Integration of CAD model with Calculation Sheet using Excel VBA

Shubham Singh[1], Roopak Rawat[2], Kuldeep[3]

[1-3]B.Tech student, Dept. of Mechanical Engineering, Delhi Technological University, Delhi, India

---***---

Abstract - *Automation is a technique to do a job without or less human assistance, and this techniques saves our lots of time and ultimately increases our productivity. Hence that is faster and more effective way of doing a job. In this publication we automated the design and modeling process of Cotter and Spigot joint, which includes the calculation behind the product design as well as the provision for auto updating in Computer Aided Design (CAD) software PTC CREO. We used Excel VBA for programing the algorithm of calculation behind the worksheet. And also made a UserForm that interact with user to collect the model input design parameters and VBA process the data as per the calculation or logic fed into the Excel VBA. The output from this excel sheet is fetched directly to CAD model's dimensions for auto updating the 3D model.*

Key Words: Automation, CAD modelling, Excel VBA, Cotter and spigot joint.

1. INTRODUCTION

CAD (Computer Aided Design) software are widely used in industries to create 2D and 3D designs of a product. It uses computer for virtual and digital development of product. Thus it is the most efficient way of creating, modifying or optimizing the model (Narayan and Lalit, 2008).

A PTC CREO is used for executing this automation work. PTC CREO is a parametric and feature based modeling software that allows to create user defined parameters and assigning those values to the dimensions of model for changing its shape and size. And for constraint based modeling we can also setup relations between input and output values so that the model is being modified with satisfying the various constraints. The geometry of model itself can modify by changing the numerical values of the parameters and there is no need to re-edit definition of the model. Therefore parametric modeling saves great time in modification for the designer especially in case of live project where things are in more dynamic and changing continuously till the end. With design module for the part in which we create number of required parameters and set the relation between them, we can get infinite variation by altering its input parameters, that base model is called "master part" (PTC creo, 2020; Fu, 2018).

VBA (Visual Basic for Application) is a programming language associated with the Microsoft applications like MS Excel, MS Word etc. It enhance the capability of doing job in more automated manner with Microsoft applications (VBA visual basic for application, 2016).

MS Excel is most common and widely used in small to large scale industries, NGOs, schools and colleges etc. (Microsoft Excel – A Handy Tool Used in Various Areas of Business – Home, 2016). We can automate our various design task by writing the program in VB editor known as "Macro" (Romeo, 2020). VB Editor (Visual Basic Editor) is working area which include the window of writing code, debugging code and code module organization environment. So it can allow us to write user defined functions and can call it anywhere within the Microsoft applications. In MS excel VB Editor can be accessed under the developer tab (VBA visual basic for application, 2016; Romeo, 2020).

VB Editor also have an option of UserForm, which is an impressive and friendlier way to interact with users. This UserForm can be customized as per the required input cell type (either combo box, list box, check box, option button, toggle button or scroll button etc.). So this is the most efficient and controlled way of taking input from the user (Excel Macros - UserForms – Tutorialspoint, 2021).

Cotter and spigot consist of three main components (cotter, socket and spigot). It is used to connect two co-axial shaft against the axial load (push or pull in nature). The wedge shape cotter itself develops tightening force and this force prevents from loosening during operation. It is very simple in construction and can be easily manufactured (Bhandari, 2017).

In complete design module of cotter and spigot joint, a master assembly is made with set of constraints. It can be further automated by auto updating the numerical values of each model parameters instead of doing manually.

The whole VBA project can be password protected for developer authentication. So that only authorized person who has knowledge of design can edit or view the algorithm of the design calculation in VB editor by entering the password at once.

2. METHOD

In order to make complete design module of cotter and spigot joint, design analysis of the cotter joint is done in excel VBA. Based on various possible failure conditions, we can find the following dimensions of the cotter joint as shown in figure 1.

Where, d = Diameter of the rods, d1 = Outside diameter of socket, d2 = Diameter of spigot or inside diameter of socket, d3 = Outside diameter of spigot collar, d4 = Diameter of socket collar, t1 = Thickness of spigot collar, c = Thickness of socket collar, b = Mean width of cotter, t = Thickness of cotter, l = Length of cotter, a = Distance from the end of the slot to the end of the rod.

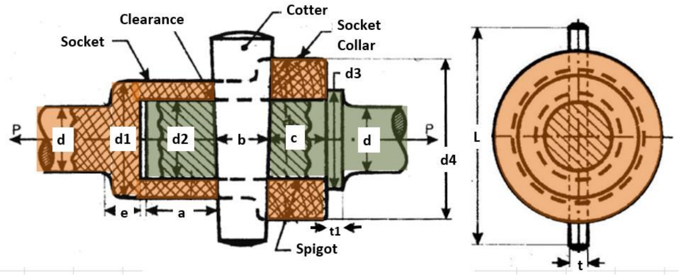


Fig - 1: Cotter and Spigot Joint

2.1 Functions Created in VBA

A function is a block of reusable codes. Which can be called anywhere within the Microsoft application by its name with parameters list (VBA visual basic for application, 2016).

The methodology of breaking main program into the functions of various failure conditions that makes it easier to understand and edit whenever required in future. Hence during the time of rectification, we can easily and quickly identify the area of correction by searching their function names. And if required to add some more design criteria, we can create a new function and can call it into our main program, that doesn't need to change or alter the whole main program.

In case of cotter and spigot joint there are total 9 cases of failure (Bhandari, 2017), for that we define the different functions in VBA. And apart from this we have used inbuilt "ceiling" function of VBA for jump the numerical value to the next multiple of given number.

2.1.1 Function for failure of rods in simple tension

```
Function rodsdia(p As Double, stress As Double, fos As Double) As Double
rodsdia = Sqr(4 * p * fos / (4 * Atn(1) * stress))          4
* Atn(1) is value of Pi
End Function
```

2.1.2 Function for failure of spigot at the weakest section (having slot)

```
Function spigotdia(p As Double, stress As Double, fos As Double, d As Double) As Double
Dim a As Double, b As Double, c As Double, d1 As Double, d2 As Double
c = -(fos * p) / stress
a = (4 * Atn(1)) / 4
b = -0.4 * d     'thickness of cotter is taken as 0.4*rod dia
d1 = (-b + Sqr(b ^ 2 - 4 * a * c)) / (2 * a)
d2 = (-b - Sqr(b ^ 2 - 4 * a * c)) / (2 * a)
If d1 > 0 And d2 > 0 Then
If d1 < d2 Then
spigotdia = d1
Else
spigotdia = d2
End If
ElseIf d2 > 0 Then
spigotdia = d2
ElseIf d1 > 0 Then
spigotdia = d1
End If
```

2.1.3 Function for failure of cotter or rod in crushing

```
Function crushtest(d2 As Double, t As Double, stress As Double, p As Double, fos As Double) As Boolean
Dim p1 As Double
p1 = d2 * t * stress / fos
If p1 > p Then
crushtest = True
Else
crushtest = False
End If
End Function
```

If crush test of cotter or rod is failed then the thickness of cotter is increased by 0.01 times of the diameter of rod until the crush test case is passed. Thus the spigot diameter "d2" needs to calculate again using new value of cotter thickness.

Function for calculate new spigot diameter "d2" with argument "t" instead of "d".

```
Function newspigotdia(p As Double, stress As Double, fos As Double, t As Double) As Double
Dim a As Double, b As Double, c As Double, d1 As Double, d2 As Double
c = -(fos * p) / stress
a = (4 * Atn(1)) / 4
b = -t
d1 = (-b + Sqr(b ^ 2 - 4 * a * c)) / (2 * a)
d2 = (-b - Sqr(b ^ 2 - 4 * a * c)) / (2 * a)
If d1 > 0 And d2 > 0 Then
If d1 < d2 Then
            newspigotdia = d1
Else
newspigotdia = d2
End If
ElseIf d2 > 0 Then
newspigotdia = d2
ElseIf d1 > 0 Then
newspigotdia = d1
End If
End Function
```

2.1.4 Function for failure of socket in simple tension at weakest section

```
Function socketoutsidedia(p As Double, stress As Double, fos As Double, d2 As Double, t As Double) As Double
Dim a As Double, b As Double, c As Double, od1 As Double, od2 As Double
c = -(((p * fos) / stress) + 4 * Atn(1) * d2 ^ 2 * 0.25 - d2 * t)
a = 4 * Atn(1) * 0.25
b = -t
od1 = (-b + Sqr(b ^ 2 - 4 * a * c)) / (2 * a)
od2 = (-b - Sqr(b ^ 2 - 4 * a * c)) / (2 * a)
If od1 > 0 And od2 > 0 Then
If od1 < od2 Then
socketoutsidedia = od1
Else
socketoutsidedia = od2
```

```
    End If
    ElseIf od2 > 0 Then
    socketoutsidedia = od2
    ElseIf od1 > 0 Then
    socketoutsidedia = od1
    End If
    End Function
```

2.1.5 Function for failure of cotter in double shear

```
    Function cotterwidth(p As Double, fos As Double, t As
Double, stress As Double) As Double
    cotterwidth = (p * fos) / (2 * t * stress)
End Function
```

2.1.6 Function for failure of socket collar in crushing

```
    Function socketcollardia(p As Double, stress As Double,
fos As Double, t As Double, d2 As Double) As Double
    socketcollardia = ((p * fos) / (stress * t)) + d2
    End Function
```

2.1.7 Function for failure of socket in double shearing

```
    Function socketcollarthick(p As Double, stress As Double,
fos As Double, d4 As Double, d2 As Double) As Double
    socketcollarthick = (p * fos) / (2 * stress * (d4 - d2))
    End Function
```

2.1.8 Function for failure of rod in double shearing

```
    Function rodlengtha(p As Double, stress As Double, fos As
Double, d2 As Double)
    rodlengtha = (p * fos) / (2 * d2 * stress)
    End Function
```

2.1.9 Function for failure of spigot collar in crushing

```
Function spigotcollardia(p As Double, fos As Double, stress
As Double, d2 As Double) As Double
spigotcollardia = Sqr((4 * p * fos / (4 * Atn(1) * stress)) + d2
^ 2)
End Function
```

2.1.10 Function for failure of spigot collar in shearing

```
    Function spigotcollarthik(p As Double, fos As Double,
stress As Double, d2 As Double) As Double
    spigotcollarthik = (p * fos) / (4 * Atn(1) * d2 * stress)
    End Function
```

2.1.11 Function for failure of cotter in bending

```
    Function cotterbendingcheck(p As Double, d2 As Double,
d4 As Double, t As Double, b As Double, stress As Double) As
Boolean
    Dim m As Double, z As Double, s As Double
    m = p * 0.5 * ((d4 - d2) / 6 + d2 / 4)
    z = t * b ^ 2 / 6
    s = m / z
    If s < stress Then
    cotterbendingcheck = True
    Else
    cotterbendingcheck = False
    End If
    End Function
```

If cotter is failed in bending check then need to recalculate the mean width of cotter "b" as per the cotter bending criteria.

Function for calculate new cotter width "b" against bending.

```
    Function newcotterwidth(p As Double, d2 As Double, d4
As Double, t As Double, stress As Double, fos As Double) As
Double
    Dim b As Double
    b = Sqr(p * (d4 + 0.5 * d2) * fos / (2 * t * stress))
    newcotterwidth = b
    End Function
```

2.2 Main Body of VBA Program

In main body of program we call the various created functions in the required sequence of design and then the VBA program process these data, which update the values in main excel sheet. Figure 2 shows the complete flow process of Excel program. The subroutine named "compute" consist of design logic statement and it execute when "COMPUTE" button is clicked in UserForm.

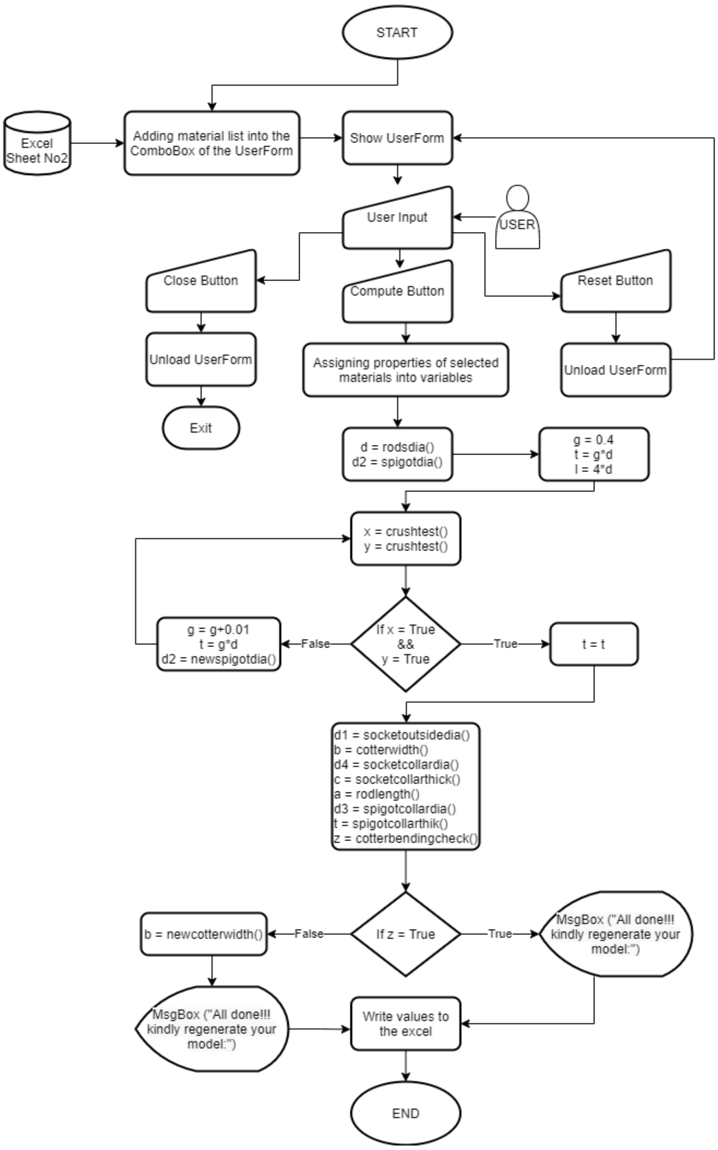


Fig - 2: Main Program Flow Chart

```
    sub compute()
```

```
Dim d As Double, d2 As Double, d1 As Double, b As Double, d4 As Double, c As Double, ts_s As Double, a As Double, d3 As Double, t1 As Double, l As Double, t As Double, load As Double, cs_s As Double, x As Boolean, y As Boolean, z As Boolean
Dim ts_c As Double, cs_c As Double, ss_c As Double, fos As Double, ss_s As Double
On Error GoTo here
load = cotterdesign.load.Value            'cotterdesign is name of UserForm
fos = cotterdesign.fos.Value      'Combocotter and Combosocket are names of ComboBox for cotter and socket material selection respectively
ts_c = Sheets(2).Range("B" & cotterdesign.Combocotter.ListIndex + 2).Value
ts_s = Sheets(2).Range("B" & cotterdesign.Combosocket.ListIndex + 2).Value
cs_c = Sheets(2).Range("C" & cotterdesign.Combocotter.ListIndex + 2).Value
cs_s = Sheets(2).Range("C" & cotterdesign.Combosocket.ListIndex + 2).Value
ss_c = Sheets(2).Range("D" & cotterdesign.Combocotter.ListIndex + 2).Value
ss_s = Sheets(2).Range("D" & cotterdesign.Combosocket.ListIndex + 2).Value
d = Application.WorksheetFunction.Ceiling(rodsdia(load, ts_s, fos), 5)
d2 = Application.WorksheetFunction.Ceiling(spigotdia(load, ts_s, fos, d), 5)
g = 0.4
t = g * d
l = 4 * d
        'check rod or cotter in crushing
crushcheck:
x = crushtest(d2, t, cs_c, load, fos)
y = crushtest(d2, t, cs_s, load, fos)
If x = True And y = True Then
t = t
Else
g = g + 0.01
t = g * d
d2 = Application.WorksheetFunction.Ceiling(newspigotdia(load, ts_s, fos, t), 5)
GoTo crushcheck
End If
d1 = Application.WorksheetFunction.Ceiling(socketoutsidedia(load, ts_s, fos, d2, t), 5)
b = Application.WorksheetFunction.Ceiling(cotterwidth(load, fos, t, ss_c), 1)
d4 = Application.WorksheetFunction.Ceiling(socketcollardia(load, cs_s, fos, t, d2), 1)
c = Application.WorksheetFunction.Ceiling(socketcollarthick(load, ss_s, fos, d4, d2), 1)
a = rodlengtha(load, ss_s, fos, d2)
d3 = Application.WorksheetFunction.Ceiling(spigotcollardia(load, fos, cs_s, d2), 1)
t1 = Application.WorksheetFunction.Ceiling(spigotcollarthik(load, fos, ss_s, d2), 1)
z = cotterbendingcheck(load, d2, d4, t, b, ts_c)
If z = True Then
MsgBox ("All done!!! Kindly regenerate your model :")
Else
b = Application.WorksheetFunction.Ceiling(newcotterwidth(load, d2, d4, t, ts_c, fos), 1)
MsgBox ("All done!!! Kindly regenerate your model :")
End If
Sheets(1).Range("I21").Value = load
Sheets(1).Range("I22").Value = d
Sheets(1).Range("I23") = d1
Sheets(1).Range("I24") = d2
Sheets(1).Range("I25") = d3
Sheets(1).Range("I26") = d4
Sheets(1).Range("I27") = t1
Sheets(1).Range("I28") = c
Sheets(1).Range("I29") = b
Sheets(1).Range("I30") = t
Sheets(1).Range("I31") = l
Sheets(1).Range("I32") = a
Exit Sub
here:
MsgBox ("An error occurred :")
End Sub
```

2.2 UserForm

UserForm is a dialogue box designed to interact with user to take the input values instead of excel cells to the main program as shown in figure 3. It makes the interface more user friendly and thus reduces the chance of error while feeding the values (Excel Macros - UserForms – Tutorialspoint, 2021).

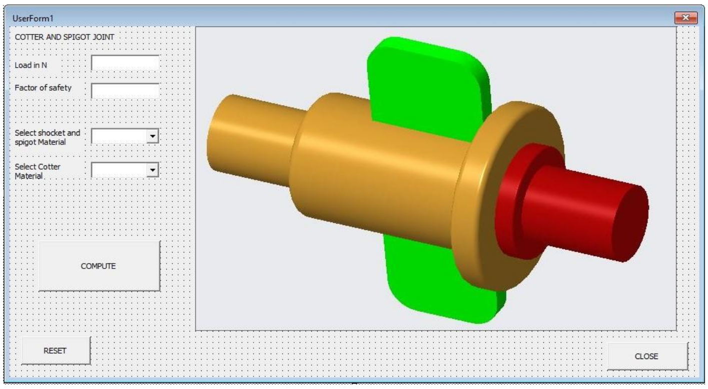
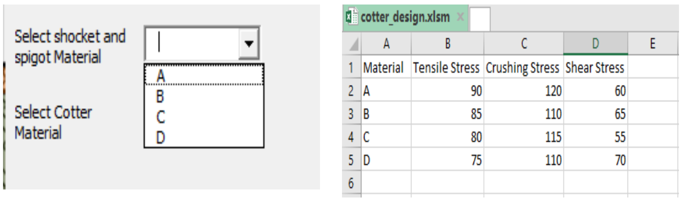


Fig - 3: UserForm

The start command button is placed on the main sheet of excel and assigned the following macro to show UserForm and also call subroutine named "combobox" to update materials list.

```
Sub runform()
Call combobox 'combobox named subroutine is called for updating the material list from the 2nd sheet of same excel into the UserForm for material selection
cotterdesign.Show
End Sub
```

We comprise dropdown option for material selection by its name as shown in figure 4(a), so that the user doesn't need to enter each properties of the material in UserForm which are required to design the cotter joint. The list of material which are appearing in the option of dropdown button are fetched from the second sheet of the same excel, where the material's name and its required properties are added to designated cells as shown in figure 4(b). Therefore it gives a flexibility to add new material anytime into part model without changing the program. After adding new material into material data base (sheet 2) and then this will instant appear in UserForm in next run.



	A	B	C	D	E
1	Material	Tensile Stress	Crushing Stress	Shear Stress	
2	A	90	120	60	
3	B	85	110	65	
4	C	80	115	55	
5	D	75	110	70	
6					

Fig -4: Material Selection: (a) UserForm Dropdown button, (b) Material Database.

Subroutine named "Combobox" which count the number the filled cells in column A of sheet 2, then it add material's name one by one into the dropdown list of user form. This subject code runs at every time before showing UserForm when subroutine "runform" is run.

```
Sub combobox()
Dim i As Integer, a As Integer
a = WorksheetFunction.CountA(Sheets(2).Columns("A:A"))
For i = 1 To a - 1
cotterdesign.Combosocket.AddItem (Sheets(2).Range("A" & i + 1))
cotterdesign.Combocotter.AddItem (Sheets(2).Range("A" & i + 1))
Next i
End Sub
```

The output of the combobox is the list index number of selected material in UserForm, on the basis of selected material's index number we assign their properties from the sheet No.2 (Material Database) into the already defined variables in main program. Here require to add 2 with index number because first row is occupied by column's name and also the index number is start from zero, but for designating the cell address row and column numbers are always start from one. Following statements are used for assign the material's required properties from sheet No.2.

```
ts_c = Sheets(2).Range("B" & cotterdesign.Combocotter.ListIndex + 2).Value
ts_s = Sheets(2).Range("B" & cotterdesign.Combosocket.ListIndex + 2).Value
cs_c = Sheets(2).Range("C" & cotterdesign.Combocotter.ListIndex + 2).Value
cs_s = Sheets(2).Range("C" & cotterdesign.Combosocket.ListIndex + 2).Value
ss_c = Sheets(2).Range("D" & cotterdesign.Combocotter.ListIndex + 2).Value
ss_s = Sheets(2).Range("D" & cotterdesign.Combosocket.ListIndex + 2).Value
```

In UserForm behind the "COMPUTE" button the subroutine named "compute" is called for design the cotter joint.

```
Private Sub CommandButton1_Click() 'COMPUTE button
Call compute
End Sub
```

Code behind the "RESET" button simply includes the codes of closing and showing the form again.

```
Private Sub CommandButton2_Click() 'RESET button
Unload cotterdesign
cotterdesign.Show
EndSub
```

2.4 Output Data in Excel

After execution of the program that updates the final output values in already defined cells of sheet 1. And this data is sufficient to fully define the 3D model of cotter joint.

2.5 Master Assembly

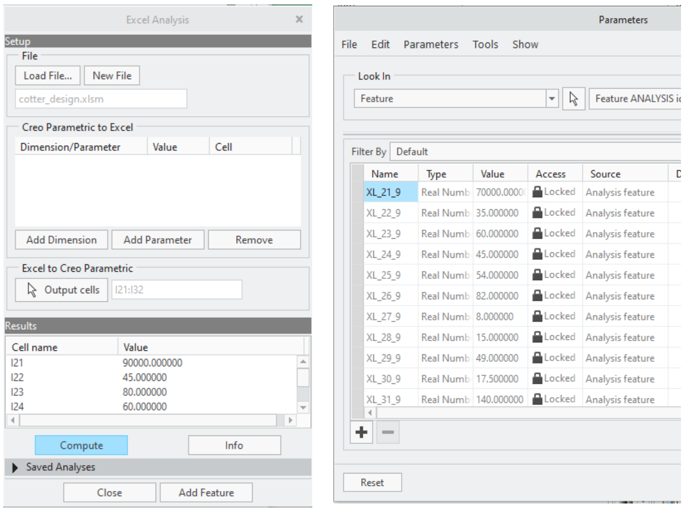
After creating the design module in excel then second step is to create a master assembly in CAD software in a top-down manner of modeling. In this modeling approach we start with top level assembly and then create parts or subassemblies instead of insert. It reduces the complexity for modification of the complete assembly because in this modeling approach various part dimensions can be modified on one assembly file. It also makes ease in adding relations between model's dimensions and excel output values.

2.5.1 Data linking from excel to cad software

The output cells of excel is linked directly to the modeling environment (PTC CREO). And this will act as locked parameters in CAD software. Hence those values are only driven by the master excel file. In order to fetch the final values from excel, "Excel Analysis" feature need to be created, which can be accessed under the "Analysis" tab of the PTC CREO (Creo Parametric Help Center, 2021). This feature is created by activating the top level assembly.

The output excel file is loaded by clicking on "Load File" button and then select the output cells. On successful run of analysis feature the values from the selected cells appear in the same dialogue box under the result section after clicking on "Compute" button as shown in figure 5(a). This result section shows an error while loading the excel file having merge cells in any sheet. Hence for linking excel data to CAD there should not be use of merge cell option in entire workbook.

After adding the analysis feature the number of local parameters are automatically created under this feature having default name XL_row number_column number. The numerical values of these parameters are not accessed in CAD environment. It act as locked parameters as shown in figure 5(b).



cotter_design.xlsm

Dimension/Parameter	Value	Cell

Output cells: I21:I32

Cell name	Value
I21	90000.000000
I22	45.000000
I23	80.000000
I24	60.000000

Look In: Feature — Feature ANALYSIS i

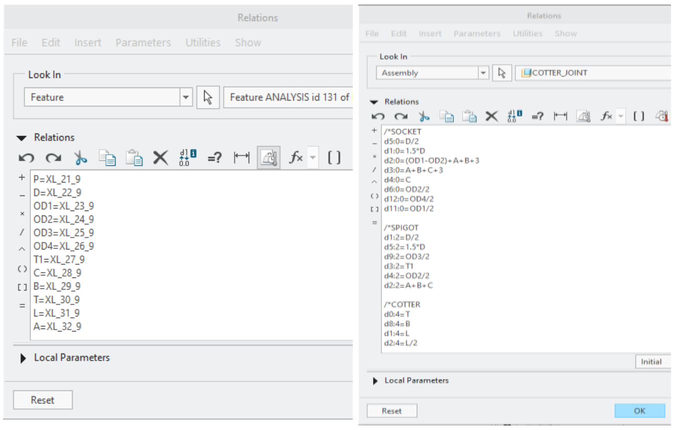
Filter By: Default

Name	Type	Value	Access	Source
XL_21_9	Real Numb	70000.0000	Locked	Analysis feature
XL_22_9	Real Numb	35.000000	Locked	Analysis feature
XL_23_9	Real Numb	60.000000	Locked	Analysis feature
XL_24_9	Real Numb	45.000000	Locked	Analysis feature
XL_25_9	Real Numb	54.000000	Locked	Analysis feature
XL_26_9	Real Numb	82.000000	Locked	Analysis feature
XL_27_9	Real Numb	8.000000	Locked	Analysis feature
XL_28_9	Real Numb	15.000000	Locked	Analysis feature
XL_29_9	Real Numb	49.000000	Locked	Analysis feature
XL_30_9	Real Numb	17.500000	Locked	Analysis feature
XL_31_9	Real Numb	140.000000	Locked	Analysis feature

Fig -5: Excel Analysis Feature: (a) Feature Dialogue Box, (b) Default Feature Parameters.

In top level assembly we create local parameters to assign the numerical values of excel to their respective dimension's ID. The default created parameters can't equate directly to the dimension's ID because in relation dialogue box of top level assembly we can use only that parameters which are created under top level assembly only. Therefore first we need to create local parameters under top level assembly and then assign these numerical values of excel into them by writing relations in relation dialogue box under the created analysis feature as shown in figure 6(a). The created local parameters act as an intermediate link to assign dimension data from excel to the dimensions ID.

After set up relations between default created parameters in analysis feature and local created parameters in top level assembly, we can stablish relations between the dimension's ID and the local created parameters in top level assembly relation dialogue box as shown in figure 6(b).



Look In: Feature — Feature ANALYSIS id 131 of

```
P=XL_21_9
D=XL_22_9
OD1=XL_23_9
OD2=XL_24_9
OD3=XL_25_9
OD4=XL_26_9
T1=XL_27_9
C=XL_28_9
B=XL_29_9
T=XL_30_9
L=XL_31_9
A=XL_32_9
```

Look In: Assembly — COTTER_JOINT

```
/*SOCKET
d5:0=D/2
d1:0=1.5*D
d2:0=(OD1-OD2)+A+B+3
d3:0=A+B+C+3
d4:0=C
d6:0=OD2/2
d12:0=OD4/2
d11:0=OD1/2

/*SPIGOT
d1:2=D/2
d5:2=1.5*D
d9:2=OD3/2
d3:2=T1
d4:2=OD2/2
d2:2=A+B+C

/*COTTER
d0:4=T
d8:4=B
d1:4=L
d2:4=L/2
```

Fig -6: Relations: (a) Relations in Analysis Feature, (b) Relations in Assembly

2.6 Regeneration of Model

Regeneration is the final step in updating of model. Once the excel compute all the cotter join dimension according to the user input, then after save this excel workbook regeneration of analysis feature and model is required. During the regeneration of analysis feature the PTC CREO extract the data from saved excel. And during regeneration of model it update the model as per the value extracted from excel.

3. RESULT AND DISCUSSION

Once the design module is created, we can save design time in calculation and modification or updating of 3D model, especially in production where large number of variations is required, and also during live projects, where things are more dynamic and changing continuously till the end of project. We can also define the same module itself in Parametric CAD software (PTC CREO) but it takes more time to process the Data and regeneration of model. Thus we split the process into two part, design algorithm of cotter joint is separate run on excel VBA and the model is updated in CAD by extracting these processed values from excel in separate run as shown in figure 7. So this can be run on an ordinary computer.

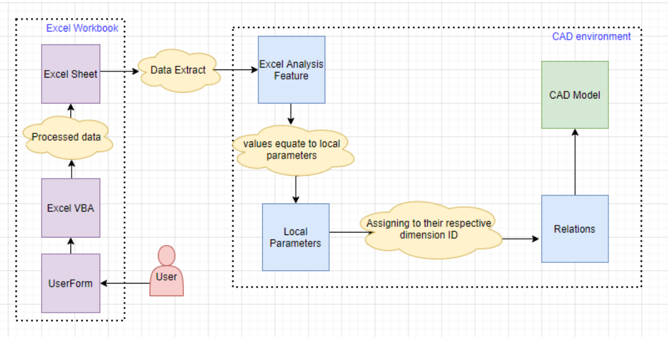


Fig -7: Workflow Chart

4. CONCLUSIONS

Thus at the end we can conclude that with the application of VBA module in Excel and UserForm incorporated with calculation sheet (where the user can enter the input values and also can select the recommended materials in dropdown button of UserForm). We can get an automated approach for model updating and also synchronizing of CAD model with calculation sheet without any need of machine learning and computer of high specifications.

REFERENCES

[1] "Creo Parametric Help Center", Support.ptc.com, 2021. [Online]. Available: http://support.ptc.com/help/creo/creo_pma/usascii/index.html#page/model_analysis/bemod/To_Create_an_Excel_Analysis.html. [Accessed: 05- Nov- 2020].

[2] "Excel Macros - UserForms - Tutorialspoint", Tutorialspoint.com, 2021. [Online]. Available: https://www.tutorialspoint.com/excel_macros/excel_macros_userforms.htm. [Accessed: 04- Jan- 2021].

[3] F. Fu, "Design and analysis of tall and complex structures", Oxford: Butterworth-Heinemann, 2018, pp. 197-199.

[4] G. Romeo, "Excel VBA, solver, and other advanced worksheet tools", Elements of Numerical Mathematical Economics with Excel, 24-Jan-2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128176481000013. [Accessed: 05-Nov-2020].

[5] "Microsoft Excel – A Handy Tool Used in Various Areas of Business - Home", Home, 2016. [Online]. Available: https://inurture.co.in/blogs/microsoft-excel-a-handy-tool-used-in-various-areas-of-business/. [Accessed: 04- Jan- 2021].

[6] Narayan, K. Lalit (2008). "Computer Aided Design andManufacturing", New Delhi: Prentice Hall of India. p. 3. ISBN 978-8120333420.

[7] PTC creo, "Parametric Design with Creo Elements/Direct Drafting Windows User Interface Creo Elements/Direct Drafting 20.1", 2020. [Online]. Available: https://support.ptc.com/help/creo/ced_drafting/r20.1.1.0/en/ced_drafting/baggage/param_win.pdf. [Accessed: 05-Nov-2020].

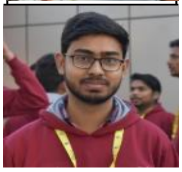
[8] "VBA visual basic for application", Tutorialspoint, 2016. [Online]. Available: www.tutorialspoint.com; https://www.tutorialspoint.com/vba/vba_tutorial.pdf. [Accessed: 05-Nov-2020].

[9] V. Bhandari, "Design of machine elements", 3rd ed. New Delhi: McGraw-Hill Education (India), 2017, pp. 85-91.

BIOGRAPHIES

Shubham Singh (B.Tech in Mechanical Engineering, Delhi Technological University)

Roopak Rawat (B.Tech in Mechanical Engineering, Delhi Technological University)



Kuldeep (B.Tech in Mechanical Engineering, Delhi Technological University)