Research on Intelligent Design of Geometric Factor Encoding for Aircraft Engine Turbine Structures

Wencong Xu, Hongyi Lu *, Lei Zhao and Borui He

School of Aircraft Engineering, Nanchang Hangkong University, Nanchang 330063, China; 1527911267@163.com (W.X.)
* Correspondence: 13964508115@163.com

Abstract: In recent years, with the rapid development of computer technology and artificial intelligence design technology, multiple possible design solutions can be quickly generated by transforming the experience and knowledge of structural design into computer executable rules and algorithms. To achieve intelligent design of aircraft engines, this paper proposes an encoding model for the turbine rotor structure of aircraft engines using geometric encoding technology. The turbine rotor structure of aircraft engines is divided into several units according to geometric similarity types, these units continue to be divided into attribute sets according to their functional types, connection relationships, and material properties. These attribute sets can be encoded using geometric encoding technology. The experiment simulated that these codes, for the point cloud modeling of turbine rotor structure, can be quickly achieved and they combine various algorithms to display the point cloud model of the turbine rotor in the Microsoft Visual studio MFC class library. The results show that by creating geometric codes for the turbine rotor of aircraft engines, it is possible to quickly create and display point cloud models of the turbine rotor structure, laying the foundation for subsequent application of machine learning to solve and find the optimal design solution.

Keywords: aircraft engines; turbine structures; intelligent design; geometric coding technology

1. Introduction

With the rapid development of computer technology, traditional product design can no longer meet the personalized design of technology as the object of specific customers or customer groups. The inconsistency between the newly adopted advanced production manufacturing process and traditional design methods is a bottleneck in design. Therefore, in manufacturing production, design becomes the largest direct or indirect cost area, and researching intelligent design of digital products has become an urgent problem to be solved [1–4]. The same is true in the aerospace field.

Aircraft engines are a key product in the development of the aviation industry, and the quality of engines directly determines the performance of aircraft. Moreover, the progress of engine development is also a decisive factor in the overall progress of aircraft development [5,6]. In the field of aviation engine turbocharging, the turbine operates at high temperatures and speeds, and the working conditions are extremely demanding, requiring a long lifespan and high reliability, while the quality is strictly controlled. Therefore, structural design is very difficult. In the structural design stage, static strength, stiffness, vibration, and rotor dynamics analysis should be conducted on the blades, discs, and shafts to determine the main dimensions of the structure. With a preliminary structural concept diagram, strength calculations need to be carried out to compare the stress levels, stress distribution, and stiffness of different structures, to select and optimize structural design. In this design process, it is always necessary to integrate, analyze, and compromise with other factors. The design process is often repetitive, and the corresponding strength calculation also needs to be repeated [7,8]; this makes the design phase time-consuming.
One of the main problems of the aircraft engine is the high costs at all stages of the design and production. Reducing costs and improving competitiveness essentially begin with the accurate and efficient estimate of the resource for the manufacture of the products, including labor and material costs [9–12].

Intelligent design, as an important aspect of the field of artificial intelligence, applies artificial intelligence to product design and is a design method for achieving automated decision-making. In traditional design methods, the decision-making tasks of product design are often completed by humans. If you want to use computers to assist decision-making, you need to find ways to use computers to automatically process various types of knowledge. The main characteristics of intelligent design methods are based on design methodology theory, utilizing 3D graphics software, intelligent design software, virtual reality technology, as well as multimedia and hypermedia tools to develop and design products, express product concepts, and describe product structures [13]. Intelligent design began with the research work of Wesley and Takeyama in the early 1980s, and it was not until the 1990s that the research on product intelligent design and its theory received widespread attention and discussion. The NATO Science and Technology Organization classified artificial intelligence as one of the eight major technological fields that can have a disruptive impact on the world in its “Science and Technology Trends: 2020–2040” released in March 2020 [14].

Geometric encoding technology is a technique that converts the three-dimensional information of an object into a computer-readable encoding format. By dividing the three-dimensional object into small units and encoding each unit, the computer can quickly recognize the position, shape, size, and other information concerning the object. Geometric encoding is often applied in CAD software, virtual reality technology, machine vision, and other fields [15].

Therefore, by transforming the experience and knowledge of aircraft engine turbine structure design into computer-executable rules and algorithms, multiple possible design schemes can be quickly generated, and the optimal design scheme can be selected through optimization algorithms. Before conducting intelligent structural design, it is necessary to establish a mathematical model for structural design. Generally, the model is created by the training of existing structural design cases and experiences. During the creation process, these basic models need to be encoded, and then learning can be used to optimize and adjust some of the parameters to achieve the optimization and design of the structural model. This article studies the encoding of the basic model of aircraft engine turbine rotors.

2. Models

The geometric coding of the structure of engine turbine rotors has been inspired by biological evolution and genetic composition. Thousands of organisms in nature are ultimately formed through mechanisms such as selection, crossover, and variation during the evolution of life. Their appearances and functions are diverse, and these appearances and functions are expressed by the internal genes of organisms. Genes are composed of many DNA fragments connected in a fixed manner, and DNA is ultimately composed of four bases, A, G, C, and T, in different ways [16]. Geometric coding parts, basic features with different functions and structures, can be used, such as circular protrusions, rectangular protrusions, center holes, rectangular grooves, and chamfers. Just like nucleotides, although in small quantities, they can form parts with different structures and functions through different combinations. Finally, these parts can be selected, crossed, and mutated to obtain more optimized parts. The steps to apply this method to the turbine rotor of an aircraft engine are as follows: The turbine rotor is divided into unit bodies, and then attribute sets are planned for the functions and constraint types of these unit bodies. Finally, these attribute sets are encoded. Their similarity comparison is shown in Figure 1.
Figure 1. Comparison of similarities between biological tissue systems and turbine rotor systems.

2.1. Geometric Coding Model for Turbine Rotor Structure

2.1.1. Turbine Rotor Structural Unit

The structural unit of the turbine rotor can be represented as Equation (1):

\[ U_n = (G_{e1} - G_{e2} - \cdots - G_{em}, G_{cs}) = \begin{bmatrix} G_{e1} - G_{e2} - \cdots - G_{em} \\ G_{c1} \\ G_{c2} \\ \vdots \\ G_{cm} \end{bmatrix} \]  

(1)

where \( G_{e1}, G_{e2}, \ldots, G_{em} \) is a type attribute set of \( U_n \), which describes the basic structure of the engine turbine rotor structure unit. The \( U_n \) can be named in the form of \( G_{e1} - G_{e2} - \cdots - G_{em} \) such as turbine disc, turbine blade, turbine shaft, connection between turbine disk and turbine blades, etc.; \( G_{cs} \) is a matrix of connected attribute sets, which contains all the connection methods and their combination sequences that make up the turbine rotor of an engine.

Structural units directly connected by geometric factors can transfer energy through pairs of connections. As shown in Figure 2, unit 1 is formed by the ordered combination of connection I, II, III, and rotor disk geometries, which determine the shape and size of unit 1. When unit 2 works, energy is transferred from unit 2 to unit 1 through connection III, and then from unit 1 to unit 3 through connection I, so that unit 3 works to achieve the purpose of energy transfer.

Figure 2. Turbine rotor structural units.
A geometric factor usually controls the synthesis of a pair of connections, and some complex connections can also be controlled by several geometric factors.

Figure 2 shows the connected attribute sets making up units. I is the connection between the turbine disk and turbine blades; II is the connection between the turbine disc and turbine shaft; III is the pin connection between the turbine disc and turbine shaft.

2.1.2. Property Family of Turbine Rotor Structural Unit

The function and properties of the engine rotor structure are determined by the element property family. The unit property family is composed of a type property set, a connection property set, and a material property set. The following formulae represent the relationship between them.

The unit property family can be expressed as Equation (2):

\[
Ge = Ge_t \cup Ge_c \cup Ge_m
\]  

(2)

where \( Ge_t \) is the type property set; \( Ge_c \) is the connection property set; \( Ge_m \) is the material property set.

The set of type attributes can be represented as Equation (3):

\[
Ge_t = (F_{c_t}, Un)
\]  

(3)

where \( F_{c_t} \) is the geometric factor that controls the synthesis of \( Ge_t \), the functional type of \( F_{c_t} \) can be named \( Ge_t \), and \( Un \) is the unit composed of \( Ge_t \).

The set of connection properties can be represented as Equation (4):

\[
Ge_c = (Un_1 - Un_2, F_{c_c}, At) = (Un_1 - Un_2, F_{c_c}, [Am, Va])
\]  

(4)

where \( Un_1 - Un_2 \) is used to name \( Ge_c \), and the set of connection attributes that make up unit 2 can be named unit 2-unit 1; \( F_{c_c} \) is a geometric factor to control the synthesis of \( Ge_c \), \( At \) is the constraint attribute of cell body; \( Am \) and \( Va \) are the attribute quantity set and attribute value set of \( At \), whose definition is the same as the constraint attribute of the geometric factors.

The set of material properties can be expressed as Equation (5):

\[
Ge_m = (F_{c_m}, Un)
\]  

(5)

where \( F_{c_m} \) is the geometric factor that controls the synthesis \( Ge_m \), \( Ge_m \) can be named by the functional type of \( F_{c_m} \), and \( Un \) is the unit composed of \( Ge_m \).

2.2. Geometric Coding Model of Turbine Rotor Unit

2.2.1. Geometric Encoding of Shaft Unit

The biggest characteristic of a shaft class is its length, as well as the type of shaft. Therefore, the main coding is the length control factor and the type of geometry of the shaft; the type of geometry of the shaft determines whether the shaft is a solid shaft or a hollow shaft, a straight shaft or a conical shaft. According to the type of the shaft, it can be further coded and divided to determine the inner and outer diameter coding of the shaft and the slope coding of the shaft. According to the coordination of the shaft and other parts, the position code needs to be added or the shape code of the slot holes needs to be added to the keyway on the shaft, and finally the complete shaft unit body is formed.

The proposed shaft model is coded as Equation (6):

\[
[Z_0, Z_1, R, r, d, \alpha, T]
\]  

(6)

where \( Z_0 \) is the initial position of the shaft growth starting point; \( R \) is the outside diameter of the shaft; \( r \) is the inner diameter of the shaft (\( r = 0 \), is the real shaft); \( d \) is the shaft length;
\( \alpha \) is the cone angle of the shaft; \( T \) is the type of connected geometry on the shaft. \( Z_0, Z_1 \) are position constraint control factors.

### 2.2.2. Geometric Encoding of Disk Unit

The disk parts have the characteristics of a large radius and small thickness. The appearance of a slot on the edge of the disk and a hole in the disk can be regarded as the result of biological evolution in nature. The result of its evolution is that the geometric factor of the receiving disk structure can be coded and controlled, and the type of mortise on the disk, the inner and outer diameter of the disk, and the thickness of the disk can be controlled.

The proposed disk model is coded as Equation (7):

\[
[T_m, T_d, Z_0, R, r, d, nz, \alpha_1]
\]  

(7)

Including Equations (8) and (9) in Equation (7):

\[
T_m = \{ ng, Fa, Fb, Z_d, \beta_2 \}
\]  

(8)

\[
T_d = \{ \alpha_3, r_1, r_2 \}
\]  

(9)

where \( T_m \) is the mortise code of the disc parts; \( nz \) is the number of mortise; \( \alpha_1 \) is the torsion angle of the disc; \( T_m \) is the fir-tree mortise code, and its internal parameters \( ng \) are the number of fir-tree mortise teeth; \( Fa \) and \( Fb \) are the arithmetic factors of the fir-tree mortise; \( Z_d \) is the position function of symmetric line of the fir-tree mortise; \( \beta_2 \) the inclination angle of the fir-tree mortise; \( T_d \) is the type of disk (straight disk, cone disk, and edge ladder cone disk); \( Z_0 \) is the initial position of the disk growth starting point; \( R \) is the outer diameter of the disk; \( r \) is the inner diameter of the disk; \( d \) is the thickness of the disk. In the special disk type code, \( \alpha_3 \) is the bevel of the cone disk, \( r_1 \) is the radius of the flange closest to the edge of \( R \), and \( r_2 \) is the radius of the flange closest to the edge of \( r \).

### 2.2.3. Geometric Encoding of Blade Unit

Because the rotor blade needs to fit with the mortise on the disk, there are many geometric factors, and the shape and function control factors are mainly the parameter coding of the blade body and the parameter coding of the top of the tenon.

The proposed disk model is coded as Equation (10):

\[
[Z_0, R, d_s, nz, T_b, s_1, s_2, H_b, m, k, t, c, Fd, d_b, \alpha_4, \alpha_5]
\]  

(10)

where \( Z_0 \) is the initial position of the blade starting point; \( R \) is the position constraint control factor of tenon length; \( d_s \) is the thickness of the tenon and \( nz \) is the number of rotor blades; \( T_b \) is the type of rotor blade, which is similar to the mortise code of the disk model; \( s_1 \) is the protruding length of the top of the tenon; \( s_2 \) is the top width of the tenon; \( H_b \) is the blade height; \( m, k, t \) and \( c \) are blade shape parameters; \( Fd \) is the blade shape amplification factor; \( d_b \) is the blade thickness; \( \alpha_4 \) is the mortise twist angle; \( \alpha_5 \) is the blade body twist angle.

### 3. Methods

#### 3.1. Modeling Algorithm for Rotor Shaft Unit Point Cloud Model

#### 3.1.1. The Type Attribute Set of the Shaft

1. Two-dimensional point cloud modeling algorithm for shafts.

Two-dimensional point cloud data of the shaft model can be obtained according to \([R, r]\) in the axial geometry coding model. The algorithms are as Equations (11) and (12).

\[
\begin{align*}
B_{xi} &= r \cdot \cos c \\
B_{yi} &= r \cdot \sin c
\end{align*}
\]  

(11)
where \( B \) is the shaft of the boundary data storage array, \( i \) is 0, 1, 2, \ldots, \( m \); \( j \) is \( n + 1, n + 2, \ldots, n + m \); array holds the \( x \), \( y \) coordinates and RGB color values of the shaft boundaries; \( c \) is an angular variable in increments of 0.1 from 0 to 360 degrees.

The scan-line filling algorithm is based on the region fill algorithm, which is based on the vector form data of polygonal borders and can be used for program fill or interactive fill. The specific implementation methods in MFC are as follows: CreatePolygonRgn function is used to determine the polygon boundary region through scanning line filling, PtInRegion function is used to scan and judge the data in the boundary region, and then assign values according to the definition. The point cloud data are saved to the array \( D[D_x, D_y, D_{rgb}] \), which stores the coordinate information of \( x \) and \( y \) of the two-dimensional cross-section of the shaft, as well as the defined pixel color information.

2. Three-dimensional point cloud modeling algorithm for shafts.

According to \( \{d, \alpha\} \) of the control factors in the type of geometry, a three-dimensional point cloud array \( T \) with basic shaft can be obtained. The two-dimensional point cloud data on the constant section of the direct shaft are basically unchanged, and only the control factor \( d \) controlling the length of the shaft is available. The algorithm is as Equation (13):

\[
\begin{align*}
B_{xi} &= R \cdot \cos c \\
B_{yi} &= R \cdot \sin c \\
B_{zi} &= \alpha \\
B_{yi+1} &= B_{yi} + \frac{d}{\tan \alpha} \\
B_{zi+1} &= B_{zi} + \delta
\end{align*}
\]  

(12)

where \( B \) is the shaft of the boundary data storage array, \( i \) is 0, 1, 2, \ldots, \( n \); \( j \) is \( n + 1, n + 2, \ldots, n + m \); array holds the \( x \), \( y \) coordinates and RGB color values of the shaft boundaries; \( c \) is an angular variable in increments of 0.1 from 0 to 360 degrees.

The scan-line filling algorithm is based on the region fill algorithm, which is based on the vector form data of polygonal borders and can be used for program fill or interactive fill. The specific implementation methods in MFC are as follows: CreatePolygonRgn function is used to determine the polygon boundary region through scanning line filling, PtInRegion function is used to scan and judge the data in the boundary region, and then assign values according to the definition. The point cloud data are saved to the array \( D[D_x, D_y, D_{rgb}] \), which stores the coordinate information of \( x \) and \( y \) of the two-dimensional cross-section of the shaft, as well as the defined pixel color information.

2. Three-dimensional point cloud modeling algorithm for shafts.

According to \( \{d, \alpha\} \) of the control factors in the type of geometry, a three-dimensional point cloud array \( T \) with basic shaft can be obtained. The two-dimensional point cloud data on the constant section of the direct shaft are basically unchanged, and only the control factor \( d \) controlling the length of the shaft is available. The algorithm is as Equation (13):

\[
\begin{align*}
T_{xi+1} &= T_{xi} \\
T_{yi+1} &= T_{yi} \\
T_{zi+1} &= T_{zi} + \delta
\end{align*}
\]  

(13)

when \( i \) takes 0, 1, 2, \ldots, \( n \), \( T_{zn} - T_{z0} = d \). where \( \delta \) is the increment in the direction of the \( z \) axis and \( T_{zn} \) is the last \( z \) coordinate value of \( T \).

The cone shaft has a similar cross-section in the \( xy \) plane and is reduced or enlarged in the \( z \)-shaf direction. The formation of the cone shaft depends on the control factor \( \alpha \) controlling the cone angle and the control factor \( d \) controlling the shaft length. The algorithm is shown in the following Equation (14):

\[
\begin{align*}
T_{xi+1} &= \left( \sqrt{T_{xi}^2 + T_{yi}^2 + \left( \frac{d}{\tan \alpha} \right)^2} \right) \cdot \cos \left( \arctan \left( \frac{T_{yi}}{T_{xi}} \right) \right) \\
T_{yi+1} &= \left( \sqrt{T_{xi}^2 + T_{yi}^2 + \left( \frac{d}{\tan \alpha} \right)^2} \right) \cdot \sin \left( \arctan \left( \frac{T_{yi}}{T_{xi}} \right) \right) \\
T_{zi+1} &= T_{zi} + \delta
\end{align*}
\]  

(14)

Through the above algorithm, the basic three-dimensional point cloud data \( T[T_x, T_y, T_{rgb}] \) of the shaft, are obtained, as shown in Figure 3.

(a) direct shaft  
(b) cone shaft

Figure 3. 3D point cloud model diagram of shaft.
3.1.2. The Connection Attribute Set of the Shaft

The connecting attribute set of the shaft includes key holes, shaft shoulders, and pin holes. Geometric factors need to be added to model key holes, shoulder, and pin holes. The geometric factor of the keyway is \( [Z_{01}, b, L, h_z, nz] \) and the geometric factor of the pin is \( [Z_{02}, r_z, h_z, nz] \). The geometric factor of material geometry is \( [E, a, \rho] \).


According to the keyway geometric factors \( [Z_{01}, b, L, h_z, nz] \), a three-dimensional point cloud model T1 of the keyway can be created. \( Z_{01} \) is the position constraint geometric factor of the axial slot, \( b \) is the width of the keyway, and \( L \) is the length of the keyway. These control factors control the basic shape of the keyway as shown in Figure 4; \( h_z \) controls the depth of the keyway, and \( nz \) can control the number of the keyways in the shaft.

![Figure 4. A-type keyway.](image-url)

It is necessary to design the algorithm of the arc segment and the straight segment and store the fourth boundary data for the 2D boundary modeling of the keyway.

The algorithm for the first arc is as Equation (15):

\[
\begin{align*}
B_{1x1} &= b \cdot \cos c \\
B_{1y1} &= \frac{b}{2} \cdot \sin c
\end{align*}
\]  

where \( c \) is an angle variable with an increment of 0.1 from 90 to 270 degrees, and \( i \) is 0, 1, 2, \ldots, \( n \).

For the second straight line, the algorithm is as Equation (16):

\[
\begin{align*}
B_{1xj} &= -\delta \\
B_{1yj} &= -\frac{b}{2}
\end{align*}
\]  

where \( \delta \) is the length variable from 0 to \( L - b \) with increments of 0.1, \( j \) is \( n + 1, n + 2, \ldots, n + m \).

The algorithm for the third arc is as Equation (17):

\[
\begin{align*}
B_{1xk} &= (b - L) + \frac{b}{2} \cdot \cos c \\
B_{1yk} &= \frac{b}{2} \cdot \sin c
\end{align*}
\]  

where \( c \) is an angular variable with an increment of 0.1 from 270 to 360 degrees, and \( k \) is \( n + m + 1, n + m + 2, \ldots, n + m + o \).

For the fourth straight line, the algorithm is as Equation (18):

\[
\begin{align*}
B_{1xl} &= b - L + \delta \\
B_{1yl} &= \frac{b}{2}
\end{align*}
\]  

where \( \delta \) is the length variable from 0 to \( L - b \) with increments of 0.1, \( l \) is \( n + m + o + 1, n + m + o + 2, \ldots, n + m + o + p \).
After obtaining the point cloud data of the boundary of the keyway, the internal data are filled. Similar to the internal fill of the shaft, CreatePolygonRgn function and PtInRegion function are used to obtain the internal boundary array \( D_1 [D_{1x}, D_{1y}, D_{1rgb}] \).

In actual engineering, the generation of a keyway is formed on the shaft by a milling cutter, which is the same in the program. The generation of keyway data needs to be modified in the three-dimensional point cloud data \( [T_x, T_y, T_{rgb}] \). Coordinate transformation of point cloud data \( D_1 [D_{1x}, D_{1y}, D_{1rgb}] \) is required in the calculation process, and the specific algorithm is as Equation (19):

\[
\begin{align*}
T_{xi} &= \delta_1 \\
T_{yi} &= D_{1yi} \\
T_{zi} &= Z_{01} + D_{1zi}
\end{align*}
\]  

where \( \delta_1 \) ranges from 0 to \( h_z \), with an increment of 0.1. The algorithm converts the keyway cross section drawn on the xy plane to the yz plane through coordinates, and then assigns values to the negative direction of the x shaft continuously. RGB is defined as the same value as the background RGB (255, 255, 255), thus completing the “milling” operation. The 3D point cloud data with keyway shaft are obtained as \( [T_x, T_y, T_{rgb}] \).


A three-dimensional point cloud model of pin holes can be created according to the pin hole geometric factors \( [Z_{02}, r_z, h_z, nz] \). \( Z_{02} \) is the constraint control factor of the position of the pin holes in the axial direction, \( r_z \) is the radius of the hole, \( h_z \) is the hole depth, and \( nz \) can control the number of evenly divided pin holes in the circumferential direction of the shaft.

The two-dimensional boundary algorithm of the pin hole is as Equation (20):

\[
\begin{align*}
B_{2xi} &= r_z \cdot \cos c \\
B_{2yi} &= r_z \cdot \sin c
\end{align*}
\]  

where \( c \) is the angle variable in 0 to 360 degree increments of 0.1.

The filling of the point cloud data inside the pin hole is like the filling of the shaft inside. The CreatePolygonRgn function and PtInRegion function are used to obtain the inner boundary data \( D_2 [D_{2x}, D_{2y}, D_{2rgb}] \). For a shaft that requires \( nz \) pin holes, the pin hole data need to be copied, and the algorithm is as Equation (21):

\[
\begin{align*}
D_{2x} &= D_{2x} \cdot \cos \left( 2 \cdot \pi \cdot \frac{1}{nz} \right) - D_{2yi} \cdot \sin \left( 2 \cdot \pi \cdot \frac{1}{nz} \right) \\
D_{2y} &= D_{2x} \cdot \sin \left( 2 \cdot \pi \cdot \frac{1}{nz} \right) + D_{2yi} \cdot \cos \left( 2 \cdot \pi \cdot \frac{1}{nz} \right)
\end{align*}
\]  

where \( D_2' [D_{2x}, D_{2y}, D_{2rgb}] \) are the replicated pin hole point cloud data, \( j \) is 1, 2, \ldots, \( nz \).

In actual engineering practice, the generation of pin holes is formed by drilling holes on the shaft, which is also the case in the program. The generation of pin hole data needs to be modified in the three-dimensional point cloud data \( [T_x, T_y, T_{rgb}] \) of the shaft. In the calculation process, coordinate transformation of point cloud data \( D_2 [D_{2x}, D_{2y}, D_{2rgb}] \) is required, and the specific algorithm is as Equation (22):

\[
\begin{align*}
T_{xi} &= \delta_1 \\
T_{yi} &= D_{2yi} \\
T_{zi} &= Z_{02} + D_{2zi}
\end{align*}
\]  

In the formula, \( \delta_1 \) ranges from 0 to \( h_z \), with an increment of 0.1. The algorithm converts the cross section of the pin hole drawn on the xy plane to the yz plane through coordinates, and then assigns a value to the negative direction of the x shaft continuously. RGB is
defined as the same value as the background RGB (255, 255, 255), and then completes the operation of “drilling”. The 3D point cloud data with pin hole shaft are \( \mathbf{T} [\mathbf{Tx}, \mathbf{Ty}, \mathbf{Trgb}] \).

In summary, the shaft point cloud data generation diagram with keyway and pin hole is shown in Figure 5.

![Shaft with connected attribute sets: (a) shows the shaft with keyway; (b) shows the shaft with pin hole.](image)

**Figure 5.** Shaft with connected attribute sets: (a) shows the shaft with keyway; (b) shows the shaft with pin hole.

3.1.3. The Material Attribute Set of the Shaft

According to the People’s Republic of China aviation industry standard HB20082-2012 “Aviation high-temperature alloy shaft forging specification” the selected material geometry are alloy steel GH2901, GH4169, and GH4500 [17]. The controlling factor of the material geometry is \( [E, \alpha, \rho] \). \( E \) is the elastic modulus; \( \alpha \) is the coefficient of linear expansion, and \( \rho \) is the density. The control factors for these three materials are shown in Table 1.

<table>
<thead>
<tr>
<th>Material</th>
<th>Elastic Modulus (MPa)</th>
<th>Coefficient of Linear Expansion (°C)</th>
<th>Density (kg/m³)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GH2901</td>
<td>198</td>
<td>15.5</td>
<td>8.21</td>
</tr>
<tr>
<td>GH4169</td>
<td>207</td>
<td>12.9</td>
<td>8.19</td>
</tr>
<tr>
<td>GH4500</td>
<td>217</td>
<td>12.9</td>
<td>8.05</td>
</tr>
</tbody>
</table>

3.2. Modeling Algorithm for Rotor Disk Unit Point Cloud Model

3.2.1. The Type Attribute Set of the Disk

Turbine disk can be based on fir-tree mortise. Its coding chain model is \( [\mathbf{ng}, \mathbf{Fa}, \mathbf{Fb}, \mathbf{Zd}, \beta_2, \alpha_3, \mathbf{r}_1, \mathbf{r}_2, \mathbf{Zd}, \mathbf{R, r, d, n, nz}, \alpha_1] \). Its modeling algorithm is as follows:

1. Boundary point cloud data \( \mathbf{B} [\mathbf{Bx}, \mathbf{By}, \mathbf{Brgb}] \) of turbine disk can be obtained according to the control factor \( [\mathbf{R}, \mathbf{r}] \), and the algorithm are as Equations (23) and (24).

\[
\begin{align*}
\mathbf{Bx}_i &= r \cdot \cos c \\
\mathbf{By}_i &= r \cdot \sin c \\
\mathbf{Bx}_j &= R \cdot \cos c \\
\mathbf{By}_j &= R \cdot \sin c
\end{align*}
\]  

where \( \mathbf{B} \) is the boundary data storage array of disk; \( i = 0, 1, 2, \ldots, \mathbf{n}; \mathbf{j} \) is \( \mathbf{n} + 1, \mathbf{n} + 2, \ldots, \mathbf{n} + \mathbf{m}; \mathbf{c} \) is an angular variable in increments of 0.1 from 0 to 360 degrees.

After obtaining the boundary data \( \mathbf{B} [\mathbf{Bx}, \mathbf{By}, \mathbf{Brgb}] \), further fill the interior to obtain two-dimensional point cloud data \( \mathbf{D} [\mathbf{Dx}, \mathbf{Dy}, \mathbf{Drgb}] \) of the disk. The point cloud data
of the Fir-tree mortise and tenon is created on the two-dimensional point cloud data \( D[D_x, D_y, D_{rgb}] \) of disk. According to the control factor \([ng, Fa, Fb, Z_d, \beta_2]\) of the Fir-tree mortise and tenon, some parameters of the mortise and tenon need to be calculated first, as shown in Figure 6.

![Design drawing of fir-tree groove.](image)

**Figure 6.** Design drawing of fir-tree groove.

The algorithm for the first fir tooth can be expressed by Equations (25)−(27):

\[
O_x = -(ng + 0.3) \cdot Fb \cdot \pi \tag{25}
\]

\[
O_y = Z_d \tag{26}
\]

\[
\begin{cases}
B_{1x1} = O_x + \delta \cdot \cos(\beta_2) - \left(Fa \cdot \sin \left(\frac{2\delta}{Fb}\right)\right) \cdot \sin(\beta_2) \\
B_{1y1} = Z_d - \delta \cdot \sin(\beta_2) - \left(Fa \cdot \sin \left(\frac{2\delta}{Fb}\right)\right) \cdot \cos(\beta_2) 
\end{cases} \tag{27}
\]

In the formula, \(O_x\) and \(O_y\) are the initial position of the fir-tree slot, which is determined by the position constraint factor. \(i = 0, 1, 2, \ldots, n; \delta\) is the length variable from 0 to \(-O_x\) in increments of \(-0.01\).

For the second straight line, the algorithm is as Equation (28).

\[
\begin{cases}
B_{1xj} = B_{1x(n)} \\
B_{1yj} = \delta
\end{cases} \tag{28}
\]

where \(j = n + 1, n + 2, \ldots, n + m\); \(\delta\) is a length variable with increments of 0.1 from \(B_{1y(n)}\) to \(-B_{1y(n)}\).

The algorithm for the third fir tooth is as Equation (29).

\[
\begin{cases}
B_{1xk} = B_{1x(n-i)} \\
B_{1yk} = -B_{1y(n-i)}
\end{cases} \tag{29}
\]

where \(k = n + m + 1, n + m + 2, \ldots, n + m + o; i\) take 1, 2, \ldots, \(n\).

For the fourth straight line, the algorithm is as Equation (30).

\[
\begin{cases}
B_{1xl} = B_{1x(n+m+o)} \\
B_{1yl} = \delta
\end{cases} \tag{30}
\]

where \(l = n + m + o + 1, n + m + o + 2, \ldots, n + m + o + p\); \(\delta\) is the length variable of \(B_{1y(n+m+o)}\) through \(-B_{1y(n+m+o)}\) increments of \(-0.1\).

At this point, the two-dimensional boundary data modeling of a single fir-tree slot is completed, and the internal points are saved to \(D1[D1x, D1y, D1_{rgb}]\) by using the internal filling algorithm. The two-dimensional point cloud data of a single fir-tree tenon is arranged
in \( n_z \) equal parts of the circumference by the rotation replication algorithm, and the algorithm is as Equation (31).

\[
\begin{align*}
D_{2x_i} &= D_{1x_i} \cdot \cos \left( 2 \cdot \pi \cdot \frac{1}{n_z} \right) - (R + D_{1y_i}) \cdot \sin \left( 2 \cdot \pi \cdot \frac{1}{n_z} \right) \\
D_{2y_i} &= D_{1x_i} \cdot \sin \left( 2 \cdot \pi \cdot \frac{1}{n_z} \right) + (R + D_{1y_i}) \cdot \cos \left( 2 \cdot \pi \cdot \frac{1}{n_z} \right)
\end{align*}
\]  

(31)

where \( D_2 \) is the two-dimensional point cloud data of \( n_z \) fir-tree slots; \( i = 0, 1, 2, \ldots, n_z \); \( j \) take 1, 2, \ldots, \( n_z \); \( R \) is the outer diameter of the disk.

Replace the internal data of \( D_2[D_{2x}, D_{2y}, D_{2rgb}] \) with, \( D[D_x, D_y, D_{rgb}] \), the point cloud data of \( n_z \) fir-tree tenon on two-dimensional turbine disk \( D'[D_{tx}, D_{ty}, D_{trgb}] \) can be obtained.

6. 3D point cloud modeling algorithm for turbine disk;

By setting the control factor \( [d, a_1] \), \( d \) can set the thickness of turbine disk; \( a_1 \) can set the inclination Angle of the tenon in the z direction. Three-dimensional point cloud data \( T[T_x, T_y, T_{rgb}] \) of turbine disk, and the modeling algorithm is as Equation (32).

\[
\begin{align*}
T_{xi} &= D_{tx_i} \cdot \cos \left( \frac{a_1 \cdot \delta}{d} \right) - D_{ty_i} \cdot \sin \left( \frac{a_1 \cdot \delta}{d} \right) \\
T_{yi} &= D_{tx_i} \cdot \sin \left( \frac{a_1 \cdot \delta}{d} \right) + D_{ty_i} \cdot \cos \left( \frac{a_1 \cdot \delta}{d} \right) \\
T_{zi} &= \delta
\end{align*}
\]  

(32)

where \( \delta \) is the length variable from 0 to \( d \) in increments of 0.1.

3D point cloud data \( T[T_x, T_y, T_{rgb}] \) of turbine disk, model is shown in Figure 7.

![Figure 7. 3D point cloud model of turbine disk.](image)

(a) \( d = 25, \ a_1 = 0^\circ \)  
(b) \( d = 35, \ a_1 = 10^\circ \)

3.2.2. The Connection Attribute Set of the Disk

The connection attribute set of the turbine disk includes cooling holes and installation edges. Control factors need to be added to model the cooling holes and installation edges. The control factor for the cooling hole is \( [Z_x, Z_y, r_z, n_z] \), and the control factor for the installation edge on the disc is \( [Z_{01}, R, r, n_1, n_y, n_z_1] \).

7. Cooling hole point cloud modeling algorithm

According to the control factor \( [Z_x, Z_y, r_z, n_z] \), after the creation of the 2D point cloud data \( D[D_x, D_y, D_{rgb}] \) of the rotor disk class is completed, further point cloud data creation of the cooling holes can be carried out. The algorithm is as Equation (33).

\[
\begin{align*}
B_{3x_i} &= Z_x + r_z \cdot \cos c \\
B_{3y_i} &= Z_y + r_z \cdot \sin c
\end{align*}
\]  

(33)
where $B3$ represents the storage array of boundary data for a single cooling hole; $i$ takes 0, 1, 2, ... $n$; $c$ is an angle variable with an increment of 0.1 from 0 to 360 degrees. Fill in and define the point cloud data within the boundary and save it to obtain the two-dimensional point cloud data $D1_x, D1_y, D1_{rgb}$ of the cooling holes for a single hole.

Arrange the cooling holes in a circle of $nz_r$ on the disk circumference, and the algorithm is as Equation (34).

$$
\begin{align*}
D2_{xi} &= D1_{xi} \cdot \cos \left( 2 \cdot \pi \cdot \frac{i}{nz_r} \right) - D1_{yi} \cdot \sin \left( 2 \cdot \pi \cdot \frac{i}{nz_r} \right) \\
D2_{yi} &= D1_{xi} \cdot \sin \left( 2 \cdot \pi \cdot \frac{i}{nz_r} \right) + D1_{yi} \cdot \cos \left( 2 \cdot \pi \cdot \frac{i}{nz_r} \right)
\end{align*}
$$

(34)

where $D2$ represents the 2D point cloud data of $nz_r$ cooling holes; $i$ take 0, 1, 2, ... $n$; $j$ take 1, 2, ..., $nz_r$.

According to the control factor $[d]$ of the rotor disc class, the three-dimensional point cloud data $T[T_x, T_y, T_{rgb}]$ of the cooling hole can be further obtained, and the algorithm is as Equation (35).

$$
\begin{align*}
T_{xi} &= D2_{xi} \\
T_{yi} &= D2_{yi} \\
T_{zi} &= \delta
\end{align*}
$$

(35)

where $i$ is taken as 0, 1, 2, ... $n$; $\delta$ is a length variable with increments of 0.1 from 0 to $d$.

8. Installation edge point cloud modeling algorithm for disks;

The edge point cloud modeling algorithm installed on the disk is like the shaft, and there are two types of algorithms: one is a hollow circular tube like the shaft, and the other is a through hole disk structure like the disk. By combining the two point cloud modeling algorithms, the edge point cloud model installed on the turbine disk can be modeled, which will not be repeated here.

3.2.3. The Material Attribute Set of the Disk

The selected material geometries include iron-based high-temperature alloy GH3030, nickel based deformed high-temperature alloy GH4169, and nickel-based cast high-temperature alloy IN718. The control factor for the material geometry is $[E, \alpha, \rho]$. $E$ is the elastic modulus; $\alpha$ is the coefficient of linear expansion, and $\rho$ is the density. The control factors for these three materials are shown in Table 2.

Table 2. Physical properties of rotor disc material.

<table>
<thead>
<tr>
<th>Material</th>
<th>Elastic Modulus (MPa)</th>
<th>Coefficient of Linear Expansion (°C)</th>
<th>Density (kg/m³)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GH3030</td>
<td>210</td>
<td>13.2</td>
<td>8.40</td>
</tr>
<tr>
<td>GH4169</td>
<td>207</td>
<td>12.9</td>
<td>8.19</td>
</tr>
<tr>
<td>IN718</td>
<td>435</td>
<td>11.8</td>
<td>8.24</td>
</tr>
</tbody>
</table>

3.3. Modeling Algorithm for Rotor Blade Unit Point Cloud Model

3.3.1. The Type Attribute Set of the Blade

The rotor blade is connected to the mortising groove on the disk, and its 3D point cloud data are complicated. When conducting 3D point cloud data modeling, rotor blades are divided into tenons and blade segments according to their structure, and the main difference between turbine blades is the twist angle and tenon of the blade body. Therefore, the coding chain model of turbine blades is as follows: $[Z_0, R, d, n_z, n_z, n_g, n_Fa', n_Fb', n_Zd', n_{\beta1}, n_{\beta2}, n_{S1}, n_{S2}, n_{d_0}, n_m, n_k, n_c, n_Fd, n_Hb, n_{\alpha4}, n_{\alpha3}]$.

9. Modeling algorithm for tenon point cloud model.

Because the tenon needs to fit with the compressor disk or turbine disk, the tenon groove on the point cloud model is similar to that on the disk. One just has to adjust the control factors $[n_g, n_Fa', n_Fb', n_Zd', n_{\beta2}, n_{S1}, n_{S2}, n_{d_0}]$ to complete the tenon point cloud model creation.
According to the control factor \([\text{ng, Fa, Fb, } Z_d, \beta_2, S_1, S_2, d_s]\), fir-tree tenon boundary point cloud data \([B_2, B_{x2}, B_{y2}, B_{2rgb}]\) can be created. The schematic diagram of the fir-tree tenon is shown in Figure 8.

![Figure 8. Design drawing of fir-tree tenon joint.](image)

Due to the similarity between the 2D point cloud model algorithm of the blade fir-tree tenon and the 2D point cloud model algorithm of the disk fir-tree tenon groove, it will not be elaborated further.

10. Modeling algorithm for blade body point cloud model.

The most important part of the blade body is the blade profile parameter. The modeling rule refers to the quadratic curve method. The idea is to first determine the curvature line \(y_c\) and the thickness line \(y_t\), and then combine the two to draw the blade profile. The bending line is divided into two sections, namely, the first half \(y_{c1}\) and the second half \(y_{c2}\). Two corners \(\theta_1\) and \(\theta_2\) are designed in the bending line. The specific implementation algorithm can be expressed as Equations (36)–(40):

\[
y_{c1}(x_0) = \frac{c \cdot k}{6} \cdot \left( x_0^3 - 3mx_0^2 + m^2x_0 \cdot (3 - m) \right)
\]

\[
y_{c2}(x_0) = \frac{c \cdot k \cdot m^3}{6} \cdot (1 - x_0)
\]

\[
y_1(x_0) = 5 \cdot c \cdot t \cdot \left( 0.2969 \sqrt{x_0} - 0.126x_0 - 0.3516x_0^2 + 0.2843x_0^3 - 0.1034x_0^4 \right)
\]

\[
\theta_1(x_0) = \arctan \left( \frac{dy_{c1}}{dx_0} \right) = \arctan \left[ \frac{c \cdot k}{6} \cdot \left( 3x_0^2 - 6m_0 + 3m_0^2 - m_0^3 \right) \right]
\]

\[
\theta_2(x_0) = \arctan \left( \frac{dy_{c2}}{dx_0} \right) = \arctan \left( \frac{c \cdot k \cdot m^3}{6} \right)
\]

where \(c\) is the string length; \(k\) is the maximum relative camber; \(m \cdot c\) is the position of maximum thickness. \(t\) is the maximum relative thickness; The value of \(x_0\) ranges from 0 to 1.

The blade profile function is obtained.

The algorithm for the first curve is as Equation (41):

\[
\begin{align*}
  u &= x_0 - y_1(x_0) \cdot \sin[\theta_1(x_0)] \\
  v &= y_{c1}(x_0) + y_1(x_0) \cdot \cos[\theta_1(x_0)]
\end{align*}
\]

where \((u, v)\) is the first blade curve coordinate, and the value of \(x_0\) is from 0 to \(m \cdot c\).
The algorithm for the second curve is as Equation (42):
\[
\begin{align*}
  \{ \ u &= x_0 - y_1(x_0) \cdot \sin[\theta_2(x_0)] \\
  \ v &= y_2(x_0) + y_1(x_0) \cdot \cos[\theta_2(x_0)] \}
\end{align*}
\] (42)

where \((u, v)\) is the curvilinear coordinate of the second blade profile, and the value of \(x_0\) is from \(m \cdot c\) to \(c\).

The algorithm for the third curve is as Equation (43):
\[
\begin{align*}
  \{ \ u &= x_0 + y_1(x_0) \cdot \sin[\theta_3] \\
  \ v &= y_2(x_0) - y_1(x_0) \cdot \cos[\theta_3] \}
\end{align*}
\] (43)

where \((u, v)\) is the curvilinear coordinate of the third segment of the blade profile, and the value of \(x_0\) is from \(m \cdot c\) to \(c\).

The algorithm for the fourth curve is as Equation (44):
\[
\begin{align*}
  \{ \ u &= x_0 + y_1(x_0) \cdot \sin(\theta_4) \\
  \ v &= y_2(x_0) - y_1(x_0) \cdot \cos(\theta_4) \}
\end{align*}
\] (44)

where \((u, v)\) is the curvilinear coordinate of the fourth segment of the blade profile, and the value of \(x_0\) is from 0 to \(m \cdot c\).

Therefore, the blade profile boundary can be modeled by adjusting the blade profile control factor \([m, k, t, c, F_d, H_b, a_4, a_5]\). According to the above algorithm, the blade profile boundary \(D[D_x, D_y, D_{rgb}]\) can be obtained. Since the blade body is not straight on the tenon, the rotation angle of the blade profile on the tenon is controlled by the control factor \(a_4\), the algorithm is as Equation (45):
\[
\begin{align*}
  B_{x_i} &= B_{x_i} \cdot \cos(a_4) - B_{y_i} \cdot \sin(a_4) \\
  B_{y_i} &= B_{x_i} \cdot \sin(a_4) + B_{y_i} \cdot \cos(a_4)
\end{align*}
\] (45)

After rotating the blade profile boundary \(B'[B_{x}, B_{y}, B_{rgb}]\), the internal point filling algorithm can be used to obtain the two-dimensional blade point cloud model data \(D[D_x, D_y, D_{rgb}]\). Finally, the control factors \(H_b\) and \(a_5\) are used to obtain the three-dimensional point cloud data \(T[T_x, T_y, T_{rgb}]\) model of the rotor blade. The algorithm is as Equation (46):
\[
\begin{align*}
  T_{x_i} &= D_{x_i} \cdot \sin\left(\frac{a_5 \cdot \delta}{H_b}\right) + D_{y_i} \cdot \cos\left(\frac{a_5 \cdot \delta}{H_b}\right) \\
  T_{y_i} &= D_{x_i} \cdot \cos\left(\frac{a_5 \cdot \delta}{H_b}\right) - D_{y_i} \cdot \sin\left(\frac{a_5 \cdot \delta}{H_b}\right)
\end{align*}
\] (46)

where \(\delta\) is the length variable from 0 to \(H_b\) in increments of 0.1.

The generated 3D point cloud model image is shown in Figure 9.

3.3.2. The Connection Attribute Set of the Blade

The connection attribute set of the turbine blades has a wide variety of types and the structure is too complex to be studied in this article.

3.3.3. The Material Attribute Set of the Blade

The rotor blades are mainly made of titanium alloy, titanium matrix composite material, and superalloy blades in the compressor section. In the turbine section, nickel-based and nickel–Al-based superalloy titanium–aluminum alloy blades and ceramic-based composite blades are mainly cast. The controlling factors of the material geometry are \([E, \rho, \alpha, \rho]\), \(E\) is the elastic modulus; \(\alpha\) is the coefficient of linear expansion, and \(\rho\) is the density. The control factors of these three materials are shown in Table 3.
Physical properties of rotor blade materials.

<table>
<thead>
<tr>
<th>Material</th>
<th>Elastic Modulus (MPa)</th>
<th>Coefficient of Linear Expansion (/°C)</th>
<th>Density (kg/m$^3$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>N07750</td>
<td>214</td>
<td>7.0</td>
<td>8.28</td>
</tr>
<tr>
<td>TC11</td>
<td>110</td>
<td>10.4</td>
<td>4.5</td>
</tr>
</tbody>
</table>

4. Results and Discussion

4.1. Display of 3D Point Cloud Model Data Results in MFC

The essence of MFC is a library containing many objects that have been defined by Microsoft. Through it, programmers can efficiently develop a variety of applications based on the Windows operating system.

The realization of the point cloud model on MFC only needs to have obtained all the three-dimensional coordinate points of the model, draw the customer area of the application window through OnDraw(CDC*pDC) function, and display the obtained entity three-dimensional coordinate points using SetPixel function. In the MFC window display rule of vs, only two-dimensional images x and y can be displayed. The control factor $\theta$ can be changed by three-dimensional coordinate transformation, and the three-dimensional display angle of geometry can be adjusted, as shown in Figure 10. The algorithm is as Equation (47):

$$\begin{align*}
T'_{xi} &= T_{xi} \cdot \cos(\theta) - T_{zi} \cdot \sin(\theta) \\
T'_{yi} &= T_{yi}
\end{align*}$$

(47)

where $T_{xi}$ and $T_{yi}$ are the position coordinates of the cell model displayed in the MFC window; $\delta$ take 0, 1, 2, ..., n; $\theta$ shows the angle for the cell body in the window.

Figure 9. 3D point cloud model diagram of turbine blades generated using a modified algorithm.

Figure 10. Turbine blades and discs at different rotation angles.
4.2. Units Coordination Display

A variety of geometry 3D point cloud models will produce display interference when displaying or when display is incomplete. The main reason is that due to the sequence of geometry display algorithms, after the completion of the 3D display of a geometry, if the next geometry needs to be displayed, the former geometry may be covered by the 3D display model of the latter geometry. To solve this problem, inspired by the bounding box algorithm for surface intersection, the 3D point cloud data of geometry are no longer saved according to the sequence of data. They are arranged according to the bounding box data sequence; a bounding box is placed to store the geometry that needs to be displayed, and then the bounding box is displayed, and the coordinate information of the data points and the display color information are saved under the numbering order. The algorithm is as Equation (48):

\[
\begin{align*}
\begin{cases}
  QT_{xi} + \frac{x}{2} + x \cdot \left( T_{yi} + \frac{y}{2} \right) + x \cdot y \cdot \left( T_{zi} + Z_{0} \right) = T_{x} \\
  QT_{yi} + \frac{y}{2} + y \cdot \left( T_{yi} + \frac{y}{2} \right) + x \cdot y \cdot \left( T_{zi} + Z_{0} \right) = T_{y} \\
  QT_{zi} + \frac{z}{2} + z \cdot \left( T_{yi} + \frac{y}{2} \right) + x \cdot y \cdot \left( T_{zi} + Z_{0} \right) = T_{z} + Z_{0} \\
  QT_{ri} + \frac{x}{2} + x \cdot \left( T_{yi} + \frac{y}{2} \right) + x \cdot y \cdot \left( T_{zi} + Z_{0} \right) = T_{r} \\
  QT_{gi} + \frac{x}{2} + x \cdot \left( T_{yi} + \frac{y}{2} \right) + x \cdot y \cdot \left( T_{zi} + Z_{0} \right) = T_{g} \\
  QT_{bi} + \frac{x}{2} + x \cdot \left( T_{yi} + \frac{y}{2} \right) + x \cdot y \cdot \left( T_{zi} + Z_{0} \right) = T_{b} \\
\end{cases}
\end{align*}
\]

where \([X, Y, Z]\) is the size of the bounding box, \([T_{x}, T_{y}, T_{rgb}]\) is the three-dimensional point cloud array of the cell body; \([QT_{x}, QT_{y}, QT_{z}, QT_{rgb}]\) is the three-dimensional point cloud array of the cell body displayed inside the bounding box. The 3D display improved by this algorithm is shown in Figure 11.

![Figure 11](image_url)

(a) 3D view  
(b) side view  
(c) section view

**Figure 11.** Structural diagram of aircraft engine turbine rotor generated by geometric model encoding.

5. Conclusions

This paper presents a geometric coding method for aeroengine turbine rotor structure and introduces a computer graphics point cloud model display algorithm. The proposed method is based on geometric coding technology, which mainly encodes turbine structures. These geometric codes are related to the design parameters of turbine structures. The codes are based on geometric structures and have certain universality and independence. In the model study of this paper, once the coding is determined and the parameters of each geometric control factor are known, the point cloud model of the turbine rotor structure can be generated, which is equivalent to the inverse transformation of the coding. The resulting point cloud model data body is prepared in advance for later visualization of the point cloud model. In general, the computational efficiency of the coding model algorithm...
and the reconstruction of the point cloud model are very good, although more points need to be calculated. In addition, the more complex the turbine rotor model is, the more the geometric control factor parameters will be increased, thus increasing the detail level of the turbine rotor model structure. On the other hand, when displaying the turbine rotor point cloud model, inspired by the bounding box algorithm, the interference display problem existing in the sequential display of turbine rotor units is solved, and multiple unit models can be assembled and displayed normally. Finally, the coding provided to the machine learning algorithm will save design time for the structure design of aeroengine turbines, and have a reference value for the structure design of aeroengine turbines.

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

Funding: This research was funded by the Natural Science Foundation of Jiangxi Province, grant number 20201BBE51002.

Data Availability Statement: Data are contained within the article.

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

References

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.