

# Autocad Programming For Reinforced Concrete Structural Detailing

Akilu Muhammad<sup>1</sup>, Ibrahim Ogiri Hassan<sup>2</sup> <sup>1</sup>Faculty of Civil Engineering,University Teknologi Malaysia,Skudai, Johor Bahru. <sup>2</sup>Department of Building Technology,Federal University of Technology,Minna, Nigeria Correspondence email; amty72@yahoo.com, +6049126263

Abstract— AutoCAD, the popular software for Architectural and Engineering design is increasingly used today to achieve and speed and accuracy in work execution. Complex and repetitive drawings are automated through AutoCAD programming to optimise the design process. In this research, AutoLISP (AutoCad language), is used to programme AutoCAD for automated reinforced concrete structural detailing. Reinforced concrete Open-well staircase is used as a case study. AutoLISP customization features are used to define the coordinates of any point the software uses to draw any element in the detailing. AutoLISP functions were used in defining function of routines, obtaining user input, defining the drawing pattern and changing the system variables were necessary before using the command function to produce the desired drawing. Accuracy and efficiency is achieved in the use of the developed software and user effort is highly minimized and simplified.

**Keywords**—AutoCAD ; AutoLISP; Structural Detailing; Openwell staircase; Programming.

### Introduction

A good drawing they say speaks thousands of words. Structural Detailing is an essential aspect of structural design work; it provides graphical representation of structural design details for ease of understanding of project requirements between design team and construction team in a construction project. As structural detailing work becomes complex, tedious and repetitive, a logical solution that simplifies the detailing work is automation of the detailing process. This is achieved through customization of AutoCAD drawing task in its built-in development environment called Visual LISP. One of the language of AutoCAD programming is AutoLISP and it is a specialized implementation of LISP programming language used in artificial intelligence. Development of AutoLISP programme for AutoCAD is carried out in a text editor, Visual LISP, it is then loaded and run in AutoCAD<sup>1</sup>. Interactive programs are developed that interact with user to obtain necessary information needed to achieve a drawing task. This enhances speed and accuracy in design and detailing, design task that takes hours to accomplish can be achieved in a very small fraction of the time.

This paper presents a research work carried out on automation of AutoCAD for reinforced concrete structural detailing using AutoLISP programming language. A case study on a programme for interactive open-well staircase drawing is presented. AutoCAD is customized to instantly produce reinforced concrete structural detailing on receiving instructions relating geometry and other parameters.

Autolisp in Autocad

AutoCAD is acknowledged to be the most powerful and widely used technical drawing software that was designed with an open architecture so that you can customize and extend its many features<sup>2</sup>. AutoCAD introduced AutoLISP as an application programming Interface (API) in release 2.1 in the mid-1980s. LISP was chosen as the initial AutoCAD API because it was uniquely suited for the unstructured design process of AutoCAD projects, which involved repeatedly trying different solutions to design problems<sup>3</sup>. AutoLISP has set the standard for customizing and extending AutoCAD functionality to the extent that one can adapt AutoCAD to ones need by automating repetitive tasks, creating new AutoCAD commands, redefining existing commands, creating new programs or use third party shareware programms<sup>4</sup>.

**III Automation of structural detailing** case study of open well staircase AutoLISP customization features were used to develop a programme for automated detailing of open-well staircase.

The steps in program design involve (1) Establishment of the name of the program or function, (2) Obtaining drawing information form user. (3) Processing user information, (4) Production of the desired drawing.

The name of the program or function is first established. This is used in running the program or to instruct the program to carry out specific task. User is then prompted for input relating to geometrical and other parameters which are processed in line with instructions in the source code to produce the drawing.

The Flowchart on figure 1.0 shows that the program will start by seeking user input regarding name, geometry and other parameters of stair to be drawn and starting point of drawing. It will follow instructions in the source code to process and draw stair plan (strplan), then draw first line of plan of step (sstepl) and create of space between the first line of step and the next to be drawn (sstep). A loop command is then issued to make the program perform a test and decide whether the test is less than number of steps plus one. If the answer to the test is yes, sstepl and sstep will be repeated and another test conducted, the process continues until the number of tests carried out is not less than nsteps plus one. The next (opta) routine offsets the position of pta (the starting point of the drawing) to a point where the first step of the second flight is located. Following the same pattern as above, a similar loop command is given to draw the plan of steps for the second flight (sstepl1) and to space the steps (sstep1). The new position of pta is again moved to commencement point of third flight by the routine (opta1) before once again, issuing a similar loop command to draw the plan of steps for third flight (sstep12) and space the steps (sstep2).

Stair section for third flight (strsection2) is processed, drawn and dimensioned. Following the same pattern, series of loop commands were issued for the drawing and spacing of section of



International Journal of Scientific Engineering and Technology Volume No.4 Issue No.4, pp: 273-276

**LJS** stair steps for third flight (stnstep2 and sstnstep2), drawing and spacing of fourth landing transverse continuity reinforcement (Tttcreinf and sTttcreinf),drawing and spacing of third flight transverse reinforcement (Tfltreinf and sTfltreinf), drawing and spacing of third landing transverse reinforcement (Tltreinf and sTltreinf), drawing and spacing of fourth landing transverse reinforcement (frltreinf and sfrltreinf). Another decision is taken by the program on the support type (lstype) to draw, if the user chooses beam it draws beam support (sptbeam2), if he chooses wall it draws wall support (sptwall2).

The program then draws the second flight in a similar manner as for the third flight. It starts with the stair section (strsection1) before following a similar loop command as in the third flight to draw section of stair steps and their spacings (sfstnstep and sfsstnstep), draw and space third landing continuity transverse reinforcement (sfttreinf, sfsttreinf), draw and space second flight transverse reinforcement (sffltreinf, sfsfltreinf), draw and space third landing transverse reinforcement (sultreinf,ssultreinf), draw and space second landing transverse reinforcement (slltreinf sslltreinf). It will then draw support stair support type according to user's choice either beam support (sfsptbeam) or wall support (sfsptwall).

The program then draws the first flight in a similar manner as for the third flight and second flights. It starts with the stair section (strsection) before following a similar loop command to draw section of stair steps and their spacings (stnstep and sstnstep), draw and space second landing continuity transverse reinforcement (sfttreinfFR, sfsttreinfFR), draw and space first flight transverse reinforcement (sffltreinfFR, sfsfltreinfFR), draw and space second landing transverse reinforcement (sultreinfFR, sultreinfFR), draw and space first landing transverse reinforcement (slltreinfFR sslltreinfFR). It will then draw support stair support type according to user's choice either beam support (sptbeam) or wall support (sptwall). The program finally comes to an end.

Fig. 2 shows the drawing of the staircase in plan and elevation. In using the software, AutoCAD drawing environment need not to be set in terms of text size, dimension text size, dimension arrow size, extension line offset, dimension extension line extension, text vertical position etc., as all these settings were built into the developed software which can detail Open-well stair of any number of steps.

#### IV Conclusion

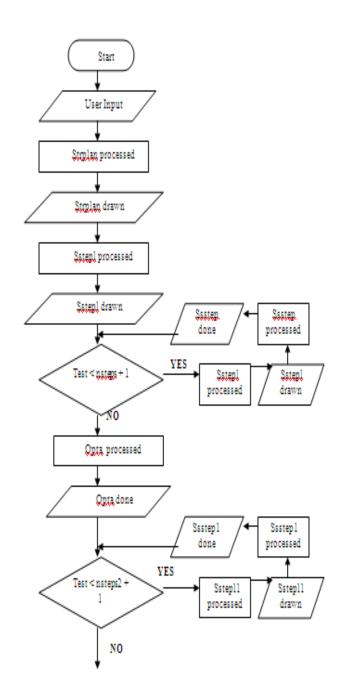
The use of AutoLISP in programming AutoCAD for automated structural detailing has been demonstrated in this paper. Reinforced concrete detailing of Open-well staircase was used as a case study in developing an interactive programme that prompts its user to obtain information relating to geometry needed to achieve the drawing task.

Once the program is loaded using its program name, the program bets for necessary user input at AutoCAD command line to enable her produce the desired drawing (see Fig 2). The program was tested and debugged to ensure its accuracy and reliability. In running the program there is no need to set the AutoCAD drawing area in terms of text size, dimension line and arrow sizes, and extension line offset, extension line extension, text

IJSET@2015

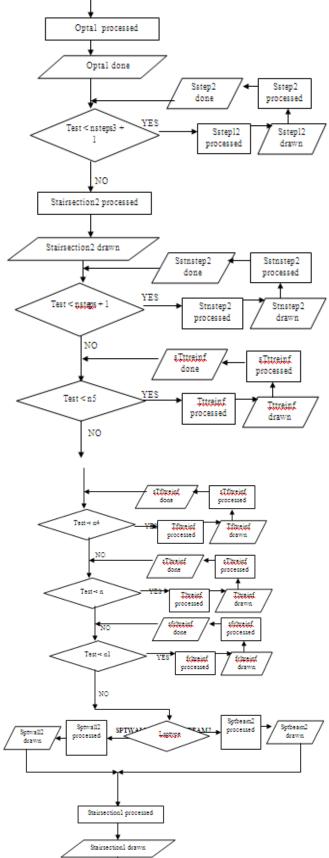
ISSN : 2277-1581 01 April. 2015

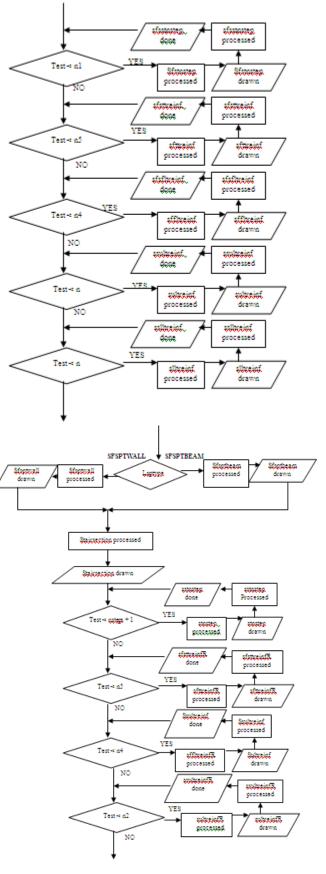
vertical position as all these settings have been built into the program.





# International Journal of Scientific Engineering and Technology Volume No.4 Issue No.4, pp: 273-276





IJSET@2015



International Journal of Scientific Engineering and Technology Volume No.4 Issue No.4, pp: 273-276

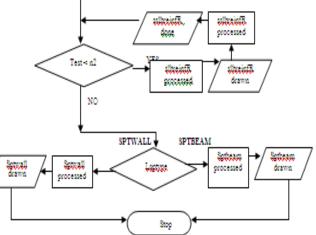
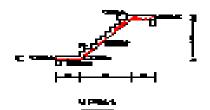
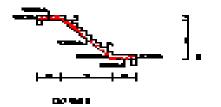


FIG. 1.0: Flowchart for Open-well Staircase Detailing Software







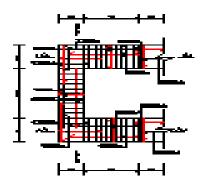


Fig. 2. Open-well Staircase as produced by the programme

## REFERENCES

i. [1 ]Auto desk (2012) AutoLISP Developer's Guide, AutoCAD 2013, Carlifornia

ii. [2] E. Fikelstein (2002) The Bible of AutoCAD. Hungry Minds, New York.

iii. [3] AutoDesk, (2002) AutoLISP Developer Documentation,

iv. AutoCAD 2002 Software, California.

v. [4] G. Omura, (1990) The ABC's of AutoLISP. Syber, Albany.