the upper left diagonal until the blade is in the same line as the intended position, then the blade will move to the left. Figure 8 shows the direction of the blade’s movement to the lower right diagonal, upper right diagonal, lower left diagonal, and upper left diagonal to the intended position.

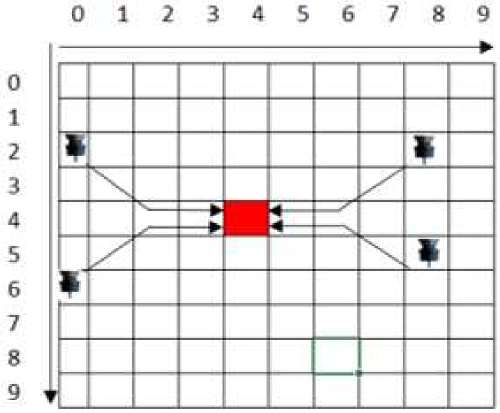


Figure 8. Movement of The Blade Diagonally

The PCB image that has been processed in accordance with the outcomes of strip erosion is adjusted using the dilation method. The breadth of the blade being used determines the 0.4mm required for one scraping. The width of the output of a single scraping determines the size of the dilation matrix that is employed. Equation (3) is used to calculate the matrix size:

$$L = Up \times Ud \tag{3}$$

where:

L = Width for one pixel scraping

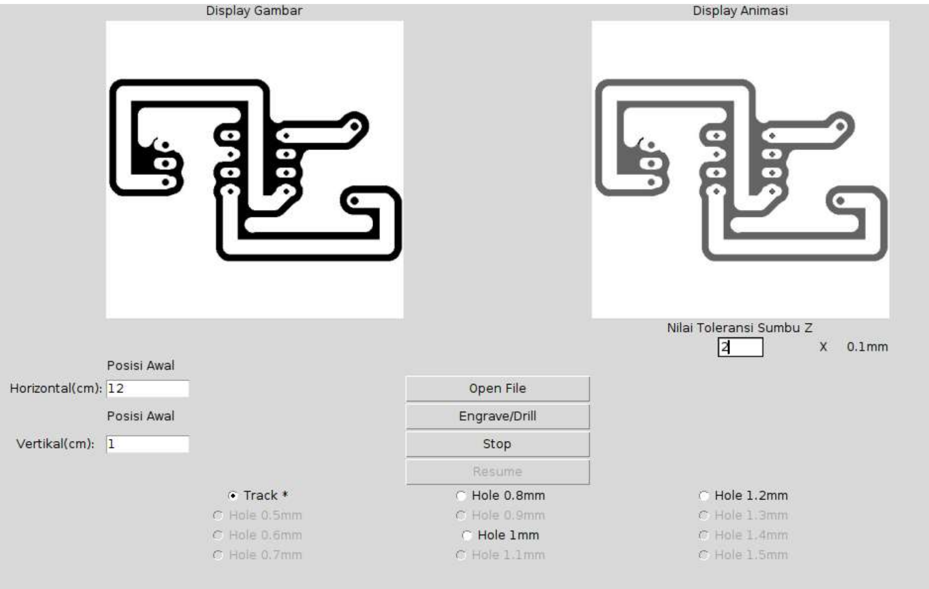
Up = Pixel size (0.0846667)

Ud = Dilation size

The matrix size used in the designed program is 5x5. If the value is substituted in Equation (3), the pixel width will be 0.4233335 mm once scraped, resulting in an error of 0.0766665 mm for one pixel scraping and a path width that is larger than the original size. This occurs as a result of pixels that are not completely scraped out during the scraping process. In contrast, if a 3x3 matrix size was selected, the route's width is 0.254mm, which means that for each pixel scraped, an inaccuracy of 0.246mm will be discovered, making the path narrower than it was originally. This occurs as a result of the pixel scraping exceeding the permitted number of pixels. Thus, the path width will be higher than the original PCB image. The larger the size of the matrix used, and conversely, the path width will be lower the smaller the dilation employed.

The diameter size chosen by the user in the GUI will then be matched using the hole placements and diameters that have been saved in the array. The drill bit will further be moved to the site of the hole by the stepper motor and lowered for a brief period of time while penetrating the PCB. It will then be raised once more and moved to the place of the next hole. This utility includes a GUI to make things easier for users. Figure 9 depicts the planned GUI design. To use the tool, click the Open File button and select the PCB image file to be printed. The picture and animation display will then show the opened PCB image. When the image displayed, the Engrave/Drill button and radio buttons for Tracks and Holes read from the PCB image are engaged automatically.

The GUI also includes initial position settings for beginning PCB printing, as well as Z-axis tolerance value options for lowering the blade/drill bit if the number of steps necessary is less. Following the execution of the Engrave/Drill process, the Stop button will become active, to stop the scraping/drilling operation. When the Stop button is hit, the Resume button becomes active to continue scraping/drilling. Following the completion of the scraping/drilling operation, the GUI display will automatically display an asterisk (*).



Nilai Toleransi Sumbu Z: 2 X 0.1mm

Posisi Awal Horizontal(cm): 12

Posisi Awal Vertikal(cm): 1

● Track *
○ Hole 0.5mm
○ Hole 0.6mm
○ Hole 0.7mm
○ Hole 0.8mm
○ Hole 0.9mm
○ Hole 1mm
○ Hole 1.1mm
○ Hole 1.2mm
○ Hole 1.3mm
○ Hole 1.4mm
○ Hole 1.5mm

Figure 9. GUI Display

3. RESULTS AND DISCUSSION

System testing is done in stages. The circuit test points on the proposed system include the Raspberry Pi circuit with stepper motors, the Raspberry Pi build with a DC motor, the limit switch circuit, the hole detection method on the PCB image, the hole diameter calculation method on the PCB image, the hole position calculation method on the PCB image, the starting point detection method, the direction of erosion determination method, the drilling method, the blade width compensation method on the scraping results, the GUI function, and overall system testing. Figure 10 shows an unprocessed PCB image.

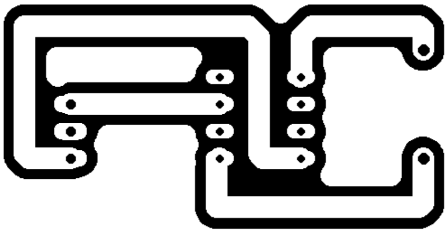
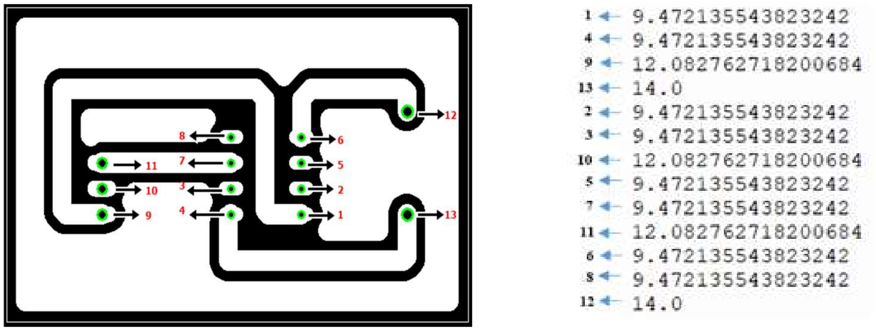


Figure 10. Unprocessed PCB Image

Figure 11(a) presents that all of the holes in the PCB image were correctly detected, and Figure 11(b) shows the discovered hole diameter data. Figure 11 demonstrates that the hole detection method on the PCB image works effectively where all of the holes on the PCB image can be detected correctly.

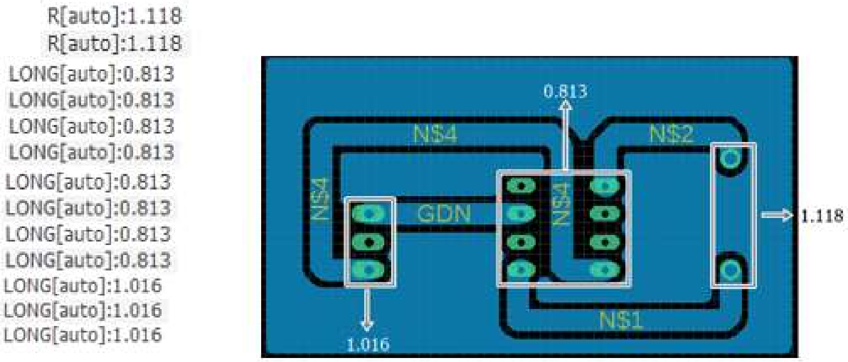


```
1 ← 9.472135543823242
4 ← 9.472135543823242
9 ← 12.082762718200684
13 ← 14.0
2 ← 9.472135543823242
3 ← 9.472135543823242
10 ← 12.082762718200684
5 ← 9.472135543823242
7 ← 9.472135543823242
11 ← 12.082762718200684
6 ← 9.472135543823242
8 ← 9.472135543823242
12 ← 14.0
```

(a) (b)

Figure 11. (a) Detected Hole in The PCB Image, (b) Diameter of The Detected Hole

The PCB image hole diameter calculation method was tested to see whether the PCB image hole diameter calculated by the designed method corresponded to the hole diameter on the PCB design created with the Eagle software.



```
R[auto]:1.118
R[auto]:1.118
LONG[auto]:0.813
LONG[auto]:0.813
LONG[auto]:0.813
LONG[auto]:0.813
LONG[auto]:0.813
LONG[auto]:0.813
LONG[auto]:0.813
LONG[auto]:0.813
LONG[auto]:1.016
LONG[auto]:1.016
LONG[auto]:1.016
```

Figure 12. Hole Diameter on The Eagle

Table 2. Hole Diameter Size on program and on Eagle

No.	Diameter on Program (mm)	Diameter on Eagle (mm)	Difference in Diameter (mm)
1	1.185	1.118	0.067
2	1.185	1.118	0.067
3	0.802	0.813	-0.011
4	0.802	0.813	-0.011
5	0.802	0.813	-0.011
6	0.802	0.813	-0.011
7	0.802	0.813	-0.011
8	0.802	0.813	-0.011
9	0.802	0.813	-0.011
10	0.802	0.813	-0.011
11	0.802	0.813	-0.011
12	1.023	1.016	0.007
13	1.023	1.016	0.007

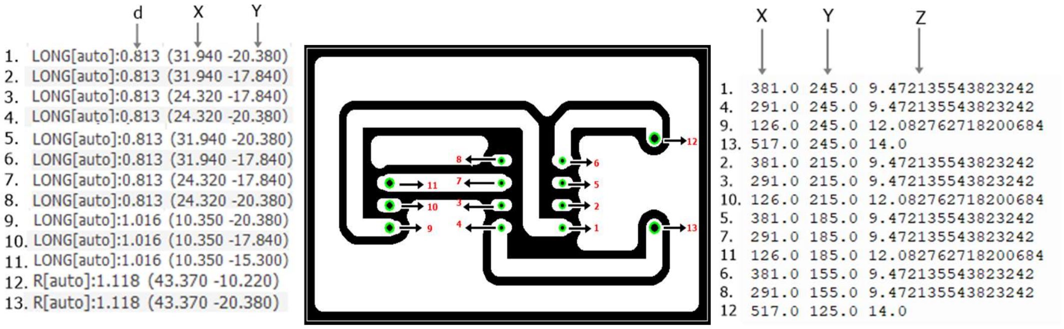
Figure 12 shows the hole diameter data on the PCB design at Eagle. The test results in Table 2 shows that the calculation of the PCB image hole diameter in Eagle is close to the PCB design diameter value with a difference of 0.007-0.067mm, where the value of this difference can still be tolerated due to the size of the drill bit available in certain markets. The two decimal places on the calculation results are rounded up. The program designed has performed well.

The method for calculating the hole position of the PCB image is tested to see whether the detected hole position is in agreement with the PCB design in Eagle, ensuring that there is no error in the PCB image drilling position. Because the diameter location collected during the hole detection procedure is still in pixels, the diameter findings will be translated to millimeters. Table 3 presents the results of a comprehensive test of the hole position calculating method on the PCB, as well as a comparison of the detection results and the hole positions on Eagle. The results in Table 3 reveal that the calculation of the PCB image hole position differs from a fixed value. This demonstrates that the distance between the holes is the same; however, the distance between the holes measured from the edge of the PCB differs. This occurs because the size of the PCB image in the Eagle design differs when converted to pixels in the image.

The starting point detection method is tested to determine whether the designed approach successfully detects the starting point of copper erosion on the PCB picture. Figure 13 shows the results of hole position detection.

Table 3. Comparison of the Hole Position

Hole Number	Detected Hole Position (Pixel)		Detected Hole Position (mm)		Hole Position on Eagle (mm)		Difference in Hole Position (mm)	
	X	Y	X	Y	X	Y	X	Y
1	381	245	32.258	20.743	31.940	20.380	0.318	0.363
2	381	215	32.258	18.203	31.940	17.840	0.318	0.363
3	291	215	24.638	18.203	24.320	17.840	0.318	0.363
4	291	245	24.638	20.743	24.320	20.380	0.318	0.363
5	381	185	32.258	15.663	31.940	15.300	0.318	0.363
6	381	155	32.258	13.123	31.940	12.760	0.318	0.363
7	291	185	24.638	15.663	24.320	15.300	0.318	0.363
8	291	155	24.638	13.123	24.320	12.760	0.318	0.363
9	126	245	10.668	20.743	10.350	20.380	0.318	0.363
10	126	215	10.668	18.203	10.350	17.840	0.318	0.363
11	126	185	10.668	15.663	10.350	15.300	0.318	0.363
12	517	125	43.773	10.583	43.370	10.220	0.318	0.363
13	517	245	43.773	20.743	43.370	20.380	0.318	0.363



```
1.  LONG[auto]:0.813 (31.940 -20.380)
2.  LONG[auto]:0.813 (31.940 -17.840)
3.  LONG[auto]:0.813 (24.320 -17.840)
4.  LONG[auto]:0.813 (24.320 -20.380)
5.  LONG[auto]:0.813 (31.940 -20.380)
6.  LONG[auto]:0.813 (31.940 -17.840)
7.  LONG[auto]:0.813 (24.320 -17.840)
8.  LONG[auto]:0.813 (24.320 -20.380)
9.  LONG[auto]:1.016 (10.350 -20.380)
10. LONG[auto]:1.016 (10.350 -17.840)
11. LONG[auto]:1.016 (10.350 -15.300)
12. R[auto]:1.118 (43.370 -10.220)
13. R[auto]:1.118 (43.370 -20.380)
```

```
1.  381.0 245.0 9.472135543823242
4.  291.0 245.0 9.472135543823242
9.  126.0 245.0 12.082762718200684
13. 517.0 245.0 14.0
2.  381.0 215.0 9.472135543823242
3.  291.0 215.0 9.472135543823242
10. 126.0 215.0 12.082762718200684
5.  381.0 185.0 9.472135543823242
7.  291.0 185.0 9.472135543823242
11  126.0 185.0 12.082762718200684
6.  381.0 155.0 9.472135543823242
8.  291.0 155.0 9.472135543823242
12  517.0 125.0 14.0
```

(a) (b)

Figure 13. (a) Hole Position on Eagle, (b) Detected Hole Position

The results of the tests reveal that the PCB starting point detection method works properly. The detection result is in the form of the initial place to begin the scraping process. Figure 14 shows the 0.0 location on Eagle and the image's beginning point.

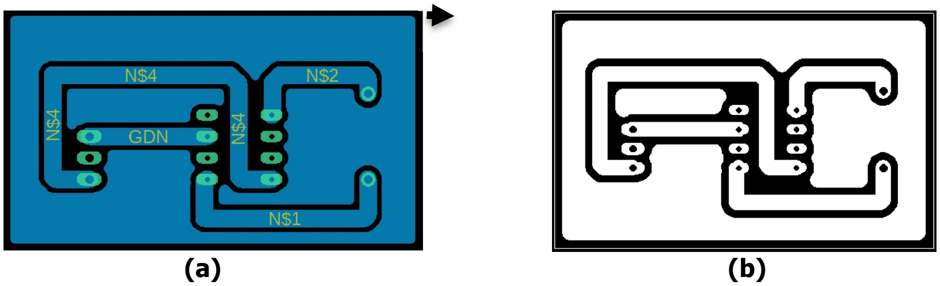


Figure 14. (a) Position 0.0 on Eagle (b) Starting Point in The Image

The beginning point can be determined based on the outcomes of the tests, implying that the starting position detection has worked successfully.

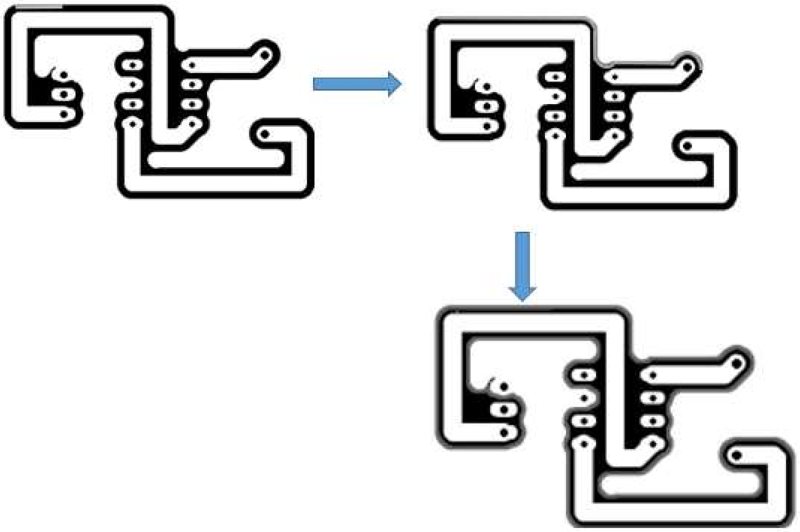


Figure 15. Movement of The Blade

To verify that the blade can move to reach all areas of copper to be scrapped, the method of identifying the direction of erosion is tested. The test results demonstrate that the planned erosion direction approach is effective. The detection produces strips on the PCB as a result. Figure 15 depicts the movement of the blade from the starting location to the start of scraping. Further, Figure 16 shows that the scraper moves in the direction specified by the procedure.

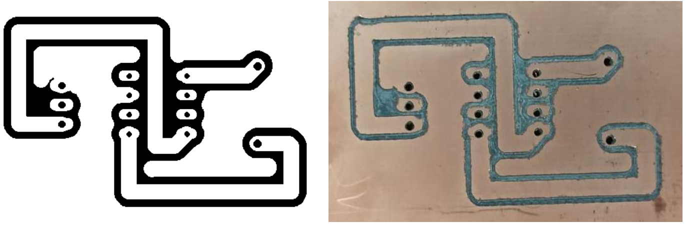


Figure 16. Hole Drilling Results

Table 4 presents a comparison of PCB path width measured from several different positions. The test results in Table 4 shows that the hole drilling on the PCB is done according to the design procedure. According to the test results, it appears that all of the holes were successfully drilled using the intended tool. The blade width compensation method is tested on scraping results to confirm that the width of the strip on the PCB image corresponds to the width of the strip of copper removed from the scrape result. From the table, there is a 0.1-0.2mm variance in strip width between the Eagle design and the erosion results. The width of the strip set on Eagle is 2mm, while the width of the strip resulting from scraping is 2.1-2.2mm. As a result, the PCB width compensation method performed as intended, with the PCB width discrepancy nearing the calculated value.

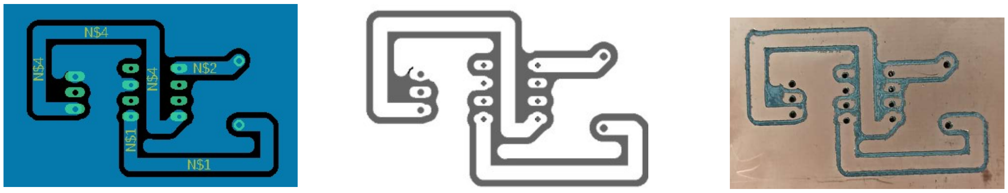
Table 4. PCB Path from Several Positions

Position	Measurement Results (mm)	Path on Design (mm)	Difference (mm)
1	2.2	2	0.2
2	2.2	2	0.2
3	2.2	2	0.2
4	2.1	2	0.1
5	2.1	2	0.1

Table 5. GUI Function Testing

Tested Menu	Menu Function	Test Results
Horizontal and vertical starting position	Set the starting position	The blade can move to the initial position according to the position that has been set
Canvases 1 and 2	Displays pictures and animations	Canvas can display PCB images and animations to be scraped
File Open button	To open image files	The open file button can execute a command to open the PCB file to be printed
Engrave/Drill button	To run the engrave and drill commands	The engrave and drill buttons can execute engrave and drill commands on the PCB
Stop button	To stop the scraping/drilling process	The stop button can work to stop the PCB scraping/drilling process
Resumes button	To continue the scraping/drilling process	The Resume button can work to continue the PCB scraping and drilling process after the previous stop process
Z Axis Tolerance Value	To lower the Z axis	The tolerance value can be used to increase the step on the Z axis
Track and All Hole Selection	As a menu of options to be executed	Tracks and holes can be selected and run

The goal of testing the designed GUI function is to guarantee that all menus run as expected. Table 5 displays the results of the GUI function testing, which shows that each menu displayed on the GUI functions according to the program that has been executed. According to Table 4, all of the menus are running well. Final test was conducted to observe whether the proposed system can function as intended. Overall testing of the image processing-based automatic PCB printing machine using the Raspberry Pi is performed by connecting all previously tested components and executing the scraping and drilling program on the Raspberry Pi.



(a) (b) (c)

Figure 18. Automatic PCB Printing Machine Results (a) Image from Eagle, (b) Animation Image, (c) Scraping Results

According to the proposed system test results in Figure 18, the proposed automatic PCB printing machine can conduct scraping based on the PCB image to be printed.

4. CONCLUSIONS

In the proposed automatic PCB printing machine, the hole detection method on the intended PCB image can detect every hole and determine its position and dimension. The diameter and position obtained varies by less than 0.1mm when compared to the size of the hole and the real size on the Eagle. The developed pixel scanning technology can accurately detect the starting point of copper erosion on the PCB. In the process of scrapping copper on the PCB, the stepper motor moves in the X, Y, and Z directions in accordance with the designed method, with the blade moving in accordance with the position of the row and column of pixels to be scrapped. The process of drilling holes on the PCB follows the designed procedure, and all holes can be drilled accurately based on the detected position and diameter. The PCB strip erosion results achieved are more than 0.1-0.2mm than the strip size on Eagle, indicates that the designed PCB strip width compensation method are adequate. The proposed system has a simple, cost-effective, and relatively high precision feature. Further research will focus on the system improvement, including the bed on which the PCB is put is repositioned so that it is not inclined, resulting in the same depth of scraping results. The depth at the moment of scraping also need to be considered in order to achieve better scraping results. The size of the drill bit that can be utilized can be further increased, allowing for the drilling of increasingly larger hole sizes.

REFERENCES

Anggi, H., Aris, S., & Nur, Y. (2020). Rancang Bangun Pembuat Layout PCB Otomatis Berbasis Android*. Journal of Electrical Vocational Education and Technology, 5*, 7-12.

Asmaidi, A., Putra, D. S., Risky, M. M., & Ulfa, F. R. (2019). Implementation of Sobel Method Based Edge Detection for Flower Image Segmentation. *Jurnal & Penelitian Teknik Informatika (SinkrOn), 3*(2).

Bharat, K. P., Priyadarshani, M. S., Shriprasad, M.S., & Raghavendra, S. R. (2016). Arduino Based 3 Axis PCB Drilling Machine. *International Journal of Emerging Technologies in Engineering Research, 4*(6), 255-259.

Budiprayitno, S., Setijadi, E., Chandra, S. D., & Pranaka, A. R. (2014). Desain Sistem Kendali Mesing PCB Milling-Berbasis Image Processing. *Prosiding Seminar Nasional Aplikasi Sains & Teknologi (SNAST)*, (pp. 23-30).

Choudhary, R., Sambhav, Titus, S. D., Akshaya, P., Matthew, J. A., & Balaji, N. (2017). CNC PCB Milling and Wood Engraving Machine. *International Conference on Smart Technology for Smart Nation,* (pp. 1301-1306).

Ghodmare, R., Tandulkar, S., & Raut, C. D. (2017). PCB Engraving and Drilling Machine. *International Journal of Research in Science & Engineering, 3*(2).

Madekar, K. J., Nanaware, K. R., Phadtare, P. R., & Mane, V. S. (2016). Automatic Mini CNC Machine for PCB Drawing and Drilling. *International Research Journal of Engineering and Technology, 3*(2), 1106-1110.

Megalingam, R. K., Vadivel, S. R. R., Sreekumar, S., Sekhar, S., Nair, T. R., & Midhun, R. R. (2018). Design and Implementation of CNC Miling Bot for Milled Circuit Board Fabrication. *International Jurnal of Engineering & Technology, 7*(3.12), 1205-1209.

Munazilin, A. (2013). *Arsitektur Komputer*. Yogyakarta: Deepublish.

Nayak, J. P. R, Anitha, K., Parameshachari, B. D., Banu, R., & Rashmi, P. (2017). PCB Fault Detection Using Image Processing. *IOP Conference Series: Material Science and Engineering*, *255*, 1-7.

Ou, J., Maldonado, A., Saephan, C., Farahmand, F., & Caggiano, M. (2014). A Low-Cost PCB Fabrications Process. *IEEE 64th Electronic Components and Technology Conference (ECTC)*, (pp. 2159-2162).

Purnomo, A. T., Hariadi, F. I., & Sasongko, A. (2015). Development of Interface and Coordination for Control Module CNC PCB Milling Machine. *International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*, (pp. 246-251).

Sathyakumar, N., Balaji, K. P., Ganapathi, R., & Pandia, S.R. (2017). A Build-Your-Own Three Axis CNC PCB Milling Machine. *Materials Today: Proceedings*, (pp. 24404-24413).

Shi, H., Lin, X., & Wang, Y. (2017). Characterization of Drill Bit Breakage in PCB Drilling Process Based on High Speed Video Analysis. *Circuit World, 43*(3), 89-96.

Susanto, A. (2019). Penerapan Operasi Morfologi Matematika Citra Digital Untuk Ekstrasi Area Plat Nomor Kendaraan Bermotor. *Jurnal Pseudocode, 6*(1), 49-57.