World Academy of Science, Engineering and Technology
International Journal of Industrial and Manufacturing Engineering
Vol:6, No:12, 2012

An Automated Approach for Assembling Modular Fixtures Using SolidWorks

Uday Hameed Farhan, Majid Tolouei-Rad, and Simona O'Brien

***Abstract*—**Modular fixtures (MFs) are very important tools in manufacturing processes in terms of reduction the cost and the production time. This paper introduces an automated approach for assembling MFs elements by employing SolidWorks as a powerful 3D CAD software. Visual Basic (VB) programming language was applied integrating with SolidWorks API (Application programming interface) functions. This integration allowed creating plug-in file and generating new menus in the SolidWorks environment. The menus allow the user to select, insert, and assemble MFs elements.

***Keywords*—**Assembly automation, modular fixtures, SolidWorks, Visual Basic.

I. Introduction

DURING the development of CAD/CAM activities for manufacturing processes, many relevant issues need to be addressed including the fixture design process for efficient machining. The traditional fixture design process cannot adapt to the rapid development of manufacturing technologies and their equipment, especially CNC machines [1]. The product quality, short production period, cost, and efficient delivery are the main goals for manufacturing processes. To achieve these goals, fixture design is considered as a vital factor that has a direct influence in machining operations [2]. Computer Aided-Fixtures Design (CAFD) was the solution for complex fixture design processes [1]. The initial use of CAFD was to apply CAD software as a tool for designing and assembling fixture elements with implementing a standard fixture library [3]. However, automated CAFD systems have become more significant in manufacturing processes since the CAM activities have improved rapidly in recent decades. Automated CAFD allow the user to define the feasible fixture configuration including, locating methods and clamping mechanisms and layout for a given workpiece. Different approaches have been used for this purpose by applying CAD software.

Babu, Valli, Kumar, and Rao [4] implemented an automated modular fixture design system that used an AutoCAD platform with AutoLISP program based on 2D drawings. Their system introduced the methodology for the machining setup and the modular fixture automation sequence. Ma, Lei, and Rong [5] developed a fixture design system called FIX-DES by using C++ programs and a specific CAD environment. The requirements of modular fixture configuration were implemented and the fixture elements were classified based. Assembly relationships were used in this system. Kong, Yangyi, JIe, Gue, Zhang, and Zhao [6] improved CAFD system based on an AutoCAD software platform. VC++ and AutoLISP languages were employed in this system and the database of the standard fixture elements was presented.

Most of the studies are focused on the 2D design concept. However, the 3D modeling approach is a key factor in developing the assembly relationships between the fixture elements and the workpieces in the fixture design process. This approach should consider all of the geometric and machining features of the workpiece. SolidWorks software is implemented in the previous study [7] as a CAD environment for assembling MFs elements and constructing the database due to its strong 3D modeling and assembly capabilities.

In this study, SolidWorks secondary development with VB has been implemented to automate the assembly process of MFs. A generated VB project is integrated with SolidWorks API functions to create new menus in the SolidWorks environment. The menus have been classified for Side /Top clamping and for selecting the fixture elements. User interfaces have been developed for inserting and assembling the fixture elements. The flowchart for the developed system is shown in Fig. 1.

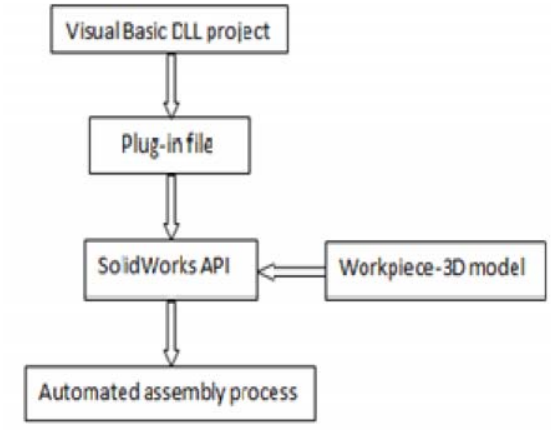


Fig. 1 The developed approach by SolidWorks secondary development

II. MFs Structure

In modular fixture assembly system, an individual fixture element cannot play the role alone; it must be assembled to

Uday Farhan, Postrgraduate Student, School of Engineering, Edith Cowan University, 270 Joondalup Drive, Joondalup WA, 6027, Australia.
Majid Tolouei-Rad, Senior Lecturer, School of Engineering, Edith Cowan University, 270 Joondalup Drive, Joondalup WA, 6027, Australia (m.rad@ecu.edu.au).
Simona O'Brien, Postrgraduate Student, School of Engineering, Edith Cowan University, 270 Joondalup Drive, Joondalup WA, 6027, Australia.

another element for defining the relationship and to define the aim of this relationship. For example, a clamp must be assembled to a baseplate in order to perform clamping function. Moreover, fixture elements can be assembled in groups, these groups are called "functional units". Each functional unit consists of one or more elements and these units are responsible for supporting the workpiece to the baseplate. The functional units can be divided into three types according to their functions, namely locating, clamping and supporting units [8]. Fig. 2 shows an example of the fixture unit. The structure of MFs is presented in Fig. 3.

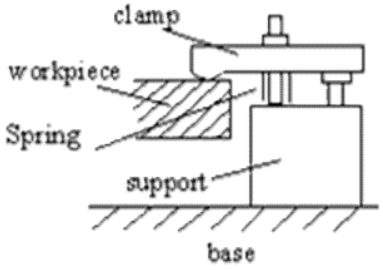


Fig. 2 A typical sample of fixture unites [9]

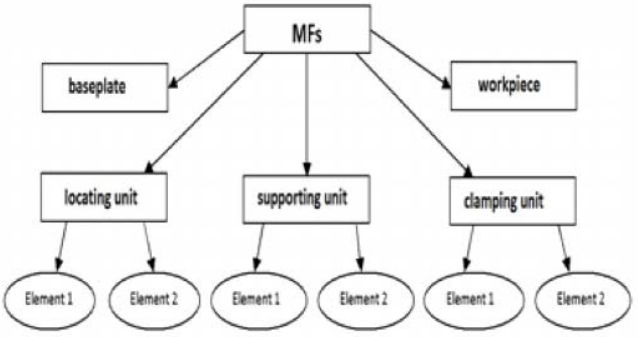
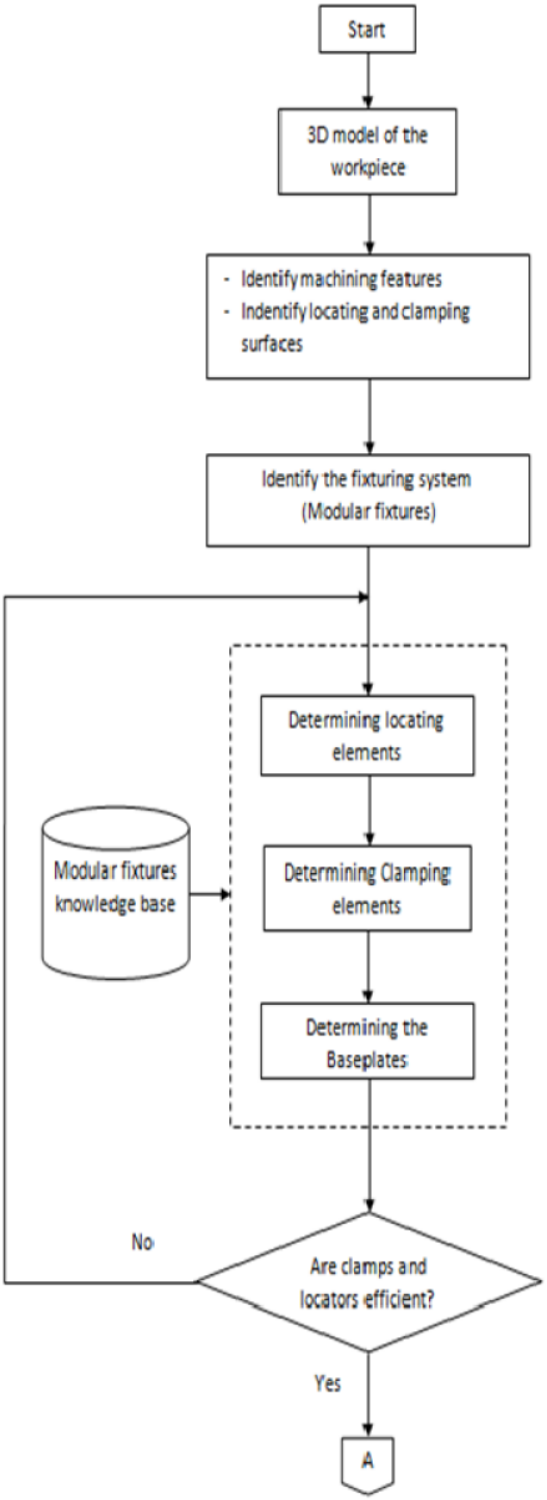


Fig. 3 A proposed modular fixture structure to assemble a workpiece to a base plate [7]

III. The Automated Approach

A 3D design system for MFs has been developed in the previous study [7]. The system applied in SolidWorks to select the fixture elements and assemble them by using assembly tools and mating features. The system has been extended in order to automate the selection and the assembly process of MFs (Fig. 4). The selection process includes determining the locating\clamping methods and their elements with the baseplate. For this purpose, the knowledge base has been established to meet the requirements for achieving the feasible layouts for MFs. After that, the fixture elements are assembled with defining the appropriate assembly relationships knowledge base between the elements.



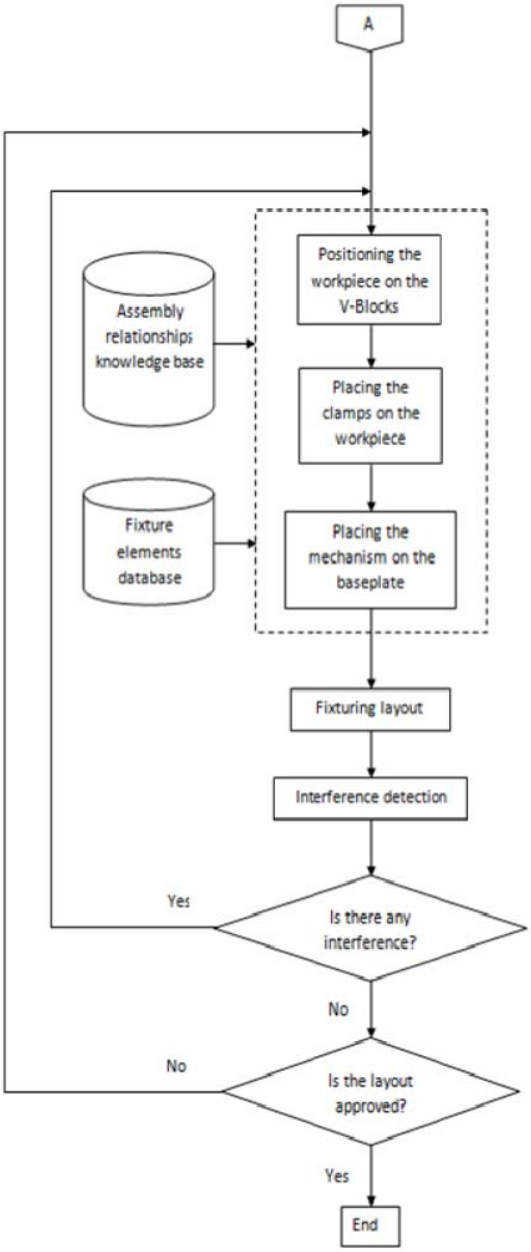


Fig. 4 Flowchart of the MFs system

IV. Visual Basic Project

In order to automate the assembly process of MFs, VB DLL project has been created and integrated with SolidWorks API. The project included SolidWorks libraries as references; these are *SldWorks Type* Library and *SolidWorks Exposed Type* library. Adding these libraries to the DLL project allows control SolidWorks commands and functions.

The result of this integration is add-in code to add new menus in the SolidWorks environment for selecting and assembling the fixture elements. The add-in code is shown below:

```
'Tell VB that you are going to provide functionality for the SwAddin interface
Implements SWPublished.SwAddin
'Declarations for addin SW connection
Dim axSldWorks As SldWorks.SldWorks
Dim axCookie As Long 'holds value created in SwAddin_ConnectToSW
'Cookie needed for menus, toolbars, CallbackInfo
Dim axToolbarID As Long 'toolbar ID if toolbars used
Dim axActiveDoc As SldWorks.ModelDoc2
Dim axTargetDoc As SldWorks.ModelDoc2
Private Function SwAddin_ConnectToSW(ByVal ThisSW As Object, ByVal Cookie As Long) As Boolean
Dim bRet As Boolean 'boolean return
Dim lRet As Long 'long return
Dim axMenuID As String
Dim lngToolbarDocTypes As Long
'1. capture SW session and cookie to class-wide variable
' store reference to SW session
Set axSldWorks = ThisSW
' store cookie from SW
axCookie = Cookie
2. set Addin Callback info
'Inform SW about the object that contains the callbacks
bRet = axSldWorks.SetAddinCallbackInfo(App.hInstance, Me, axCookie)
'App.hInstance = program's handle
'Me = this class module
3. add menus
axMenuID = "MFs"
 lRet = axSldWorks.AddMenu(swDocASSEMBLY, axMenuID, 6)
 'b. set up menu strings
 'we have two functions, we need two menu picks
Dim axMenu1 As String
axMenu1 = "MFs Form@MFs@" & axMenuID
'c. add menus using axSldWorks.AddMenuItem2
 'this call needs to be made once for each document type:
'NONE, Part, Assembly, and Drawing
'PARTS:
bRet = axSldWorks.AddMenuItem2(swDocPART, axCookie, axMenu1, 0, "CallMfsForm", "EnableIfAssembly", "Launch MFs Form.")
'ASSEMBLIES:
bRet = axSldWorks.AddMenuItem2(swDocASSEMBLY, axCookie, axMenu1, 0, "CallMfsForm", "EnableIfAssembly", "Launch MFs Form.")
 '---------------------------------
'from SW API documentation:
'IsMenuItemAdded = SldWorks.AddMenuItem2(DocumentType, Cookie, MenuItem, position, MenuCallback, MenuEnableMethod, HintString)
'MenuCallback: this is the subroutine that will be called by the menu
'This subroutine must reside in this class module.
'MenuEnableMethod:
'function that returns a long value 0 to 3
'must reside in this class module
.
.
.
.
```

Fig. 5 shows the developed menus. User interfaces have been also developed to make the process more flexible. An example for that is shown in Fig. 6 which reeled to the selection and inserting the appropriate baseplate.

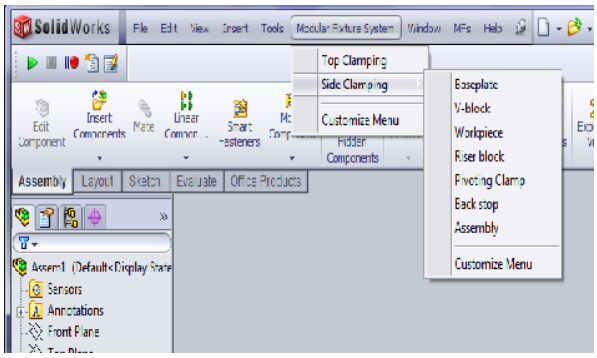


Fig. 5 The new developed menus

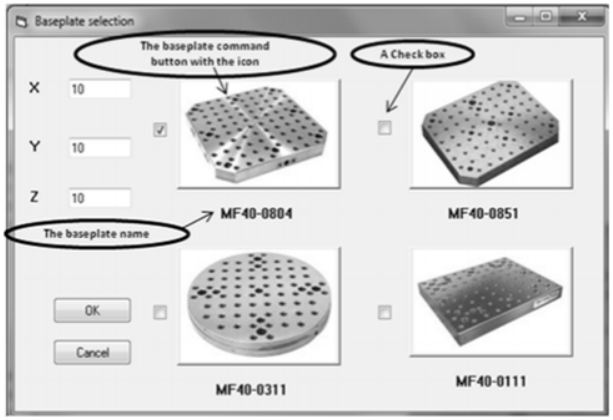


Fig. 6 The user interface for the base plate

V. Case Study

The developed system can be applied for different MFs configurations for side and top clamping. In this paper, the system has been used to assemble MFs elements for a specific workpiece as shown in Fig. 7. The system has been designed to be used with quick change subplates for CNC machines for several manufacturing process.

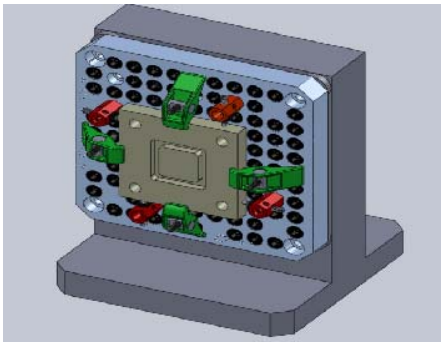


Fig. 7 MFs layout for quick change subplates for CNC machines

VI. Conclusion

Developing an automated modular fixture assembly system is the major objective of this paper. The automating process includes the integration of SolidWorks and Visual. SolidWorks is considered as the core of the developed as powerful CAD software due to its 3D modeling capabilities for design and assembly processes. Similarly, Visual Basic made its attractive as a programming language in this study.

SolidWorks API is employed with VB to create menus in SolidWorks in order to automate the assembly process. Moreover, user interfaces also developed for more flexibility. The system can be used for a variety of workpieces and for different MFs configurations in order to achieve the feasible layout for machining operations.

References

[1] L. Kailing, *et al.*, "Development of an intelligent jig and fixture design system," in *7th International Conference on Computer-Aided Industrial Design and Conceptual Design*, 2006, pp. 1-5.

[2] I. M. Boyle and R. Kevin, "CAFixD: A Case-Based Reasoning Fixture Design Method. Framework and Indexing Mechanisms," presented at the ASME 2004 design engineering technical conferences, Salt Lake,Utah USA, 2004.

[3] Y. Rong and X. Li, "Locating method analysis based rapid fxiture configuration design," *IEEE*, pp. 27-32, 1997.

[4] B. S. Babu, *et al.*, "Automatic modular fixture generation in computer aided process planning systems," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, pp. 1147-1152, 2005.

[5] W. Ma, *et al.*, "FIX-DES:A computer - aided modular fixture configuration design system," *Advanced manufacturing technology*, vol. 14, pp. 21 - 32 1998.

[6] X. Kong, *et al.*, "Reaserch and development of the software on computer aided fixtures designing," *IEEE*, pp. 1233-1236, 2009.

[7] U. H. Farhan and M. T. Rad, "Design of modular fixtures using a 3D-modelling approach," in *In:19th International Congress on Modelling and Simulation*, Perth, Australia, 2011, pp. 405-411.

[8] G. Peng, *et al.*, "A pragmatic system to support interactive modular fixture configuration design in desktop virtual environment," *IEEE*, vol. 2, 2008.

[9] *Development of Automated Dedicated Fixture Design Systems*. Available: http://www.me.wpi.edu/research/CAMLab/research%20areas/new/ded_principle.htm