B-CODE GENERATION FOR A CNC DENTISTRY WIRE BENDING MECHANISM

Rahimah Abdul Hamid[1]*, Kenta Morisaki[2†]and Teruaki Ito[1]

Abstract. In the present study, the generation of CNC wire bending code (B-code) is explored and an automatic converter is developed in Excel through some mathematical formulae. The B-code comprises of three parameters which have been determined from the previous work, known as feeding length (L), plane rotation angle (R) and bending angle (A). These parameters control the movement of the designed wire bending machine in this way; (L) tells the feeding mechanism how long the straight wire to be fed, (R) constructs the turning mechanism to rotate in certain degrees before the bending operation starts and (A) determines the degrees of bend, performed by the bending mechanism. The input of the converter is the Cartesian XYZ coordinate of each bend point, obtained from the CAD file. This paper also describes the XYZ extraction process from an IGES file, in addition to the CNC code generation. The practicality of these procedures is demonstrated through an example, starting from the wire design, follows by the XYZ extraction and ends with the B-code generation. Moreover, the graphical simulation of the bending operation by the designed mechanism is also presented. However, the capability of these codes in controlling the wire bending machine in the real bending operation will not be discussed in the present work.

Keywords. CNC wire bender, dentistry wire bending, code generation, XYZ extraction, IGES

1. INTRODUCTION

The paper reports on the adopted methodology in creating the CNC bending code through a developed converter in Excel. In a computer numerical controlled (CNC) technology, a set of G-code is sent to the control unit to drive the mechanisms in accordance with the desired final target shape. The G-code is generated from the designed features in CAD through various feature extraction algorithms as reported in Babic et al. (2008), Sunil and Pande (2008), Gupta and Gurumoorthy (2012), Madurai et al. (2012) and Kumar et al. (2014). The tool path generation is also depending on the machining process. The CNC code which is required for a bending operation is different from the additive and subtractive technology, for instance. In the present work, the B-code consists of three parameters, known as the length of a straight wire to be fed (L), the relative rotation angle between two adjacent bending planes (R) and the bend angle (A). These parameters have been determined from the previous work, as reported in (Rahimah and Teruaki, 2016a). LRA terminologies as referred to Liu et al. (2015) are adopted from the tube bending technology into this wire bending mechanism in order to establish the CAM data for the bending operation. The CAD file which contains the design of a target output somehow needs to be translated into machining language (CAM). These translated data helps to control the mechanism in performing the desired bending operation, in conjunction with the desired output. These parameters control the movement of the designed wire bending machine in this way; (L) tells the feeding mechanism how long the straight wire to be fed, (R) constructs the turning mechanism to rotate in certain degrees before the subsequent bending operation starts and (A) determines the degrees of bend, performed by the bending mechanism.

The converter is developed in Excel by using several mathematical formulae which would be explained in the next section. The input of this converter is the XYZ Cartesian coordinate of each bend point of the wire, obtained from the CAD wire design. In this study, the design of the target shape is constructed in CAD and even if the wire design is performed in different environment other than CAD, the generation of this B-code can also be executed, as long as the XYZ Cartesian coordinate for each bend point is given to the converter. In this paper, however, only the CAD design would be demonstrated and in addition to this, an XYZ extraction of the bend points from the IGES file would also be presented. This is necessary in consideration to various CAD users who might require to share their design. IGES file has been chosen as the CAD standard since the file is written in an ASCII format and could be easily understood in comparison to a binary file. Therefore, a brief explanation concerning to this IGES file format would be elaborated in the following section.

The final aim of this study is to develop a CNC dental wire bender. Till recently, the dental wire is bent manually by a dental technician due to customization. The automation of this operation only takes place in orthodontics application, where most of the trials are focusing on archwire bending (braces wire). However, many other wire bending in both prosthodontics and orthodontics applications are still done manually (Rahimah and Teruaki 2016b). Therefore, an attempt to design a new wire bending mechanism to automate this wire bending operation is made. In this paper, the generation of CAM data for the operational procedure of this mechanism would be demonstrated.

[1] Graduate School of Advanced Technology & Science, Tokushima University, 2-1 Minamijosanjima-cho, 770-8506, Japan

[2] Dept of Mechanical Engineering, Tokushima University

[1] Graduate School of Science and Technology, Tokushima University

[*] E-mail address: c501542005@tokushima-u.ac.jp

[†] E-mail address: knt.morisaki@gmail.com

[1] E-mail address: tito@tokushima-u.ac.jp

2. WIRE DESIGN

In this present study, CAD is used as the design platform to create the intended wire shape and any CAD software could be employed for this operation. Wire design in CAD is a series of vector lines, joined together to form a shape. Line geometry is used for the sketch where an estimation of the target shape is drawn on the scanned teeth. Line geometry is used instead of the spline as human dental technician could not draw a spline in reality, rather than estimating the wire bending process on the cast impression teeth through a sketch of multiple lines or arcs by a pencil. A theory of 3D line segmentation as introduced by Zhang (2013) is adopted to this recent work. In this theory, the final target shape is constructed from a series of ultra-line segments. This theory has been elaborated in Rahimah and Teruaki (2016c). A line segment is defined by Spiegel et al. (2013) as a part of a line that contains every point on the line between its endpoint. For example, if the target shape is a curve, therefore the curve is made of multiple ultra-line segments. The reason behind this approach is that XYZ coordinate for each point in these line segments would be manipulated to generate the B-code. If the dental technician makes a rough sketch on the physical dental cast impression, for example, a 3D digitizing of the sketch could also be obtained by using a magnetic sensor. The data has to be filtered and only the relevant points would finally be considered for the conversion process. In this paper, the use of the first method (CAD) would be presented and elaborated instead. Figure 1 exhibits the general procedure for this CNC code generation process, starting with the wire design, XYZ extraction and finally the B-code generation. Details of each process would be elaborated subsequently.

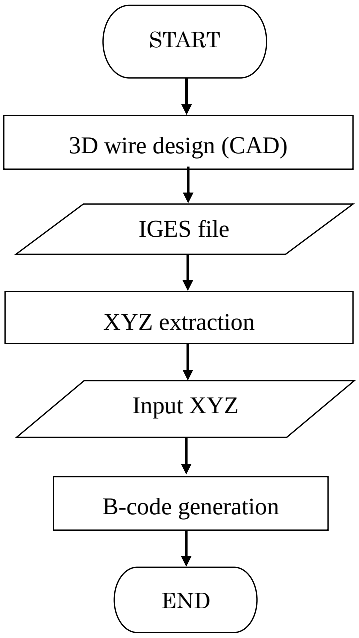


Fig. 1 Flowchart of the B-code generation process

2.1. XYZ Coordinate Extraction

The second procedure after the wire design activity is the XYZ coordinate extraction. The XYZ coordinate for each point along the 3D line is important for the B-code generation. Therefore, a set of point coordinate must be extracted first from the CAD file before the conversion process starts. There are a few ways to extract the XYZ coordinate data from the CAD file. One way is to directly extract the information in CAD environment, either manually or automatically. A manual extraction might be a cumbersome process if it involves many points. Alternatively, the automated extraction from CAD file needs a VBA macro program which extracts all the coordinates and exports them to an Excel file. This is fast and efficient, but an additional consideration to extract the coordinate directly from any CAD standard format is somehow explored. In this case, the CAD file is saved in an IGES (Initial Graphics Exchange Specification) file format. IGES file format is used as a standard format for a data transfer in CAD/CAM system. This IGES file gives a text file (ASCII format) where the coordinate of all points could be read according with the file formatting style. Data extraction from various CAD standard format such as IGES, STEP, and DXF for various machining technology have been explored by many researchers, such as Pratt et al. (2005), Bhandarkar et al. (2000) and Kim et al. (2008). A few researchers have chosen IGES as a standard format, as reported in Liu et al. (1996) and Nasr and Kamrani (2006a). In order to extract the desired information, the IGES file structure has to be defined first. Thus, Section 2.2 explains about this file structure, in brief.

2.2. IGES file structure

Similar to most CAD systems, IGES is based on the concept of entities. Entities in IGES are divided in three categories which are geometric entities, annotation entities and structure entities. Out of these three entities, geometric entities contain information that define the object, such as points, lines and arcs. All of these geometries have their own number, for example a line is numbered as 110. IGES file consists of 5 sections according to Nasr and Kamrani (2006b) which must appear in the following order (Table 1).

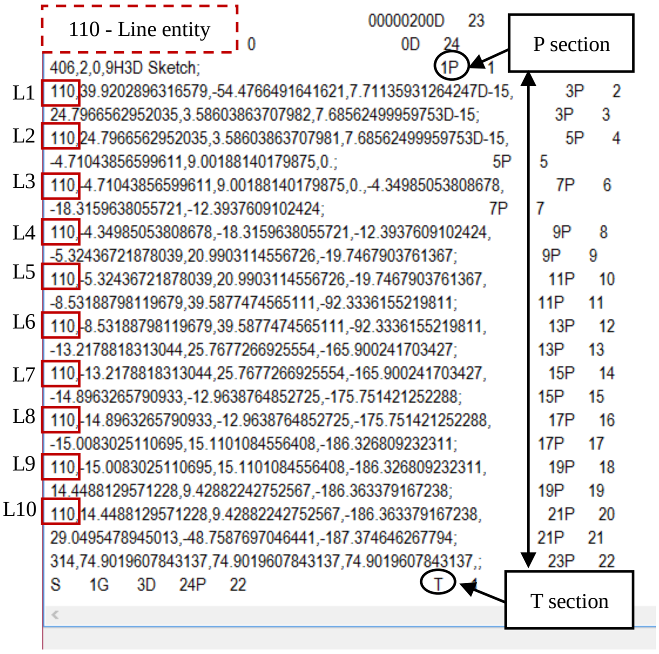
Table 1 IGES file structure

Start section	S	
Global section	G	
Direct entry section	D	All geometric entities are given here.
Parameter data section	P	All geometric entities are given here.
Terminate section	T	

Based on this IGES file structure, all geometric entities are given in the direct entry section (D) and the parameter data section (P). Direct entry section (D) is a list of all entities defined in the IGES file together with certain attributes associated with them, while parameter data section (P) section contains the actual data defining each entity listed in the D section. Hence, the XYZ coordinates for each point could be obtained from this parameter data (P) section.

2.3 Coordinate Extraction from an IGES file

Figure 2 shows a part of the IGES file that contains the parameter data section (P), listing all the start point and the end point coordinate for each line segment. As mentioned earlier, the line geometry is numbered as 110 in this text file format. Therefore, based on the IGES file example, the final target (as in Fig. 3) is constructed from a multiple line, referred as 110. Each 110 entity refers to 1 single line segment. In this IGES file format, a line segment is described by two points; the start point and the end point. The coordinate for these points is listed after entity index 110, where it starts with the start point coordinate and follows by the end point coordinate. Therefore, based on this IGES file, both coordinate for each line segment is extracted manually from row to row along the parameter data section (P section), as depicted in Fig. 2. Table 2 summarizes this information.



```
                                        00000200D    23
                                 0               0D    24
406,2,0,9H3D Sketch;                                  1P     1
110,39.9202896316579,-54.4766491641621,7.71135931264247D-15,      3P     2
24.7966562952035,3.58603863707982,7.68562499959753D-15;         3P     3
110,24.7966562952035,3.58603863707981,7.68562499959753D-15,      5P     4
-4.71043856599611,9.00188140179875,0.;                     5P     5
110,-4.71043856599611,9.00188140179875,0.,-4.34985053808678,      7P     6
-18.3159638055721,-12.3937609102424;                       7P     7
110,-4.34985053808678,-18.3159638055721,-12.3937609102424,      9P     8
-5.32436721878039,20.9903114556726,-19.7467903761367;        9P     9
110,-5.32436721878039,20.9903114556726,-19.7467903761367,      11P    10
-8.53188798119679,39.5877474565111,-92.3336155219811;        11P    11
110,-8.53188798119679,39.5877474565111,-92.3336155219811,      13P    12
-13.2178818313044,25.7677266925554,-165.900241703427;        13P    13
110,-13.2178818313044,25.7677266925554,-165.900241703427,      15P    14
-14.8963265790933,-12.9638764852725,-175.751421252288;       15P    15
110,-14.8963265790933,-12.9638764852725,-175.751421252288,     17P    16
-15.0083025110695,15.1101084556408,-186.326809232311;        17P    17
110,-15.0083025110695,15.1101084556408,-186.326809232311,      19P    18
14.4488129571228,9.42882242752567,-186.363379167238;         19P    19
110,14.4488129571228,9.42882242752567,-186.363379167238,       21P    20
29.0495478945013,-48.7587697046441,-187.374646267794;        21P    21
314,74.9019607843137,74.9019607843137,74.9019607843137,;       23P    22
S    1G    3D    24P    22                              T
```

Fig. 2 IGES file example which shows parametric section (P) with all XYZ coordinate for each LINE geometry in the content, numbered as 110 per column.

Table 2 lists the XYZ coordinate for each line segment, starting with line segment 1 (L1) until line segment 10 (L10). The endpoint of line segment 1 (L1) is the start point of a line segment 2 (L2) and so on. This is as a result of line continuity, which indicates that all of these 10 line segments are constructed continuously, starting from line segment 1 until line segment 10. Therefore, only the coordinate which is included in the red box as in Table 2 would be used for the B-code conversion. These coordinates would be applied to some mathematical formulae which would be described in the following section, in order to generate the desired LRA values. In the future work, an automatic IGES reader that is capable to extract these coordinates out of the original IGES file would be developed. Moreover, the program has to be able to filter the coordinate data at once and prepare the required point coordinate data only, as highlighted in the red box below. Then, the CNC code generation program can directly use the necessary coordinate data for the B-code conversion. This eliminates a few unnecessary steps, such as a manual XYZ extraction and also the appropriate XYZ data preparation which can simultaneously lessen the processing time and reduce the human error.

Table 2 Extracted XYZ coordinate for each line segment

	Start point			End point		
	x	y	z	x	y	z
L1	39.92	-54.48	7.71	24.80	3.586	7.69
L2	24.80	3.586	7.69	-4.71	9.002	0.00
L3	-4.71	9.002	0.00	-4.35	-18.32	-12.39
L4	-4.35	-18.32	-12.39	-5.32	20.99	-19.75
L5	-5.32	20.99	-19.75	-8.53	39.59	-92.33
L6	-8.53	39.59	-92.33	-13.22	25.77	-165.9
L7	-13.22	25.77	-165.9	-14.90	-12.96	-175.75
L8	-14.90	-12.96	-175.75	-15.01	15.11	-186.33
L9	-15.01	15.11	-186.33	14.45	9.43	-186.36
L 10	14.50	9.43	-186.36	29.05	-48.76	-187.38

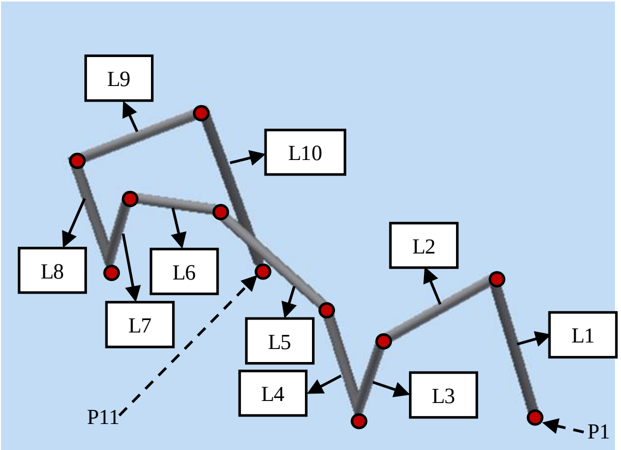


Fig. 3 Wire design of the example shape, consists of 10 line segments (L1-L10) which the coordinate of each point (P1-P11) is listed in Table 2.

3. BENDING CODE (B-CODE) GENERATION

This section explains about the general framework in generating the B-code data, starting from the definition of each parameter (L,R,A) and after that introducing each mathematical formula employed for the conversion process. In this wire bending mechanism, three bending parameters have been recognized to control the machine, known as length of bend (L), rotation of angle (R) and bend angle (A). An exemplary demonstration on the conversion process in the developed converter would be presented in Section 3.3 for a better clarification, by adopting the desired target shape as previously depicted in Fig. 3.

3.1. Definition of parameters

The following Fig. 4 illustrates the graphical representation of LRA parameters. L refers to the bending length, where in this example L1 is the length between point 1 (P1) and point 2 (P2), indicated by the arrow, while L2 is the length between point 2 (P2) and point 3 (P3). In this illustration, it clearly illustrates that the XYZ coordinate of point 2 (P2) is used twice in the calculation of L1 and L2, as P2 is the end point of L1 and also the start point of L2. This L parameter tells the feeding mechanism on how length a straight wire to be fed into the bending operation. The second parameter, which is the plane rotation angle (R) somehow refers to the change of plane between two adjacent bending. For instance, if L1 lies on plane A and L2 lies on plane B, therefore, the turning mechanism which is attached to the bending mechanism must at first make a rotation in accordance with the R value before the next bending operation starts. In this illustration, R is represented by a twist angle (β), instead. On the other hand, the bend angle (A) is indicated by θ, where the value of θ is obtained by determining the included angle (α), first. Details about this mathematical formulae would be discussed in the following section.

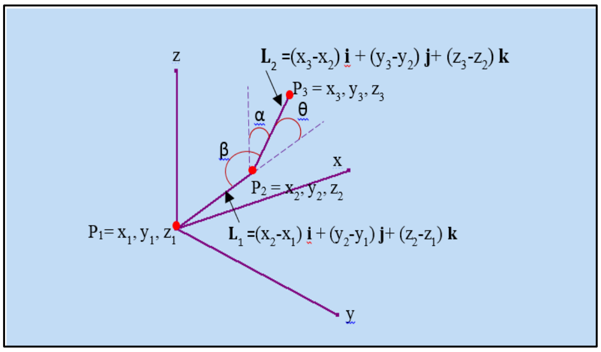

z
$L_2 = (x_3 - x_2)\,\mathbf{i} + (y_3 - y_2)\,\mathbf{j} + (z_3 - z_2)\,\mathbf{k}$
$P_3 = x_3, y_3, z_3$
α
θ
β
x
$P_2 = x_2, y_2, z_2$
$P_1 = x_1, y_1, z_1$
$L_1 = (x_2 - x_1)\,\mathbf{i} + (y_2 - y_1)\,\mathbf{j} + (z_2 - z_1)\,\mathbf{k}$
y

Fig. 4 Definition of bending parameters in between points (P1, P2, P3), with L = length (L), θ = bend angle (A), α = included angle and β = the plane rotation angle (R).

3.2. Mathematical formulae

This section discusses how some mathematical equations are employed in the conversion process. As briefly stated earlier, the B-code consists of three parameters, which are the bending length (L), the plane rotation angle (R) and the bend angle (A). Definition of each parameter has been described in the previous section. Firstly, the following Eq. 1 is used to get the L data. In this formula, x, y and z refer to the coordinate of a start point and the end point for each line segment, where the root sum square is adopted in calculating the length, or the magnitude of the position vector.

$$L_i = \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2 + (z_{i+1} - z_i)^2} \quad (1)$$

In the meantime, Eq. 2 and Eq. 3 are utilized in generating the plane rotation angle (R), where Eq. 3 is an elaborated version of Eq. 2, with a detail explanation for each variable is also enclosed. In this equation, x, y and z are all referring to the point coordinate of the line segment, either indicating to the point before or after the present point. This equation produces an answer in degrees. The formula calculates the length of vectors perpendicular to the end plane and the plane before the bend is executed. In addition, the direction of rotation is not considered in this Excel converter, where a counterclockwise direction should be assumed as negative, while a clockwise direction holds a positive value. The consideration of this direction sign would be considered in the future study, in developing a better converter by using Matlab. Matlab has been used by Masood et al. (2015) for a tool path generation in complex machining and an attempt to use Matlab for a wire bending would be soon explored. Nevertheless, the value of (R) is still the same, regardless of the sign. In addition, the first rotation angle (R1) is always zero, since the bending and turning mechanisms are still in the normal position during the first bending operation.

$$\beta = \left(\frac{\mathbf{p_1} \cdot \mathbf{q_2}}{\|\mathbf{p_1}\|\|\mathbf{q_2}\|}\right) \qquad (2)$$

$$\beta = \text{arcos}\left[\frac{A_1A_2 + B_1B_2 + C_1C_2}{\sqrt{{A_1}^2 + {B_1}^2 + {C_1}^2} \cdot \sqrt{{A_2}^2 + {B_2}^2 + {C_2}^2}}\right] \qquad (3)$$

where;

$$\begin{aligned}
A_1 &= (y_i - y_{i-2})(z_{i-1} - z_{i-2}) - (y_{i-1} - y_{i-2})(z_i - z_{i-2}) \\
A_2 &= (y_{i+1} - y_{i-1})(z_i - z_{i-1}) - (y_i - y_{i-1})(z_{i+1} - z_{i-1}) \\
B_1 &= (x_{i-1} - x_{i-2})(z_i - z_{i-2}) - (x_i - x_{i-2})(z_{i-1} - z_{i-2}) \\
B_2 &= (x_i - x_{i-1})(z_{i+1} - z_{i-1}) - (x_{i+1} - x_{i-1})(z_i - z_{i-1}) \\
C_1 &= (x_i - x_{i-2})(y_{i-1} - y_{i-2}) - (x_{i-1} - x_{i-2})(y_i - y_{i-2}) \\
C_2 &= (x_{i+1} - x_{i-1})(y_i - y_{i-1}) - (x_i - x_{i-1})(y_{i+1} - y_{i-1})
\end{aligned}$$

Lastly, Eq. 4, Eq. 5 and Eq. 6 describe some mathematical formulae in producing the bend angle (A). Equation 4 and Eq. 5 are the same in terms of definition, where Eq. 5 is a detailed version of Eq. 4. In this calculation, firstly, the length of a vector in two subsequent line segments are calculated and the included angle (α) between these two vectors are determined through a dot product of these vectors, as shown in Eq. 4. Inverse cosine of Eq. 4 produces an included angle, α which needs to be subtracted from 180° in order to obtain the bend angle value (θ), as in Eq. 6. In this paper, the value of (R) and (A) are in degrees, not radian.

$$\alpha = \left(\frac{\mathbf{v_1} \cdot \mathbf{v_2}}{\|\mathbf{v_1}\|\|\mathbf{v_2}\|}\right) \qquad (4)$$

$$\alpha = \text{arcos}\left[\frac{D_0D_1+E_0E_1+F_0F_1}{\sqrt{D_0{}^2+E_0{}^2+F_0{}^2}\cdot\sqrt{D_1{}^2+E_1{}^2+F_1{}^2}}\right] \tag{5}$$

where;

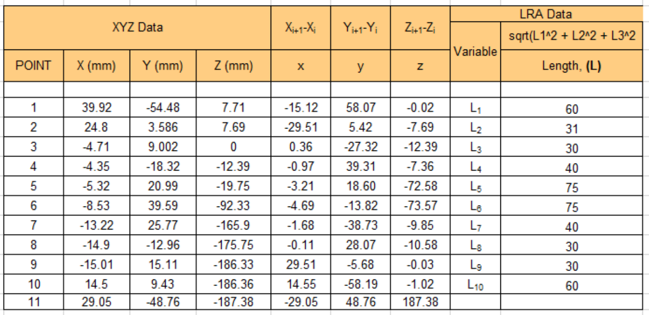
$$D_0 = (x_0 - x_1);\ E_0 = (y_0 - y_1);\ F_0 = (z_0 - z_1)$$
$$D_1 = (x_2 - x_1);\ E_1 = (y_2 - y_1);\ F_1 = (z_2 - z_1)$$

Finally,

$$\theta = 180° - \alpha \tag{6}$$

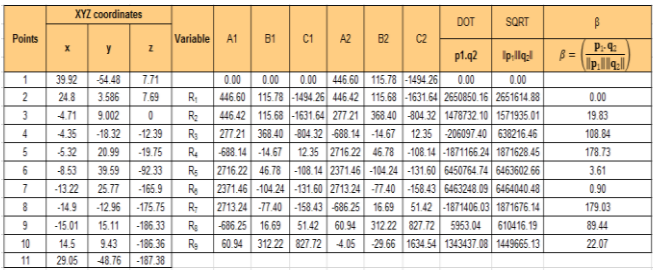
3.3. Exemplary calculation

In this section, an example of the calculation for each parameter would be demonstrated. For this demonstration, the data from a target shape as in Fig. 3 is adopted. Therefore, all 11 points for this calculation are referring to this figure. These coordinate of points have been extracted in the previous section (Section 2.3) and listed in Table 2. The coordinate for every point is inserted into each formula (Eq. 1 – Eq. 6), accordingly. The following Fig. 5a, Fig. 5b and Fig. 5c illustrate the screenshot of this converter in Excel for all conversion (L, β and θ). In the end, a compilation for all data is summarized in Table 3.



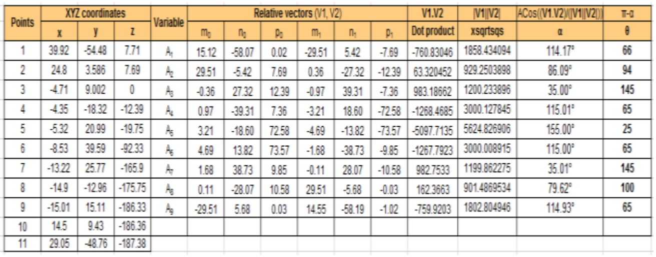
XYZ Data				X_{i+1}-X_i	Y_{i+1}-Y_i	Z_{i+1}-Z_i	LRA Data	
							Variable	sqrt(L1^2 + L2^2 + L3^2
POINT	X (mm)	Y (mm)	Z (mm)	x	y	z		Length, **(L)**
1	39.92	-54.48	7.71	-15.12	58.07	-0.02	L_1	60
2	24.8	3.586	7.69	-29.51	5.42	-7.69	L_2	31
3	-4.71	9.002	0	0.36	-27.32	-12.39	L_3	30
4	-4.35	-18.32	-12.39	-0.97	39.31	-7.36	L_4	40
5	-5.32	20.99	-19.75	-3.21	18.60	-72.58	L_5	75
6	-8.53	39.59	-92.33	-4.69	-13.82	-73.57	L_6	75
7	-13.22	25.77	-165.9	-1.68	-38.73	-9.85	L_7	40
8	-14.9	-12.96	-175.75	-0.11	28.07	-10.58	L_8	30
9	-15.01	15.11	-186.33	29.51	-5.68	-0.03	L_9	30
10	14.5	9.43	-186.36	14.55	-58.19	-1.02	L_{10}	60
11	29.05	-48.76	-187.38	-29.05	48.76	187.38		

Fig. 5a Screenshot of the converter which shows the feeding length (L) (in mm) generation process



Points	XYZ coordinates			Variable	A1	B1	C1	A2	B2	C2	DOT	SQRT	β
	x	y	z								p1.q2	$\lVert p_1\rVert\lVert q_2\rVert$	$\beta = \left(\frac{p_1 \cdot q_2}{\lVert p_1\rVert\lVert q_2\rVert}\right)$
1	39.92	-54.48	7.71		0.00	0.00	0.00	446.60	115.78	-1494.26	0.00	0.00	
2	24.8	3.586	7.69	R_1	446.60	115.78	-1494.26	446.42	115.68	-1631.64	2650850.16	2651614.88	0.00
3	-4.71	9.002	0	R_2	446.42	115.68	-1631.64	277.21	368.40	-804.32	1478732.10	1571935.01	19.83
4	-4.35	-18.32	-12.39	R_3	277.21	368.40	-804.32	-688.14	-14.67	12.35	-206097.40	638216.46	108.84
5	-5.32	20.99	-19.75	R_4	-688.14	-14.67	12.35	2716.22	46.78	-108.14	-1871166.24	1871628.45	178.73
6	-8.53	39.59	-92.33	R_5	2716.22	46.78	-108.14	2371.46	-104.24	-131.60	6450764.74	6463602.66	3.61
7	-13.22	25.77	-165.9	R_6	2371.46	-104.24	-131.60	2713.24	-77.40	-158.43	6463248.09	6464040.48	0.90
8	-14.9	-12.96	-175.75	R_7	2713.24	-77.40	-158.43	-686.25	16.69	51.42	-1871406.03	1871676.14	179.03
9	-15.01	15.11	-186.33	R_8	-686.25	16.69	51.42	60.94	312.22	827.72	5953.04	610416.19	89.44
10	14.5	9.43	-186.36	R_9	60.94	312.22	827.72	-4.05	-29.66	1634.54	1343437.08	1449665.13	22.07
11	29.05	-48.76	-187.38										

Fig. 5b Screenshot of the converter which shows the rotation angle (R) or β (in degrees) generation process



Points	XYZ coordinates			Variable	Relative vectors (V1, V2)						V1.V2	$\lvert V1\rvert\lvert V2\rvert$	ACos((V1.V2)/($\lvert V1\rvert\lvert V2\rvert$))	π-α
	x	y	z		m_0	n_0	p_0	m_1	n_1	p_1	Dot product	xsqrtsqs	α	θ
1	39.92	-54.48	7.71	A_1	15.12	-58.07	0.02	-29.51	5.42	-7.69	-760.83046	1858.434094	114.17°	66
2	24.8	3.586	7.69	A_2	29.51	-5.42	7.69	0.36	-27.32	-12.39	63.320452	929.2503898	86.09°	94
3	-4.71	9.002	0	A_3	-0.36	27.32	12.39	-0.97	39.31	-7.36	983.18662	1200.233896	35.00°	145
4	-4.35	-18.32	-12.39	A_4	0.97	-39.31	7.36	-3.21	18.60	-72.58	-1268.4685	3000.127845	115.01°	65
5	-5.32	20.99	-19.75	A_5	3.21	-18.60	72.58	-4.69	-13.82	-73.57	-5097.7135	5624.826906	155.00°	25
6	-8.53	39.59	-92.33	A_6	4.69	13.82	73.57	-1.68	-38.73	-9.85	-1267.7923	3000.008915	115.00°	65
7	-13.22	25.77	-165.9	A_7	1.68	38.73	9.85	-0.11	28.07	-10.58	982.7533	1199.862275	35.01°	145
8	-14.9	-12.96	-175.75	A_8	0.11	-28.07	10.58	29.51	-5.68	-0.03	162.3663	901.4869534	79.62°	100
9	-15.01	15.11	-186.33	A_9	-29.51	5.68	0.03	14.55	-58.19	-1.02	-759.9203	1802.804946	114.93°	65
10	14.5	9.43	-186.36											
11	29.05	-48.76	-187.38											

Fig. 5c Screenshot of the converter which shows the bend angle (A) or θ (in degrees) generation process

Table 3 Summary of the B-code conversion result

BEND	L (mm)	β (deg.)	θ (deg.)
1	60	0.00	66
2	31	19.83	94
3	30	108.84	145
4	40	178.73	65
5	75	3.61	25
6	75	0.90	65
7	40	179.03	145
8	30	89.44	100
9	30	22.07	65
10	60		

4. BENDING MECHANISM SIMULATION

In total, based on the desired target (as in Fig. 3), the total number of bends required for this bending operation is 10. This is confirmed by the conversion process, as shown in Table 3. In this section, a bending simulation of the designed bending mechanism would be presented, in accordance with the target shape. Figure 6 illustrates the bending sequence for bend 1 and bend 2 only, where step 1- step 4 illustrate the bend operation for the first bend, while step 5- step 8 exhibit the subsequent bend (bend 2).

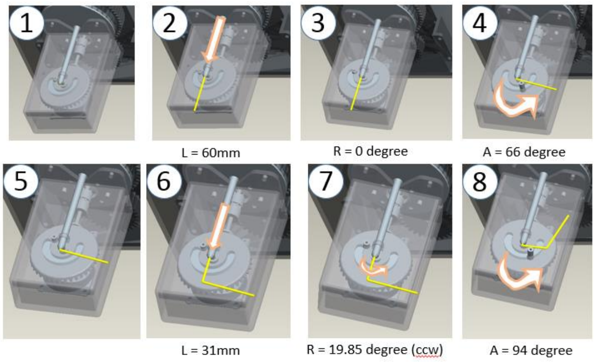


Fig. 6 Bending simulation for the specified target, illustrated by the mechanism for L1 bend (step 1- step 4) and L2 bend (step 5- step 8).

In step 1, the bending mechanism is in a normal position. Based on Table 3, the length (L) that needs to be fed by the sending mechanism is 60 mm. Therefore, in step 2, 60 mm of wire is fed by the wire sending mechanism towards the bending mechanism. Then, the turning mechanism rotates and make the first bend, with the bend angle (A) of 66 degree. The rotation angle (R) for the first bend is zero. The second bend operation follows the same procedure, where in step 6, the sending mechanism feeds 31 mm wire to the bending mechanism. Then, the turning mechanism rotates the wire into 19.83 degree in order to change the working plane from a previous bend. Finally, the bending mechanism makes the bend operation for bend 2, with a bend degree (A) of 94 degree. In this simulation, the rotation of plane (R) for the second bend is supposedly counterclockwise (CCW) and should be denoted by a negative sign. However the developed Excel converter could not show the sign difference. Therefore, a more robust converter would be developed in Matlab in the future study, as an alternative to improve the existing converter in Excel.

5. CONCLUSION

To conclude, the generation of B-code based on several mathematical equations for a specified target shape has been demonstrated, in conjunction with the simulation of some bending operation with the designed mechanism. Three parameters are considered significant to drive the machine, known as the straight wire feeding length (L), the rotation of plane angle in between bend (R) and the bending angle (A). The developed converter in Excel produces these parameters with a limitation in indicating the rotational degree (R) sign, either the turning mechanism shall make a counterclockwise (-) or clockwise (+) rotation. In addition, the XYZ extraction from the IGES file is manually executed. Therefore, an automatic XYZ extraction program from the IGES file and the LRA converter which is capable to indicate the rotational degree sign would be developed simultaneously in Matlab, as a future work of this study. Also, some more experiments to measure the robustness of the developed program would also be conducted to confirm its practicality for the real operation.

Acknowledgement

Authors would like to acknowledge Universiti Teknikal Malaysia Melaka (UTeM), Malaysia and Tokushima University, Japan for both financial support and provision of facility in conducting this study. The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper.

References

Babic, B., Nesic, N., and Miljkovic, Z. (2008). A review of automated feature recognition with rule-based pattern recognition. In *Computers in Industry*, **59**(4), 321–337.

Bhandarkar, M. P., Downie, B., Hardwick, M., and Nagi, R. (2000). Migrating from IGES to STEP: One to one translation of IGES drawing to STEP drafting data. In *Computers in Industry*, **41**(3), 261–277.

Gupta, R. K., and Gurumoorthy, B. (2012). Automatic extraction of free-form surface features (FFSFs). In *Journal of Computer Aided Design*, **44** (2), 99–112.

Kim, J., Pratt, M. J., Iyer, R. G., and Sriram, R. D. (2008). Standardized data exchange of CAD models with design intent. In *Journal of Computer Aided Design*, **40**(7), 760–777.

Kumar, S. P. L., Jerald, J., and Kumanan, S. (2014). Automatic Feature Extraction and CNC Code Generation in a CAPP System for Micromachining. In *Procedia Materials Science*, **5**(2014), 1986–1997.

Liu, S. C., Gonzalez, M., and Chen, J. G. (1996). Development of an automatic part feature extraction and classification system taking CAD data as input. In *Computers in Industry*, **29**(3), 137–150.

Liu, B.-Q., Yong-Chen, Xie, C., and Shi, L. (2015). Inventor-based simulation analysis for NC tube bending process. In *Proceedings of the 2015 International Conference (MME 2015)*, World Scientific, pp.376–388

Madurai, S. S., and Lin, L. (1992). Rule-based automatic part feature extraction and recognition from CAD data. In *Computers & Industrial Engineering*, **22**(1), 49-62.

Masood, A., Siddiqui, R., Pinto, M., Rehman, H., and Khan, M. A. (2015). Tool path generation, for complex surface machining, using point cloud data. In *Procedia CIRP*, **26**(2015), 397–402.

Nasr, E. A., and Kamrani, A. K. (2006a). A new methodology for extracting manufacturing features from CAD system. In *Journal of Computers and Industrial Engineering*, **51**(3), 389–415.

Nasr, E. A., & Kamrani, A. K. (2006b). IGES Standard Protocol for Feature Recognition CAD System. In *Rapid Prototyping: Theory & Practice*, Springer US, pp. 25–62.

Pratt, M. J., Anderson, B. D., and Ranger, T. (2005). Towards the standardized exchange of parameterized feature-based CAD models. In *Journal of Computer Aided Design*, **37**(12), 1251–1265.

Rahimah, A. H., and Teruaki, I. (2016a). Computer-aided wire design using sheet metal for dentistry application. In *International Design and Concurrent Engineering Conference 2016*, pp. 1–6.

Rahimah, A. H., and Teruaki, I. (2016b). Verification of sheet metal based wire bending procedures, In *JSME 26th Design & Systems Conference*, pp. 1–6.

Rahimah, A. H., and Teruaki, I. (2016c). 3D Prosthodontics Wire Bending Mechanism with a Linear Segmentation Algorithm. In *Journal of Advanced Manufacturing Technology*, **Special Issue (TMAC) Symposium 2016**, 33–46.

Rath, K. C., Kumar, A., & Tigga, A. M. (2012). Design of IGES reader software for downstream manufacturing system in support to CAX and Computer graphics. In *International Journal of Computer Application*, **3**(2), 399–409.

Spiegel, M. R., Lipschutz, S., and Liu, J. (2013). *Mathematical Handbook of Formulas and Tables. McGraw Hill* (Vol. 53).

Sunil, V. B., and Pande, S. S. (2008). Automatic recognition of features from freeform surface CAD models. In *Journal of Computer Aided Design*, **40**(4), 502–517.

Zhang. W. (2013). Research of 3D Virtual Design and Automated Bending of Oral Orthodontic Archwire. In *International Journal of Advancements in Computing Technology*, **5**(8), 741–749.