2D AUTOCAD GEOMETRIC DATA EXTRACTION AND POST-PROCESSING FOR NUMERICAL CONTROL

Tito E. Mwinuka* and Beda M. Mutagahywa **
Department of Mechanical and Industrial Engineering,
University of Dar es Salaam, P. O. Box 35131 Dar es Salaam, Tanzania
E-mail: * tmwinuka@udsm.ac.tz ** bmutag@udsm.ac.tz

ABSTRACT

In this paper, a method of extracting 2D geometric data from AutoCAD drawing and post-processing for Numerical Control (NC) is presented. Geometric data already available in the Drawing Interchange Files (DXF) can be utilized to accurately and economically prepare part programs for controlling NC/CNC machine tools. AutoCAD, being general purpose software, stores its geometric data in a form that cannot be directly applied for automatic generation of Numerical Control (NC) codes. The way around the discrepancy between the structure of a drawing database (DXF files) and the information requirement for part programming is addressed in this work. NC programming of a test component drawn in AutoCAD is also presented.

Keywords: CAD data extraction; NC programming; CAD/CAM integration

INTRODUCTION

In order to accomplish machining tasks, numerically controlled (NC) machine tools have positions and speeds of axes controlled by insertion of numerical data into the Machine Control Unit (MCU). Computer Numerically Controlled (CNC) machine tools are the modern generation of NC machine tools that incorporate a computer to assist preparation, storage and transfer of part programs and diagnosis of the machine tool. A set of instructions (numerical data) used to manufacture a component is called a part program. A process of preparing a part program is known as part programming. Several methods of preparing part programs exist. These are manual part programming, Computer-Aided Part Programming and the use of Computer-Aided Design and Computer-Aided Manufacturing (CAD/CAM) systems.

In manual part programming, the programmer interprets the drawing, performs necessary calculations regarding the locations of the tool and manually writes a part program using codes that can be understood by a particular controller. These codes are unfamiliar and needs a lot of experience in writing and understanding them. Further to that, manual part programming is tedious, prone to errors and limited to only simple components. Instead of writing part programs using unfamiliar codes, high level programming languages in computers can be used to define part geometry and direct the tool along the path to get the cutter location data (CLDATA) which can be post-processed to make a program that suits a particular machine tool/controller configuration. Several computer-aided part programming systems, such as, APT (Automatically Programmed Tools) exists. These systems make use of English-like statements; GOTO (Go to), GOLFT (Go left), COOLNT/ON (Coolant on), SPNDL CW (Spindle to rotate clockwise) etc which may be easier to use than unfamiliar machine codes like G01 (linear interpolation), M03 (spindle

clockwise) etc. The use of these APT-like programming systems involves mainly geometric statements which defines all geometric entities like points, lines, circles, planes associated with component geometry and motion statements that directs to tool movements based on the geometric entities already defined.

Nowadays most of drawing and modeling of components is done very efficiently and user friendly in CAD systems like AutoCAD, SolidWorks, Pro-Engineer etc. So it is more convenient to use geometric data of a component already present in the drawing database than using the high level language like APT. This has lead to the evolvement of CAD/CAM systems in which a component is modeled in CAD and part programs are automatically prepared by a CAM system that can access a CAD database. CAM systems can also import drawings from different CAD systems. It is very unlikely to find APT-like programming still in use today.

CAD based programming for NC machine tools require preparation of manufacturing process and modifying part geometry into contour representation (tool path) (Rembold and Dillman, 1986). The complexity of tool path depends on machining process like, wire EDM, milling, turning and drilling (Mwinuka, 1995) and the type and combination of tools used. Since CAD systems are general purpose, customization tools are needed for engineers to customize the software to their process and products (Machine Design, 1992). AutoCAD, for instance, has got common architecture which allows one to customize and extend many AutoCAD features to suit particular requirements (Autodesk, 1990). One can create script files to automate length command sequences, use DXF files to transmit drawing geometry to other programs for analysis, use AutoLISP or ADS (AutoCAD Development System) to perform calculations, automate repetitive tasks, create new commands or redefine existing commands.

Aslan *et al*. (1999) presented a method of extracting data from a CAD model for rotational parts to be machined by the turning centre. The method involved vertex extraction and feature extraction. However in this method, the entities in a DXF file have to be in sequenced form already, which is not always the result of drafting activity. Yildiz *et al*. (2006) presented a feature based CAM system for rotational parts that uses DXF files. Lack of automatic sequencing of entities is one of the weaknesses of this system. Full automation of CAM is currently hampered by the discrepancy between design entities e.g. lines, arcs, faces, volumes etc, which have no relevance to manufacturing and manufacturing features like pockets, slots, holes and profiles. This has been addressed by the use of Feature-Based CAD systems (Hanada and Hoshi, 1992) and Feature Recognition Systems where a program recognizes machining features from a model done in design features. These feature recognition systems are just evolving from recent researches and are yet to go commercial. Lack of feature recognition capabilities in most of the commercial CAD/CAM system is one of the drawbacks in automating process planning which is a link between CAD and CAM. In depth discussion on automatic feature recognition techniques are beyond the scope of this paper. Owodunni and Hinduja (2002) presented a substantive description and evaluation of existing feature recognition techniques.

STRUCTURE OF A DXF FILE

The size and content of the DXF file has been changing over the years from one

version to another. This necessitates having programs that will correctly extract the right data no matter what version of AutoCAD is used. The DXF file consists of six sections which are Header section, Class section, Tables section, Block section, Entities section and Objects section. All geometric data of the drawing is kept in Entities section as line segments, arc segments, elliptical segments, circles and ellipses. Data that is kept in the entity section is, for a line segment, x, y and z coordinates of a starting point and x, y and z coordinates of the end point. For an arc, a centre point, radius, start angle and end angle are kept. Also, the start angle and end point angle as referred to the arc centre. For a circle, a centre point and radius can be found in a DXF file.

The main problems associated with data arrangement in DXF file are: (i) Entities are not orientated, since two segments can meet head to head or tail to tail. This is one of the drawbacks for their direct application in NC. (ii) Entities not sequenced but are randomly arranged in the file in the order they were drawn. This is again a drawback in their use for NC applications especially in determining tool paths that require a sequenced list of entities. Though it is possible to have the entities sequenced in a DXF file by manually selecting them in a particular order, this method is time consuming and is only suitable for simple contours. Very short entities are always prone to be missed out during selection process.
If a starting point of a line segment is denoted by (x1,y1,z1) and endpoint by (x2, y2, z2) then, in the entities section, x1 is preceded by " 10", y1 by " 20", z1 by " 30", x2 by " 11", y2 by " 21" and z2 by " 31". If an arc segment, x, y and z coordinates of the centre point are preceded by " 10", " 20" and " 30" respectively, radius is preceded by " 40", starting angle by " 50" and ending angle by "51". Strings "ARC", "LINE", "CIRCLE" etc are used to indicate the type of entity and the string "ENTITIES" is used to identify the entities section. The section is terminated by the string "ENDSEC".

What can be computed from this information, relevant to NC part programming, is x increment and y increment between end points of line segment. Additionally for an arc segment, working angle can be computed, and interpolation type whether clockwise or anti-clockwise can be determined. The lengths of the segments can also be computed which can lead to computation of total length of the tool path, machining time and machining cost.

DATA EXTRACTION ALGORITHM

Important Bits of the Algorithm

In order to be able to extract geometric data of the drawing, a drawing must be saved as DXF file. If a tool path is not the same as part contour, a path has to be constructed using the part contour and then saved as DXF file.

In accessing the data, a program needs to skip all the items in the file before the entities section. A "while….do" loop is used. When in entities section, a program has to progressively visit all the entities until the end of entities section is encountered. Entities which are joined must have common endpoint. Sequencing, starts from the entering element. The program has to go through the file to identify a particular entity with one of its terminal points coinciding with the endpoint of the previous entity. If the contour is continuous, the program must pick at least one entity on its way down the file. The program first counts the number of

entities in the file then it is set to go through the file repeatedly several times equal to the number of entities. The cycles are however terminated when the starting point on the contour is reached. In actual fact the program ends after few cycles well below the number of entities since it can pick more than one entity on its way down the file.

If a contour has a break, then no entity will be intersecting with the endpoint at the break. Discontinuity will be detected if no intersecting entity is picked when the entity section is fully traversed once. Error message is printed whenever discontinuity is detected. When the intersection is found, the entity orientation has to be reversed (swapping the end points) if the entities appear to be joined head to head. In the case of arcs the modes of interpolation are also changed from counterclockwise to clockwise. A number of computations need to be performed, including, endpoints of arcs, start and working angles (needed for AGIE wire EDM machine). Expressions for calculating working angles have to be used depending on the mode of interpolation, determined as described in section 3 and the quadrants on which the start and end angle resides.

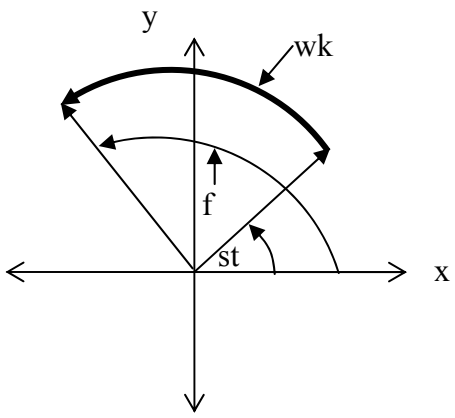
Start point, (x1, y1) and endpoint, (x2, y2) of an arc, with centre point (x, y) and radius r, as shown in Figure 1a are determined as follows;

$$x1 = x + r*\cos(\text{start angle}) \quad (1)$$

$$y1 = y + r*\sin(\text{start angle}) \quad (2)$$

$$x2 = x + r*\cos(\text{finish angle}) \quad (3)$$

$$y2 = y + r*\cos(\text{finish angle}) \quad (4)$$



Key:
st=starting angle
fn=finish angle
wk=working angle

Figure 1a: Angles defining an arc

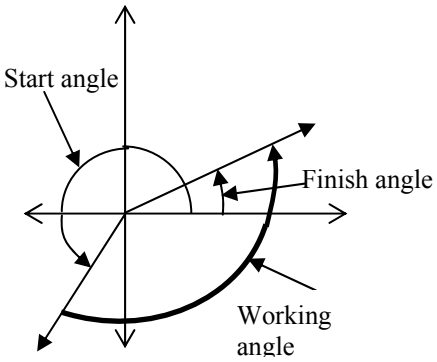


Figure 1b: Situation of finish angle greater than start angle in CCW

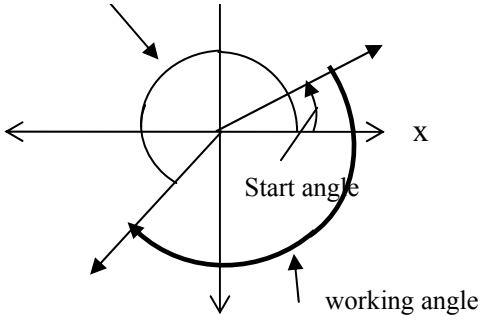


Figure 1c: Situation of finish angle less than start angle for CW

Working angle of an arc segment is computed as follows;

- For counterclockwise interpolation, if finish angle is greater than start angle;
- Working angle=finish angle-start angle.
- However if finish angle is less than start angle as shown in Figure 1b, then working angle=(360-start angle) + finish angle.

For clockwise interpolation, if finish angle is less than start angle, work angle=start angle-finish angle. However if finish angle is greater than start angle, as shown in Figure 1c, then work angle = 360-(finish angle –start angle).

Software Structure and Implementation

The software structure is divided into four modules namely ENTER, AUTOSEQUENCE, AGIEPOST and LINK. It is implemented in AutoLISP language in AutoCAD2002. However, the programs are independent of AutoCAD version and can be used with any other version of AutoCAD. These are defined as follows;

Enter: Used to select, using a mouse an entering (or starting) entity in a contour. This segment has to be added to the drawing, as it may not be part of the component 2D geometry. The module simply creates a DXF file consisting of that single entity.

Autosequence: Used to automatically sequence the entities into a tool path and therefore generating a cutter location data file that is used as an input to the post-processing module. In this work cutter location data refers to a DXF file containing a sequenced and orientated lines and arc segments. Any discontinuity in the contour is detected by this module and instructs the user to edit the drawing accordingly. The flow chart for the module autosequence is shown in Figure 2.

Agiepost: This is a post processing module for a target machine used in this work, that is, AGIECUT DEM15 wire Electro Discharge Machine. The flow chart for this module is shown in Fig. 3. However using cutter location data obtained using *autosequence* module, different pieces of software can be written to post process the data for other types of NC/CNC machine tools.

Link: This is used to link several contours into one part program. *Enter*, *Autosequence* and *Agiepost* have to be executed for each contour and several pieces of part program codes are saved in different files. These files are opened by the module "*link"* in order to create one part program for the entire component. The contours are linked blocks of linear interpolation in a positioning (non-cutting) mode so that machine axes can be moved from endpoint of a previous contour to starting point of next contour before starting another machining phase.

TESTING AND RESULTS

Several 2D contours have been tested. However a representative component shown in Figure 4 is used as an example. This component is chosen since it has enough complexity and contains both line and arc segments. The component is drawn in AutoCAD. ARRAY command is used to get circumferentially distributed repetitive features. And entering element (not part of the contour) is added. A drawing is saved in DXF format.

The programs *enter, autosequence, agiepost* and *link* are loaded during AutoCAD session. New commands "enter", "autosequence", "agie" and

"link" which are associated with previously loaded AutoLISP programs can be invoked by typing them on a command line. When sequencing progresses, the contour is drawn, in a different colour, on AutoCAD graphics area. This indicates the status of sequencing process and in case of discontinuous contour, a point where the contour breaks. A portion of part program, in tab sequential format for the target machine is shown in Table 1. A part program is punched into a tape using Facit 4070 Tape Punch connected to a computer via the RS 232 port. Component contour was machined correctly when the part program was run in the machine.

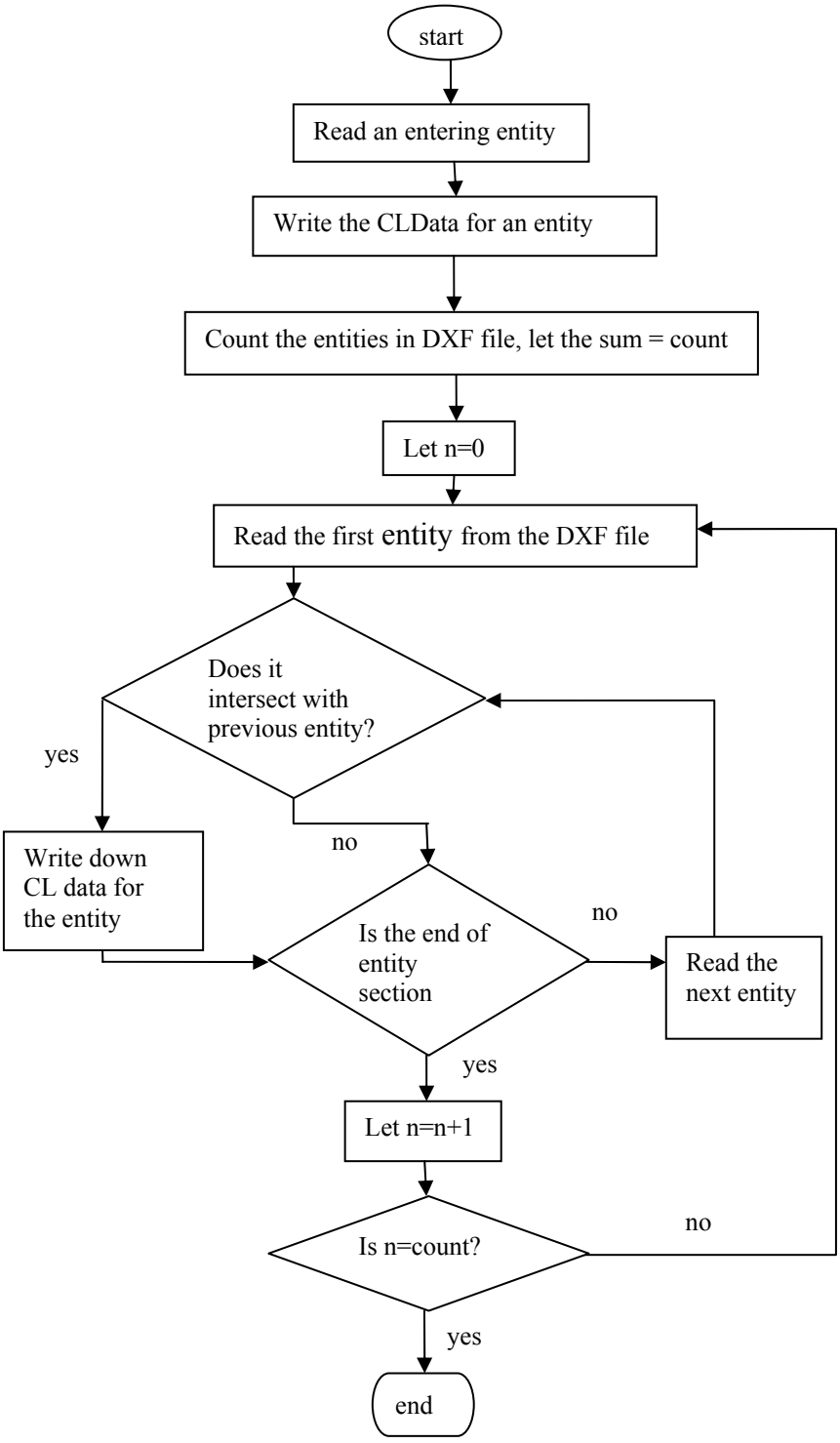


Figure 2: Flow chart for the module "autosequence"

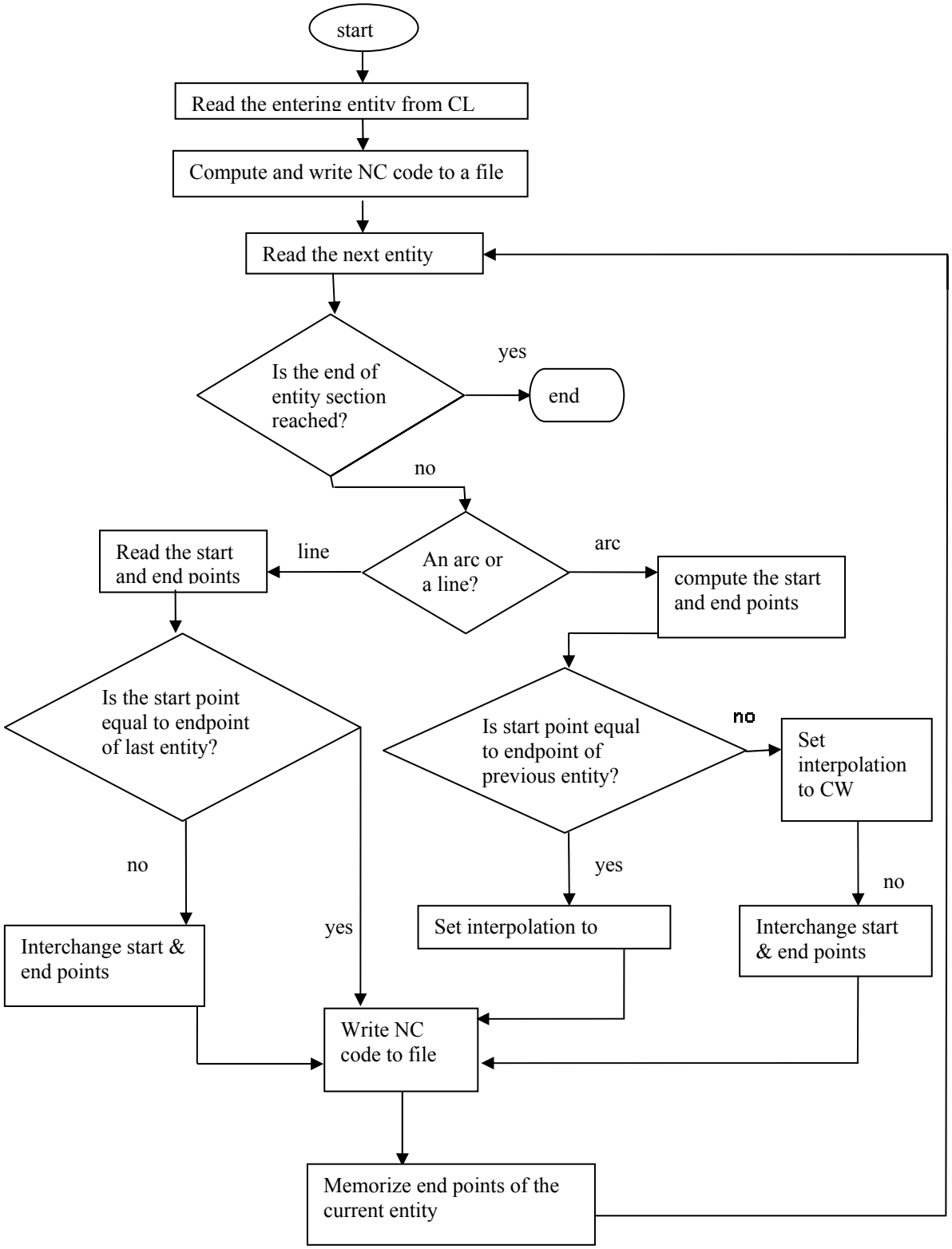


Figure 3: Flow chart for the module "agiepost"

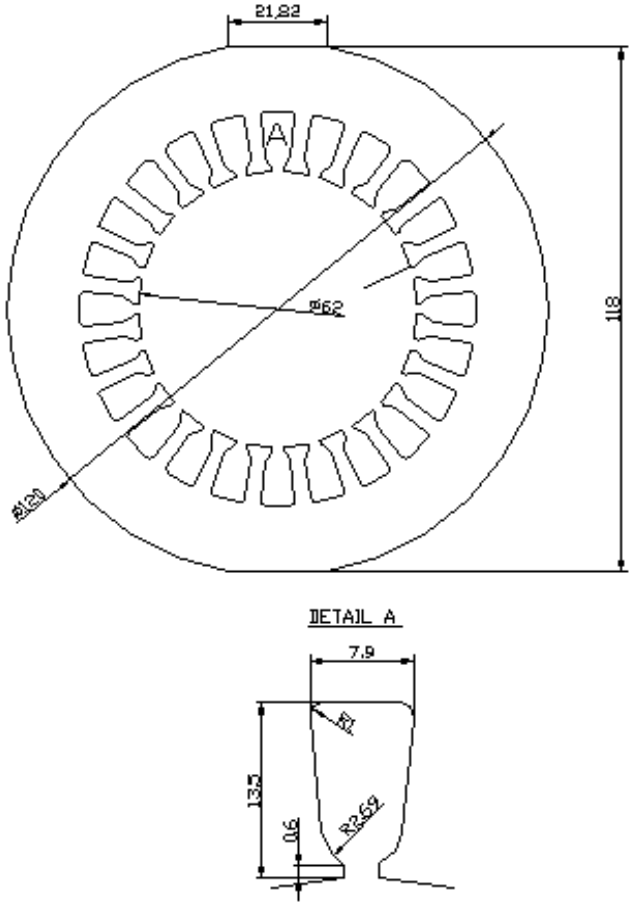


Figure 4: A test component

Table 1: A portion of part program for a test component

Program Start	Sequence number	Interpolation type	Start angle	Working angle	Arc radius/ x-increment	Arc radius/ y-increment	Operation mode	Program end
%	001	03	180	060	+000400	+000400	02	
	002	03	210	060	+000400	+000400	02	
	003	01	000	000	+020835	-005788	02	
	004	01	000	000	+000752	+000599	02	
	005	03	247	053	+002690	+002690	02	
	006	03	300	001	+002361	+002361	02	
	----	---	----	---	----------	----------	---	
	441	01	000	000	-000866	-000418	02	
	442	03	028	008	+031000	+031000	02	
	443	03	036	001	+005207	+005207	02	02

This NC code is in tab sequential format. Symbol % means program start. The first column represents block sequence number, second column represent type of interpolation, i.e. 03 for counterclockwise interpolation, 02 for clockwise

interpolation and 01 for linear interpolation. Third column represents starting angle and fourth column represents working angle (applicable for arc segments). Fifth column represents radius of an arc or x increment for line segment. Sixth column also represent radius of an arc or y increment for line segment. Seventh column represent the mode i.e. 02 for cutting (sparking) and 01 for non cutting (positioning). The last word 02 denotes program end.

DISCUSSION

AutoCAD and other CAD systems can be customized to suit particular requirements. Though this work has only dealt with Numerical Control programming from 2D AutoCAD drawing, many other customizations can be done depending on one's needs. This system is said to have "generative" numerical control programming where numerical control operation sequences are determined automatically by the system unlike the old APT programming where all tool positions and commands are normally generated by explicit user input. For wire erosion operations dealt with in this work, the tool path coincides with the feature geometries of the component. For other processes like turning and milling, however, the tool path has to be drawn fully before embarking into these programs. Tool path generation from 2D contours is beyond the scope of this work. OFFSET command is useful if one wants to manually create a tool path for operation like pocket milling.

However, this work is limited to contour made up of line segments, arc segments and circles. The programs need to be modified to be able to extract elliptical arcs and ellipses.

CONCLUSIONS

Customization of CAD systems, including AutoCAD, is achievable and can lead to optimum utilization of this expensive software. A method of extracting 2D geometry presented in this work gives useful results for downstream applications including Numerical Control. This has addressed the problem of discrepancy between drawing database structure and requirements of downstream applications that make use of the drawing geometry.

REFERENCES

AutoCAD Reference Manual Release 11, (August 1990). Autodesk Ltd.

Aslan, E., Seker, U. and Alpdemir, N., (1999). Data Extraction from CAD model for rotational parts to be machined at turning centres. *Turkish Journal of Engineering and Environmental Science,* Vol. 23, pp. 339-347.

Hanada, T. and Hoshi, T., (1992). Block like component CAD/CAM system for fully automated CAM processing. *Annals of the CIRP,* Vol. 41 No.1, pp. 551.

Machine Design Magazine October 22, 1992, 129p.

Mwinuka, T. E. (1995).Numerical Control Postprocessing from CAD Models. MSc Dissertation, University of Dar es Salaam, Tanzania.

Owodunni, O. O and Hinduja, S. (2002). Evaluation of Existing Feature recognition algorithms. Part 1 Theory and Implementation.*Proceedings of the Institution of Mechanical Engineers*, 216 Part B, pp. 839-851.

Rembold, U and Dillman, R. (1986). Computer-Aided Design and Manufacturing. Springer-Verlag Berlin Heidelberg New York Tokyo, 201p.

Yildiz, Y., Korkut, I. and Seker, U. (2006). Development of a Feature-Based CAM System for Rotational Parts. *Gazi University Journal of Science*, Vol. 19, No. 1, pp. 35 - 40.