energies

MDPI

Article

Spatial Straight-Line Drawing Algorithm Based on Method of Discriminate Regions—A Control Algorithm of Motors

Jianping Wang [1], Shiguang Xiao [1], Tao Song [2], Junqi Yue [1], Pingyan Bian [1] and Yu Li [1,*]

[1] School of Mechanical and Power Engineering, Henan Polytechnic University, Jiaozuo 454000, China; wjp@hpu.edu.cn (J.W.); xiaoshiguang-2008@163.com (S.X.); 15839160286@163.com (J.Y.); bianpy@hpu.edu.cn (P.B.)

[2] State Grid Jiaozuo Power Supply Company, Jiaozuo 454000, China; bloodingbat79211@163.com

* Correspondence: liyu@hpu.edu.cn; Tel.: +86-136-0344-6313

Received: 19 July 2020; Accepted: 19 September 2020; Published: 23 September 2020

Abstract: A novelty algorithm of spatial straight-line drawing based on a method of discriminate regions is proposed in this paper based on Bresenham's algorithm. Three-dimensional space is divided into innumerable three-dimensional meshes according to the given rule; the distance between the start and the end points of the three coordinates is Δx, Δy, and Δz, respectively; the distribution types of spatial straight line and the position of the end point are determined by judging the relationship among Δx, Δy, and Δz; then, the active-passive directions can be determined. The plane of the ending point of the straight line in a three-dimensional mesh is divided into four regions; then, the discriminant is obtained; and this discriminant determine which region the point is located in The algorithm is verified and analyzed by the method of contrastive analysis; the results show that: the error of the algorithm is related to the step length L; the maximum theoretical error is 0.7071*L. The discriminants are all integers, so the problem of deviation from the theoretical straight line caused by the retention of decimals of significant digits can be avoided. Finally, the algorithm is applied to the cooperative control of multiple motors, and conversion between unit grid number and pulse number of motors is performed.

Keywords: Bresenham's algorithm; spatial straight line; discriminate regions; integer discriminants

1. Introduction

Straight line drawing is among the important fields of various industries, such as straight-line interpolation [1], stepper motor cooperative control [2], 3D printing [3], 3D model building [4], and image drawing [5]. To quickly draw straight lines, the speed of the applied algorithm is critical. The Bresenham's algorithm [6,7], DDA algorithm [8–10], and midpoint algorithm [11–13] are two-dimensional straight-line generation algorithms. Among which, the most famous is the Bresenham's algorithm, which is introduced during the 1960s. The advantage of Bresenham's algorithm is that all the operations are integers, without division or decimals. This approach is efficient at selecting a set of grid points to represent a straight line in a two-dimensional space. Space symmetry and simplifying computation are two major aspects of the algorithm towards high efficiency [14].

Stephenson et al. [15] present a line drawing algorithm based on runs of runs and discussed a number of special cases in the structure of runs and runs; it is proved that the algorithm can also be applied to short straight lines; this algorithm is formed to draw the line that has an identical iterative structure to Bresenham's algorithm, except that at each iteration, it is not a pixel that is set but a run of runs; thus, the speed of line drawing is improved. Li et al. [16] proposed a fast line

drawing algorithm using circular subtraction based on Bresenham's algorithm. This algorithm makes use of the corresponding relationship between the two ends of the line and its symmetry to quickly draw multiple pixels, which enhances the speed of drawing straight lines. Liu et al. [1] extended Bresenham's algorithm to a spatial straight line; the straight line is decomposed into the motion of two planes, which realize a three-dimensional Bresenham's algorithm; the actual errors of the algorithm at each sampling point are less than the maximum theoretical error; however, because of the discriminant containing decimals, the actual line may deviate from the theoretical straight line due to the problem of retaining decimal digits. Dai et al. [2] also extended Bresenham's algorithm from two dimensions to three, and applied it to multidimensional stepper motor cooperative control. Although the control accuracy requirements can be met by this algorithm, it is triggered by the interruption of the timer during motion, which leads to a noncontinuous algorithm.

To solve the above problems, a spatial straight line drawing algorithm based on the method of discriminate regions is proposed in this paper. Our innovation is that the discriminant in our algorithm is the integer; only the coordinates of the starting and end point of the spatial straight line are needed to be input in this algorithm; then, the distribution of spatial straight lines in three-dimensional space and the location of the end point can be quickly determined, and then the corresponding discriminant is obtained and the selection points of lines in each three-dimensional mesh are determined. Finally, the selected points are connected to complete the spatial space line drawing. This method can avoid the decimal in the discriminant completely, so all the operations are integers in the algorithm of this paper; therefore, the problem of deviation from the theoretical straight line caused by the retention of decimals of significant digits will be avoided. This algorithm is applied to the cooperative control of the stepping motor. A proportional integral differential (PID) control algorithm is often used for the control of a single motor and the parameters need to be tuned [17]. This paper mainly explains the process of multistep motor cooperative control.

2. Space Line Drawing Algorithm

In this paper, Bresenham's algorithm is extended from a two-dimensional plane to a three-dimensional space. Meantime, in order to avoid the problem of deviation between the actual straight line and the theoretical straight line caused by the decimals of the discriminant, a method of discriminate regions is designed, and the discriminant is improved.

2.1. Analysis of the Two-Dimensional Straight Line of Bresenham's Algorithm

A recursive step method is used in Bresenham's algorithm [18]. The axis with the largest change is taken as the active direction and it progresses one grid at a time; the other direction is taken as the passive direction, along which whether to progress one grid is decided according to the sign of the discriminant, as shown in Figure 1.

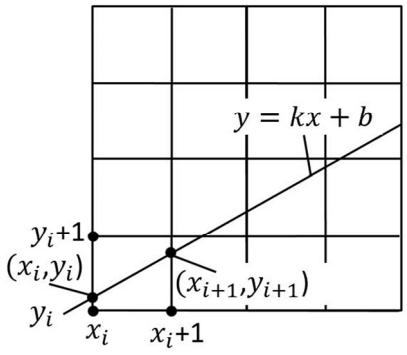


Figure 1. Schematic of Bresenham's algorithm. Note: $(x_{i+1} - x_i) > (y_{i+1} - y_i)$, so x is the active direction, y is passive direction.

2.2. Analysis of the Spatial Straight Line

From the starting to the ending points of the spatial straight line, the displacement of the three axes are generally different, but each axis will reach the corresponding end coordinates at the same time. The axis with the greatest distance from the beginning to the end of the straight line is the active direction and progresses a unit at a time; the other two axes are the passive directions and only progress at specific times to maintain their relative slopes (stop). The drawing speed and the precision of the straight line are directly determined by where and how long they will stop [19]. Therefore, the following spatial straight-line drawing algorithm is designed.

The starting point of spatial straight line is set as $p_s(x_s,\ y_s,\ z_s)$, and the end point is $p_e(x_e, y_e, z_e)$, where $\Delta x = x_e - x_s$, $\Delta y = y_e - y_s$, and $\Delta z = z_e - z_s$. Therefore, the equation for the spatial straight line is:

$$\frac{x - x_s}{\Delta x} = \frac{y - y_s}{\Delta y} = \frac{z - z_s}{\Delta z} \tag{1}$$

As three-dimensional space has intrinsic symmetry [11], only the following types for spatial straight lines in which x, y, and z are all positive directions are discussed. Assume three-dimensional space consists of innumerable three-dimensional meshes; the side length of each grid is one pixel or one step angle. In this paper, three-dimensional space is regarded as a cube, and the six surfaces of the cube are, respectively, called: front surface, back surface, left surface, right surface, upper surface, and lower surface; each three-dimensional mesh is called a unit grid, the side length of unit grid is called a unit, and its length is L, in Figure 2.

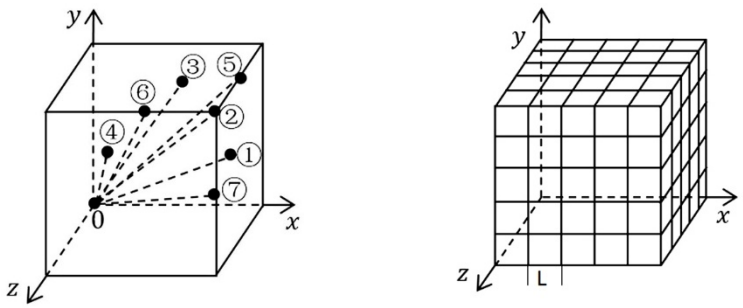


Figure 2. Three-dimensional space. (**a**) Distribution types of straight lines in three-dimensional space. (**b**) Three-dimensional space meshing.

Figure 2a shows the distribution types of straight lines in three-dimensional space; assume that the starting point of each straight line is at the origin, and the distribution of the straight line is determined by the end point. There are seven types, as shown in Figure 2a; by judging the relationship between the values of Δx, Δy, and Δz, the position of the end point can be determined as follows:

(1) If $\Delta x > \Delta y \geq 0,\ \Delta x > \Delta z \geq 0$, the straight line is type ①, as shown in Figure 2a, and the end point is on the right surface of the cube; this type of straight line corresponds to algorithm 1; x is taken as the active direction and it progresses one grid at a time; y and z are taken as passive directions, along which whether to progress one grid is decided according to the sign of the discriminant;

(2) If $\Delta x = \Delta y = \Delta z \geq 0$, the straight line is type ②, as shown in Figure 2a, and the end point is on the vertex opposite to the origin; this type of straight line corresponds to algorithm 2; x, y, and z are taken as the active direction, and progress one grid at a time;

(3) If $\Delta y > \Delta x \geq 0,\ \Delta y > \Delta z \geq 0$, the straight line is type ③, as shown in Figure 2a, and the end point is on the upper surface of the cube; this type of straight line corresponds to algorithm 3; y is taken as the active direction and it progresses one grid at a time, x and z are taken as passive directions, along which whether to progress one grid is decided according to the sign of the discriminant;

(4) If $\Delta z > \Delta x \geq 0$, $\Delta z > \Delta y \geq 0$, the straight line is type ④, as shown in Figure 2a, and the end point is on the front surface of the cube; this type of straight line corresponds to algorithm 4; z is taken as the active direction and it progresses one grid at a time, x and y are taken as passive directions, along which whether to progress one grid is decided according to the sign of the discriminant;

(5) If $\Delta x = \Delta y > \Delta z \geq 0$, the straight line is type ⑤, as shown in Figure 2a, the end point is on the intersection line of the upper surface and the right surface of the cube; This type of straight line corresponds to algorithm 5; x and y are taken as the active direction, and they are progress one grid at a time, z is taken as the passive direction along which whether to progress one grid is decided according to the sign of the discriminant;

(6) If $\Delta y = \Delta z > \Delta x \geq 0$, the straight line is type ⑥, as shown in Figure 2a, and the end point is on the intersection line of the upper surface and the front surface of the cube; this type of straight line corresponds to algorithm 6, y and z are taken as the active directions, and they progress one grid at a time; x is taken as the passive direction, along which whether to progress one grid is decided according to the sign of the discriminant;

(7) If $\Delta x = \Delta z > \Delta y \geq 0$, the straight line is type ⑦, as shown in Figure 2a, and the end point is on the intersection line of the front surface and the right surface of the cube; this type of straight line corresponds to algorithm 7; x and z are taken as the active directions, and they progress one grid at a time; y is taken as the passive direction, along which whether to progress one grid is decided according to the sign of the discriminant.

Seven types of straight lines correspond to seven algorithms. By judging the relationship of Δx, Δy, and Δz, the position of the end of the straight line is determined, so as to determine the corresponding algorithm. The flow chart of this algorithm is shown in Figure 3.

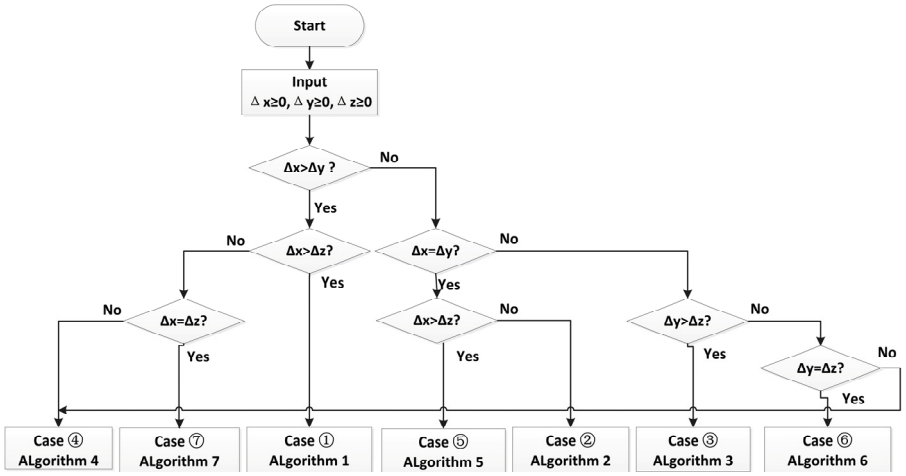


Figure 3. Flow chart for judging the distribution of straight lines in three-dimensional space.

2.3. Method of Discriminate Regions

The basic principle of all the above algorithms is the same, except for the variable (Δx, Δy, and Δz). The corresponding algorithm is determined by judging the relationship among Δx, Δy, and Δz. As an example, this algorithm 1 is designed and discussed in this paper; and the method of discriminate regions is proposed.

Three-dimensional space is divided into innumerable three-dimensional meshes; one of the unit grid is taken, as shown in Figure 4a.

(**a**) (**b**)

Figure 4. Method of discriminate regions. (**a**) A straight line in a unit grid. (**b**) The position of the desire point in the plane.

For case ①, the end point of the straight line is on the right surface of the cube, so the end point of all the straight lines in the unit grid is also on the right surface of the unit grid. Figure 4a shows a straight line in a unit grid, and the coordinate system here is the local coordinate system of each unit grid. Where p_i is the selection point of point *i*, the origin of the $(i+1)$ unit grid, and the starting point of the straight line in the $(i+1)$ unit grid. p_{i+1} is the intersection of the theoretical straight line and the upper right surface of the unit grid; because the selection point of p_{i+1} is an integer point, as shown in Figure 4a, the selection point of p_{i+1} can only be one of A, B, C, D. Each time a point is selected, it will serve as the origin of the local coordinate system of the next unit grid and the starting point of the line in the next unit grid, where $(i = 0,\ 1,\ 2 \ldots n)$. The right surface of the unit grid is divided into four regions (a, b, c, and d) by four lines (e, f, g and h); the corresponding selection points of each region are A, B, C, and D, respectively, as shown in Figure 4b. According to the discriminant, the location of point p_{i+1} in the region is determined, and the selection point is chosen. The coordinates of the four points (A, B, C, and D) are, respectively, $A(x_i+1,\ y_i+1,\ z_i)$, $B(x_i+1,\ y_i,\ z_i)$, $C(x_i+1,\ y_i,\ z_i+1)$, and $D(x_i+1,\ y_i+1,\ z_i+1)$,where d_{y1} is the distance between the actual y coordinate of point p_{i+1} and y_i, d_{y2} is the distance between the actual y coordinate of point $p_{i+1,}$ and (y_i+1), d_{z1} is the distance between the actual z coordinate of point p_{i+1} and z_i, and d_{z2} is the distance between the actual z coordinate of point p_{i+1} and (z_i+1). The y and z coordinates are determined using Equation (1), as follows:

$$\begin{cases} y = \frac{\Delta y}{\Delta x}(x_{i+1} - x_s) + y_s \\ z = \frac{\Delta z}{\Delta x}(x_{i+1} - x_s) + z_s \end{cases} \tag{2}$$

$$\begin{cases} d_{y1} = y - y_i = \frac{\Delta y}{\Delta x}(x_i + 1 - x_s) + y_s - y_i \\ d_{y2} = (y_i + 1) - y = (y_i + 1) - \frac{\Delta y}{\Delta x}(x_i + 1 - x_s) - y_s \\ d_{z1} = z - z_i = \frac{\Delta z}{\Delta x}(x_i + 1 - x_s) + z_s - z_i \\ d_{z2} = (z_i + 1) - z = (z_i + 1) - \frac{\Delta z}{\Delta x}(x_i + 1 - x_s) - z_s \end{cases} \tag{3}$$

$$\begin{cases} d_y = d_{y1} - d_{y2} \\ d_z = d_{z1} - d_{z2} \end{cases} \tag{4}$$

Then, the discriminant is obtained as:

$$\begin{cases} d_y = 2\frac{\Delta y}{\Delta x}(x_i + 1 - x_s) - 2y_i + 2y_s - 1 \\ d_z = 2\frac{\Delta z}{\Delta x}(x_i + 1 - x_s) - 2z_i + 2z_s - 1 \end{cases} \tag{5}$$

2.4. Improved Discriminant

To simplify the discriminant in Equation (5), d_y and d_z being all multiplied by Δx; then, an improved discriminant is proposed:

$$\begin{cases} M_i = 2\Delta y x_i - 2\Delta x y_i - 2\Delta y x_s + 2\Delta x y_s + 2\Delta \mathrm{y} - \Delta \mathrm{x} \\ N_i = 2\Delta z x_i - 2\Delta x z_i - 2\Delta z x_s + 2\Delta x z_s + 2\Delta \mathrm{z} - \Delta \mathrm{x} \end{cases} \tag{6}$$

because $\Delta x > 0$, M_i and d_y have the same sign, and N_i and d_z have the same sign. Therefore, we can decide which point to choose next from the signs of M_i and N_i. From Equation (6), the discriminant at $i = i + 1$ is:

$$\begin{cases} M_{i+1} = 2\Delta y x_{i+1} - 2\Delta x y_{i+1} - 2\Delta y x_s + 2\Delta x y_s + 2\Delta \mathrm{y} - \Delta \mathrm{x} \\ N_{i+1} = 2\Delta z x_{i+1} - 2\Delta x z_{i+1} - 2\Delta z x_s + 2\Delta x z_s + 2\Delta \mathrm{z} - \Delta \mathrm{x} \end{cases} \tag{7}$$

Then, the recursive equations are obtained as:

$$\begin{cases} M_{i+1} = M_i + 2\Delta y - 2\Delta x(y_{i+1} - y_i) \\ N_{i+1} = N_i + 2\Delta z - 2\Delta x(z_{i+1} - z_i) \end{cases} \tag{8}$$

As the starting point of the spatial straight line is $p_s(x_s, y_s, z_s)$, it is obtained that:

$$\begin{cases} x_1 = x_s \\ y_1 = y_s \\ z_1 = z_s \end{cases} \tag{9}$$

Considering Equation (6), the initial value of the discriminant is obtained as:

$$\begin{cases} M_1 = 2\Delta y - \Delta x \\ N_1 = 2\Delta z - \Delta x \end{cases} \tag{10}$$

2.5. The Process of Discriminate Region

Considering the positive and negative signs of the above discriminants $\mathrm{M_i}$ and $\mathrm{N_i}$, which region the point $\mathrm{p_{i+1}}$ is in can be determined, and then which point of A, B, C, and D will be selected.

(1) If $M_i > 0$ and $N_i < 0$, the point p_{i+1} is in region a; the point A will be selected at this point.
(2) If $M_i < 0$ and $N_i < 0$, the point p_{i+1} is in region b; the point B will be selected at this point.
(3) If $M_i\langle 0$ and $N_i\rangle 0$, the point p_{i+1} is in region c; the point C will be selected at this point.
(4) If $M_i > 0$ and $N_i > 0$, the point p_{i+1} is in region d; the point D will be selected at this point.
(5) If $M_i = 0$ and $N_i < 0$, the point p_{i+1} is on segment e; the point A or B will be selected at this point.
(6) If $M_i < 0$ and $N_i = 0$, the point p_{i+1} is on segment f; the point B or C will be selected at this point.
(7) If $M_i = 0$ and $N_i > 0$, the point p_{i+1} is on segment g; the point A or D will be selected at this point.
(8) If $M_i > 0$ and $N_i = 0$, the point p_{i+1} is on segment h; the point D or A will be selected at this point.
(9) If $M_i = 0$ and $N_i = 0$, the point p_{i+1} is at central point o; the point A or B or C or D will be selected at this point.

Therefore, only the signs of M_i and N_i are needed to judge to determine the next selection point. Assuming that n steps are required from the starting point to ending point of the spatial straight line, the judging algorithm flow chart is given in Figure 5.

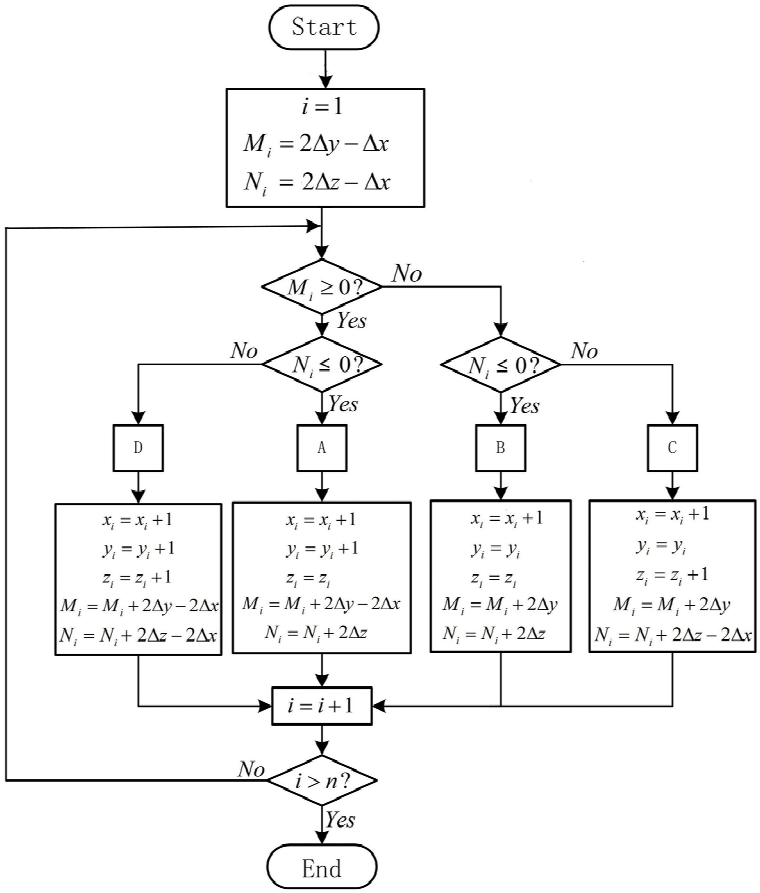


Figure 5. The judging algorithm flow chart for determining the next point.

3. Algorithm Verification and Application

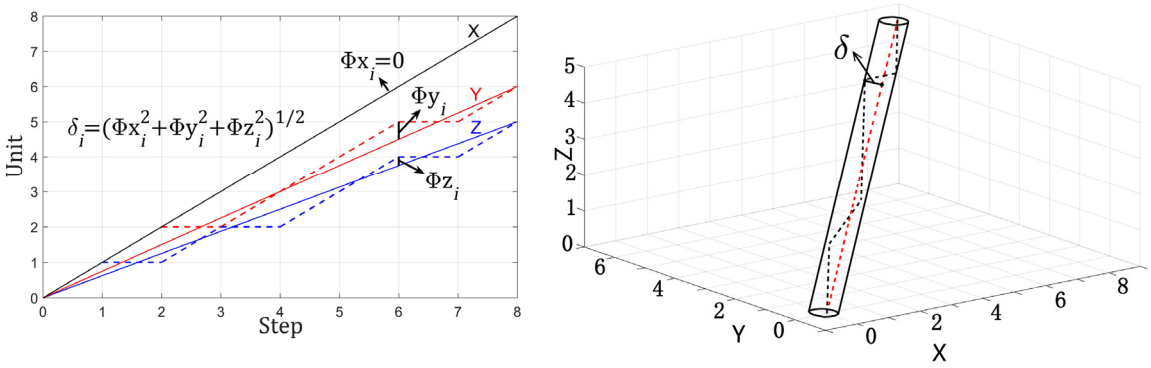
3.1. Algorithm Verification Based on Matlab

For the purpose of verification of the algorithm in this paper, it is assumed that the starting point of the spatial straight line is $p_s(0,\ 0,\ 0)$, 300 integers of 0–100 are randomly generated in MATLAB, and three numbers of them are a group of coordinates as the end point of a straight line; There are totally 100 groups, and one group of them is taken as an example, such as $p_e(8,\ 6,\ 5)$; therefore, $\Delta x = 8$, $\Delta y = 6$, and $\Delta z = 5$.

From Equation (10), the initial value of the discriminants are $M_1 = 4$ and for $N_1 = 2$. The above algorithm is used to draw the spatial straight lines, as shown in Figure 6.

In this case study, the end point coordinate for the x-axis is eight, which required eight steps forward. Thus, the x-axis progressed one unit at a time, and the actual x-axis value coincided with the theoretical value. The end points for the y and z-axis are six and five, respectively. Therefore, there are two steps without progression along the y-axis, and there are three along the z-axis. In Figure 6a, δ_i is the error of each step, and n steps needed totally for the straight-line, then:

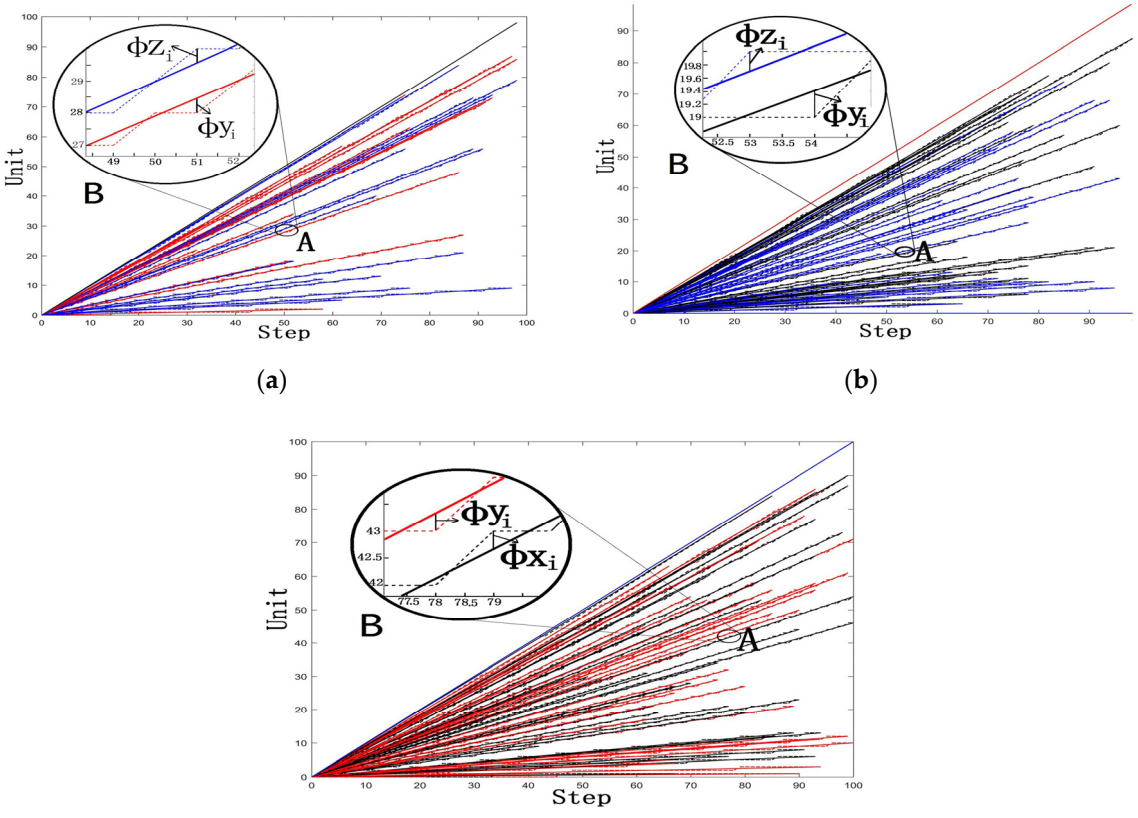
$$\delta_i = \sqrt{\phi x_i{}^2 + \phi y_i{}^2 + \phi z_i{}^2}\ , (i = 1, 2 \ldots\ n) \tag{11}$$



(**a**) (**b**)

Figure 6. Results of the spatial straight-line algorithm. (**a**) The motion process of each axis. (**b**) Spatial straight line and straightness error. Note: Figure 6a shows the motion process for each axis. The ordinate is the number of units, and the abscissa is the number of forward steps. The black, red, and blue lines represent the x, y, and z -axis. The solid line is the theoretical value, while the dashed line is the actual value.

In Figure 6b, the black dashed line is the actual generated spatial straight line, and the red dashed line is the theoretical Spatial straight line. The cylinder is the minimum inclusive region of the spatial straight-line straightness error [20], which is parallel to the theoretical straight line. All parts of the spatial straight line drawn by the algorithm are in the cylinder, whose diameter is the maximum error of the spatial straight line [21,22], and $\delta = max(\delta_i)$, all the results are shown in Figure 7.



(**a**) (**b**)

(**c**)

Figure 7. The 100 groups of straight-line results. (**a**) The x-axis with the largest change. (**b**) The y -axis with the largest change. (**c**) The z -axis with the largest change.

Figure 7a–c, respectively, show the largest change of x-axis, y-axis, and z-axis in 100 groups of data and the local enlarged drawing, and B is the enlarged figure of A, where the black line is the x-axis, red line is the y-axis, and blue line is the z-axis.

3.2. Error Analysis

When the point p_{i+1} is at the center O of the right surface of the unit grid, the distance from p_{i+1} to any point of A, B, C, and D is the same (δ_t), which is called the maximum theoretical error of the spatial straight line, as shown in Figure 8.

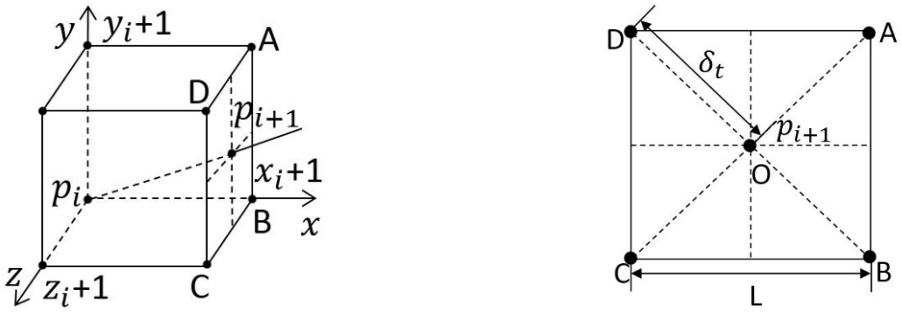


Figure 8. The maximum theoretical error of a spatial straight line. (**a**) A straight line in a unit grid. (**b**) The position of the maximum theoretical error point in the plane.

According to Figure 8, the maximum theoretical error $\delta_t = 0.7071 * L$. Where L is the side length of the unit grid, that is, the step length of each step of each axis—for example, in general industrial machine tools—L represents the moving distance of the actuator driven by a pulse of stepping motor. The step angle of the stepping motor is θ, subdivided is s; the pulse number of one turn of the stepping motor is p_m; and the screw pitch is p, then:

$$p_m = \frac{360}{\theta} * s \tag{12}$$

The stepper motor receives one pulse, and the lead screw movement distance L is

$$L = \frac{P}{p_m} \tag{13}$$

If $\theta = 1.8°$, $s = 2$, $P = 2$ mm, obtain:

L $= 5 * 10^{-3}$ mm, and the maximum theoretical error $\delta_t = 3.54 * 10^{-3}$ mm. It can be seen that the value of L is related to the parameters of the stepping motor and the actuator, and the error distribution of each straight line is as shown in Figure 9.

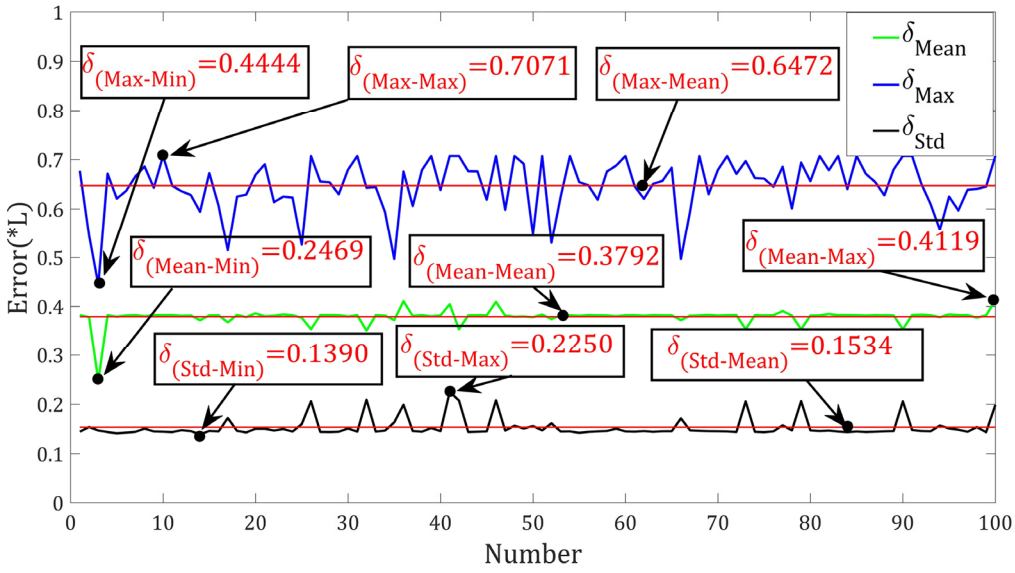
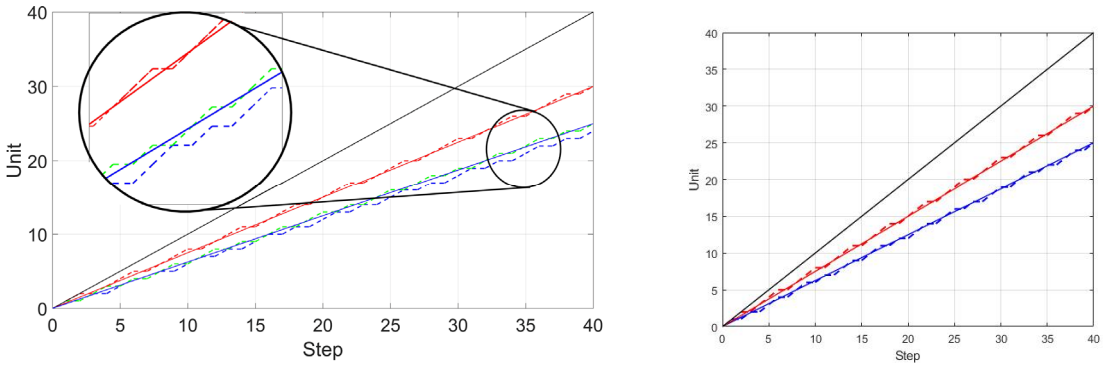


Figure 9. The error distribution.

Let L be 1, and in Figure 9, δ_{Mean} is the average value of error of each straight line in 100 straight lines; this reflects the concentration trend of error of each straight line. $\delta_{(Mean-Max)}$ is the maximum value of δ_{Mean}, which is 0.4119; here, the straight-line error is relatively large. $\delta_{(Mean-Min)}$ is the minimum value of δ_{Mean}, which is 0.2469; here, the straight-line error is relatively small. $\delta_{(Mean-Mean)}$ is the average value of δ_{Mean}, which is 0.3792; this reflects the concentration trend of errors of all straight lines. δ_{Max} is the maximum error of each straight line in 100 straight lines, that is, the straightness error. $\delta_{(Max-Max)}$ is the maximum value of δ_{Max}, which is 0.7071; here, the straightness error is relatively large. $\delta_{(Max-Min)}$ is the minimum value of δ_{Max}, which is 0.7071; here, the straightness error is relatively small. $\delta_{(Max-Mean)}$ is the average value of $\delta_{Max,}$ which is 0.6472; this reflects the concentration trend of the straightness error of the straight line. δ_{Std} is the standard deviation of the error of each straight line in 100 straight lines; this reflects the measures of dispersion of each straight-line error. $\delta_{(Std-Max)}$ is the maximum value of δ_{Std}, which is 0.2250; here, the measures of dispersion are relatively large. $\delta_{(Std-Min)}$ is the maximum value of $\delta_{Std,}$ which is 0.1390; here, the measures of dispersion are relatively small. $\delta_{(Std-Mean)}$ is the average value of $\delta_{Std,}$ which is 0.1534; this reflects the concentration trend of the measures of dispersion of all straight-line errors.

From the above results, all errors are less than or equal to the maximum theoretical error of 0.7071.

In addition, according to the algorithm provided by Liu [1], a random function in MATLAB to set several sets of samples including origin and target points is used. Then, Liu's algorithm and our algorithm are all programmed in MATLAB to generate a straight line with the above-mentioned randomly generated origin and end points. In this way, the ideal line is drawn, and the actual line generated by these two different algorithms are obtained, respectively. Finally, the ideal curve and the actual curve generated by the two algorithms are drawn in one graph, as shown in Figure 10 (green and blue lines represent the results of retaining four and one decimal places, respectively, and the detailed explanation of Figure 10a is in the sixth paragraph of the fourth section).



(**a**) (**b**)

Figure 10. The result of algorithm to generate straight lines. (**a**) Liu's algorithm keeps one decimal place. (**b**) Algorithm in this article.

A specific example is now given to explain the difference. As Liu's paper shows: given that the two end-points of a straight-line segment are specified at positions (0, 0, 0) and (10, 7, 6), it follows that $k_{yx} = 0.7$, and $k_{zx} = 0.6$, then

$$\varepsilon(y_1) = y_1 - y_{0r} - 0.5 = k_{yx} - 0.5 = 0.2 \tag{14}$$

$$\varepsilon(z_1) = z_1 - z_{0r} - 0.5 = k_{zx} - 0.5 = 0.1 \tag{15}$$

From the above two equations, it is possible to calculate the generated pixels and the decision parameters at each inter x position, starting from 0, 0, 0, as listed in Table 1. The error listed in the last column of the table is the distance between the generated pixel and the given line path at each inter x position.

Table 1. Line generation based on Liu's algorithm.

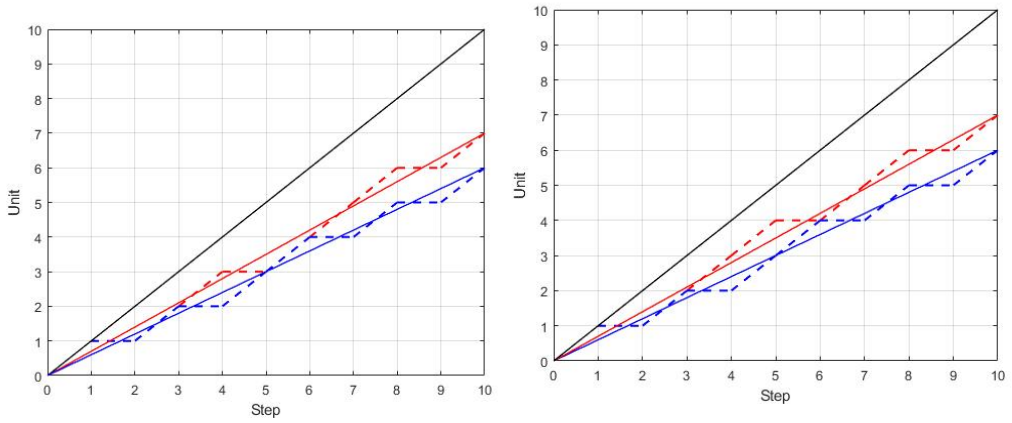
i	x_i	y_{ir}	z_{ir}	$\varepsilon(y_{i+1})$	$\varepsilon(z_{i+1})$	Error
0	0	0	0	0.2	0.1	0
1	1	1	1	−0.1	−0.3	0.374887
2	2	1	1	0.6	0.3	0.336918
3	3	2	2	0.3	−0.1	0.220564
4	4	3	2	0	0.5	0.441129
5	5	4	3	−0.3	0.1	0.428700
6	6	4	4	0.4	−0.3	0.441129
7	7	5	4	0.1	0.3	0.225064
8	8	6	5	−0.2	0.1	0.336918
9	9	6	5	0.5	0.5	0.374887
10	10	7	6	0.2	0.1	0

Bringing this case into the algorithm in this article can obtain the data in Table 2.

Table 2. Line generation based on this article's algorithm.

i	x_i	y_i	z_i	M_i	N_i	Error
0	0	0	0	0	0	0
1	1	1	1	4	2	0.374887
2	2	1	1	−2	−6	0.336918
3	3	2	2	12	6	0.220564
4	4	3	2	6	−2	0.441129
5	5	4	3	0	18	0.428700
6	6	4	4	−6	10	0.441129
7	7	5	5	8	2	0.220564
8	8	6	5	2	−6	0.336918
9	9	6	6	−4	6	0.334534
10	10	7	6	10	−2	0

In addition, the difference between the seventh and ninth steps of the two methods can be clearly seen below in Tables 1 and 2, and Figure 11. When step 9 is performed, the error is different, which can be seen from Tables 1 and 2, respectively, and the error is sightly less than that in the literature.



(**a**) (**b**)

Figure 11. Differences between two different algorithms. (**a**) Line generation based on Liu's algorithm (**b**) Line generation based on this article's algorithm.

3.3. Conversion of Unit Grid Number and Pulse Number

Step motor is an open-loop control motor which transforms electric pulse signal into angular displacement or linear displacement [23]. When a pulse is received, it will drive the step motor to rotate a fixed angle (step angle θ) according to the set direction. The algorithm in this paper can be used for controlling three stepping motors cooperatively. In the research of Dulf [24,25] et al., a new method for designing a fractional-order controller and a method for designing a DC motor speed control fractional-order PI controller are proposed, and the method is proved through an example analysis. The feasibility and superiority of the fractional-order controller can greatly reduce the overshoot and make its performance better than the traditional Kessler controller. Through specific experiments, its performance is evaluated and compared to other types of implementation possibilities. Three stepper motors control the x-axis, y-axis, and z-axis, respectively; the three-dimensional space described in the algorithm is the working space of three stepping motors; and the unit is a pulse. Δx, Δy, and Δz are the number of pulses required to reach the target point for the x motor, y motor, and z motor, respectively. The number of forward steps mentioned in this paper is the number of pulses required for the axis with the largest moving distance.

In industrial machine tools, the three coordinate axes are all driven by lead screw, and the lead screw pitch is P mm. Three axes are set to move x mm, y mm, z mm, respectively; then, the step motor turns a circle while lead screw moves P mm(one pitch), and three motors rotate $\frac{x}{P}$ circle, $\frac{y}{P}$ circle, and $\frac{z}{P}$ circle, respectively, to reach the target point. The step angle θ of all three stepper motors is 1.8°, and the subdivision s is 8. According to Equation (12), 1600 pulses are needed for one turn of the stepper motor. The number of pulses of the three stepper motors for reaching the target point are $(1600 \times \frac{x}{P})$, $(1600 \times \frac{y}{P})$, $(1600 \times \frac{z}{P})$, respectively.

3.4. Algorithm Extension

The algorithm in this paper can be extended to five dimensions; assuming that the other two axes are A and B, respectively, the corresponding discriminant is H and K, respectively. According to Equation (8), the H and K are obtained:

$$\begin{cases} H_{i+1} = H_i + 2\Delta A - 2\Delta x(A_{i+1} - A_i) \\ K_{i+1} = K_i + 2\Delta B - 2\Delta x(B_{i+1} - B_i) \end{cases} \tag{16}$$

Considering Equation (10), the initial value of the discriminant H and K are obtained as:

$$\begin{cases} H_1 = 2\Delta A - \Delta x \\ K_1 = 2\Delta B - \Delta x \end{cases} \tag{17}$$

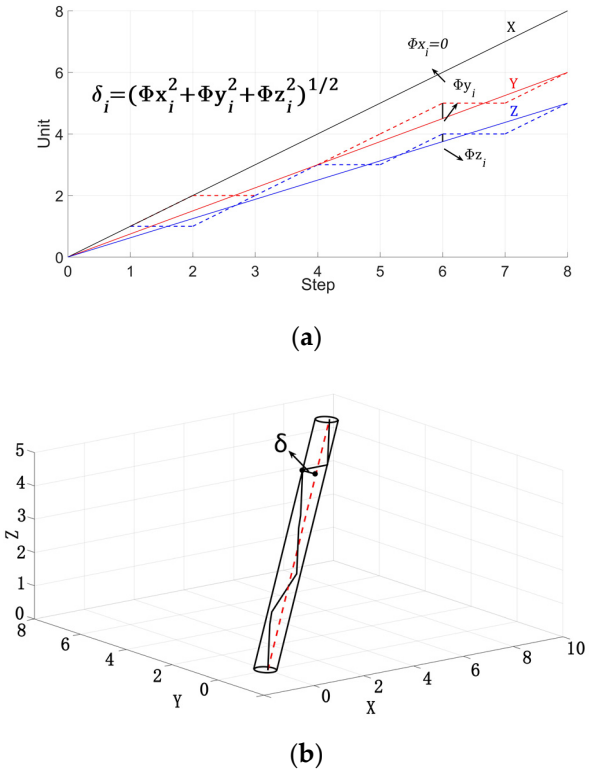
According to the above rules, when $H_{i+1} > 0$, $A_{i+1} = A_i + 1$, and when $K_{i+1} > 0$, $B_{i+1} = B_i + 1$. Therefore, the algorithm in this paper can be extended to five or more dimensions, which means that more motors can be controlled cooperatively by this algorithm in this paper.

4. Discussion

Matlab Simulation Discussion

The type ① is discussed as one example of the algorithm in this paper; that is, the end point of the straight line is on the right surface of the cube, and algorithm 1 is designed. The x axis is taken as the active direction, and progresses one unit at a time; y and z are taken as the passive direction; by judging the sign of discriminant M_i and N_i, which regions of the right surface of the unit grid the point is located in will be determined, and then whether or not to progress one unit will be determined. There are seven types of spatial line distribution; for the other six types, the end position of the straight line is different, and the variables (Δx, Δy, and Δz) are different, but the algorithm structure is the same: only the corresponding variables need to be exchanged. In addition, in this paper, the Δx, Δy, and Δz in the discriminant of the algorithm possess a proportion relation. For example, if $\Delta x = 8000$, $\Delta y = 6000$, and $\Delta z = 5000$, the corresponding step size of 8000, and there is the greatest common factor (C_f), that is 1000, then $\Delta x^* = \frac{\Delta x}{C_f} = 8$, $\Delta y^* = \frac{\Delta y}{C_f} = 6$, and $\Delta z^* = \frac{\Delta z}{C_f} = 5$; the Δx^*, Δy^*, and Δz^* are taken as variables of discriminants; moreover, the discriminants are all integers. Therefore, the proposed algorithm could be simplified to calculate for drawing spatial straight lines. The 100 groups of data are randomly generated by MATLAB, and the algorithm is verified in this paper. It is proved that the algorithm is suitable for straight lines in any direction.

According to the algorithm verification results in Section 3, several sets of data generated by MATLAB are brought into the three-dimensional algorithm of Liu [1], and the results are shown in Figure 12.



(**a**)

(**b**)

Figure 12. Results of the spatial straight-line algorithm. (**a**) The motion process of each axis. (**b**) Spatial straight line and straightness error.

It can be seen from the results of the comparison algorithm that this algorithm produced the same results as the algorithm in this paper; meanwhile, the same case is brought into Dai [2], in which also the same results occurred. It is noted that the selection points for the *y* and *z*-axis in the comparison algorithm are judged separately in the XY and XZ planes. In addition, the discriminant contains decimals in the reference algorithm; however, there are not the above two phenomena in the algorithm of this paper.

Because of the retention number of significant digits and the gradual accumulation of the discriminant the positive and negative signs of the discriminant may be changed, the selective point also may be altered. Furthermore, there is a greater deviation of the drawn straight line from the theoretical straight line, which increases the straightness error. In order to verify the possibility of the above problems, the discriminant of different decimal places (four, three, two, and one decimals) is calculated by the comparison algorithm, after testing; when the discriminant retains one decimal, the straight line will deviate from its ideal straight line, as shown in Figure 10a.

In the Figure 10a, B is enlarged pictures of A; the black and red lines represent the *x* and *y*-axis, respectively; green and blue lines represent the results of retaining four and one decimal places, respectively. It can be seen in the figure that the z-axis deviates from the theoretical straight line with the reserved one decimal place: at this time, the straightness error δ is greater than the maximum theoretical error of 0.7071. The z-axis oscillates on the theoretical straight line with the reserved four decimal places.

Additionally, the following conclusions have been reached through multiple verifications; the Liu [1] algorithm will not deviate from the theoretical line when more than two decimals are reserved, and may deviate from the theoretical line when one decimal is reserved.

In this paper, according to a given rule (such as a pixel or a step angle), three-dimensional space is divided into innumerable three-dimensional meshes, namely unit grids. The right surface of each unit grid is divided into four regions, and two discriminants are determined at the same time to determine the selection point. One point is selected each time: this point would be the origin of the

coordinate system of the next unit grid, and also the starting point of the straight line in this unit grid. The discriminants of the algorithm in this paper are all integers; that the problem of deviation from the theoretical straight line caused by the retention of decimals of significant digits can be avoided. Some other literatures also proposed some new algorithms or related error analysis: they verify the effectiveness and advantages of the algorithm through comparison and analysis with other related literatures [26,27]. In this article, the effectiveness and advantages of our algorithm are shown through MATLAB simulation, comparison, and analysis with other related literature. There are still some shortcomings in this algorithm. A large number of samples are needed for comparative analysis to verify whether the computational efficiency of this algorithm is improved; at present, this algorithm is only suitable for the space straight line, and not for the space curve.

5. Conclusions

First, the Bresenham's algorithm is extended to three-dimensional space in this paper. Bresenham algorithm is two-dimensional. To draw three-dimensional space straight lines, a novelty method of discriminate regions is proposed in this paper. Based on the discriminant, the selection points for each step will be provided. Only the coordinates of the starting and ending points of the spatial straight line are needed to be input in this algorithm, and the discriminants of the region can be quickly determined. All the discriminants are integers, making the calculations simple to quickly draw the spatial straight line; and the problem of deviation from the theoretical straight line caused by the retention of decimals of significant digits can be avoided. By analyzing the error of the straight line, the result shows that the error of the algorithm is related to the step length L. The maximum theoretical error is $\frac{\sqrt{2}}{2} \times \mathrm{L}$, where $\mathrm{L} = \mathrm{P}/\mathrm{p}_m$. In addition, in this paper, the Δx, Δy, and Δz in the discriminant of the algorithm possess a proportion relation, which could be replaced by the simplest integer. Moreover, the discriminants of this paper are all integers; therefore, the proposed algorithm in this paper could be used to calculate drawing both short and long straight lines more simply than the algorithm of decimals of other discriminants, while the accuracy is guaranteed. Meanwhile, the algorithm's conversion application between unit grid number of algorithm and pulse of motors is performed, and the algorithm in this paper can be used for controlling three stepper motors cooperatively or be extended to five dimensions.

Author Contributions: Conceptualization, J.W.; methodology, S.X.; software, J.Y.; validation, S.X., J.Y.; formal analysis, S.X.; investigation, J.W., Y.L.; resources, Y.L.; data curation, S.X.; writing—original draft preparation, S.X.; writing—review and editing, P.B.; visualization, T.S.; supervision, Y.L., T.S.; project administration, Y.L.; funding acquisition, J.W. All authors have read and agreed to the published version of the manuscript.

Funding: This investigation is supported by the National Natural Science Foundation of China (31370999). (Information for disclosures can be taken from the online abstract system after entering all authors).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Liu, X.W.; Cheng, K. Three-dimensional extension of Bresenham's algorithm and its application in straight-line interpolation. *Proc. Inst. Mech. Eng. Part B J. Eng. Manuf.* **2002**, *216*, 459–463.
2. Dai, M.; Chen, Y.; Zheng, C.; Yiming, G. Design of multistep stepper motor coordinated control system based on bresenham algorithm. In Proceedings of the 2017 24th International Conference on Mechatronics and Machine Vision in Practice (M2VIP), Auckland, New Zealand, 21–23 November 2017.
3. Chen, Z.; Shen, Z.; Guo, J.; Cao, J.; Zeng, X. Line drawing for 3D printing. *Comput. Graph.* **2017**, *66*, 85–92.
4. Meng, Q.; Geng, G.; Zhang, J. A line drawing method of the 3D models based on partial sharpening. In Proceedings of the 2012 International Conference on Image Analysis and Signal Processing, Zhejiang, China, 9–11 November 2012; pp. 9–11.

5. Guo, T.; Wang, Y.; Zhou, Y.; He, Z.; Tang, Z. Geometric object 3D reconstruction from single line drawing image with Bottom-Up and Top-Down classification and sketch generation. In Proceedings of the 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Kyoto, Japan, 9–15 November 2017; pp. 670–676.
6. Skala, V. An intersecting modification to the bresenham algorithm. *Comput. Graph. Forum* **1987**, *6*, 343–347.
7. Bresenham, J.E. Algorithm for computer control of a digital plotter. *IBM Syst. J.* **1965**, *4*, 25–30.
8. Nguyen, D.T. A rotation method for binary document images using DDA algorithm. *ACM Symp. Doc. Eng.* **2008**, 267–270. [CrossRef]
9. Govil-Pai, S. *Principles of Computer Graphics: Theory and Practice Using OpenGL and Maya*, 1st ed.; Springer: Berlin/Heidelberg, Germany, 2010.
10. Cieśliński, J.L.; Moroz, L.V.; Walczyk, C.J. Fast exact digital differential analyzer for circle generation. *Appl. Math. Comput.* **2015**, *271*, 68–79.
11. Lau, W.-C.; Erramilli, A.; Wang, J.; Willinger, W. Self-similar traffic generation: The random midpoint displacement algorithm and its properties. In Proceedings of the IEEE International Conference on Communications, Seattle, WA, USA, 6 August 2002.
12. Jilesen, J.; Kuo, J.; Lien, F.-S. Three-dimensional midpoint displacement algorithm for the generation of fractal porous media. *Comput. Geosci.* **2012**, *46*, 164–173.
13. Garcia-Falset, J.; Marino, G.; Zaccone, R. An explicit midpoint algorithm in Banach spaces. *J. Nonlinear Convex Anal.* **2017**, *18*, 1933–1952.
14. Au, C.; Woo, T. Three dimensional extension of Bresenham's Algorithm with Voronoi diagram. *Comput. Des.* **2011**, *43*, 417–426.
15. Stephenson, P.; Litow, B. Running the line: Line drawing using runs and runs of runs. *Comput. Graph.* **2001**, *25*, 681–690.
16. Li, X.; Shao, X. Fast line drawing algorithm by circular subtraction based on Bresenham. In Proceedings of the SPIE—The International Society for Optical Engineering, Bellingham, WA, USA, 13 January 2012; Volume 8349, p. 20.
17. Muresan, C.; Copot, C.; Birs, I.; De Keyser, R.; Vanlanduit, S.; Ionescu, C.M. Experimental Validation of a Novel Auto-Tuning Method for a Fractional Order PI Controller on an UR10 Robot. *Algorithms* **2018**, *11*, 95. [CrossRef]
18. Pitteway, M.L.V.; Watkinson, D.J. Bresenham's algorithm with Grey scale. *Commun. ACM* **1980**, *23*, 625–626. [CrossRef]
19. Labatut, V. Continuous average straightness in spatial graphs. *J. Complex Netw.* **2017**, *6*, 269–296. [CrossRef]
20. Zhang, Q.; Fan, K.-C.; Li, Z. Evaluation method for spatial straightness errors based on minimum zone condition. *Precis. Eng.* **1999**, *23*, 264–272. [CrossRef]
21. Endrias, D.H.; Feng, H.-Y.; Ma, J.; Wang, L.; Abu Taher, M. A combinatorial optimization approach for evaluating minimum-zone spatial straightness errors. *Measurement* **2012**, *45*, 1170–1179. [CrossRef]
22. Wen, X.; Song, A. An improved genetic algorithm for planar and spatial straightness error evaluation. *Int. J. Mach. Tools Manuf.* **2003**, *43*, 1157–1162. [CrossRef]
23. Li, J.; Meng, D. Dynamic and Adjustable New Pattern Four-Vector SVPWM Algorithm for Application in a Five-Phase Induction Motor. *Energies* **2020**, *13*, 1856. [CrossRef]
24. Dulf, E.-H.; Susca, M.; Kovács, L. Novel Optimum Magnitude Based Fractional Order Controller Design Method. *IFAC-PapersOnLine* **2018**, *51*, 912–917. [CrossRef]
25. Muresan, C.I.; Folea, S.; Mois, G.; Dulf, E. Development and implementation of an FPGA based fractional order controller for a DC motor. *Mechatronics* **2013**, *23*, 798–804. [CrossRef]
26. Hassan, T.-U.; Abbassi, R.; Jerbi, H.; Mehmood, K.; Tahir, M.F.; Cheema, K.M.; Elavarasan, R.M.; Ali, F.; Khan, I. A Novel Algorithm for MPPT of an Isolated PV System Using Push Pull Converter with Fuzzy Logic Controller. *Energies* **2020**, *13*, 4007. [CrossRef]
27. Du, L.; Ma, Q.; Ben, J.; Wang, R.; Li, J. Duality and Dimensionality Reduction Discrete Line Generation Algorithm for a Triangular Grid. *ISPRS Int. J. Geo-Inf.* **2018**, *7*, 391. [CrossRef]