Journal of Mines, Metals and Fuels, 72(10): 1075-1091; 2024. DOI: 10.18311/jmmf/2024/45765

Print ISSN : 0022-2755

Journal of Mines, Metals and Fuels

Contents available at: www.informaticsjournals.com/index.php/jmmf

A Python-Based GUI Approach for Efficient Component Design of Compound Die for Composite Material: 3D and 2D CAD Modeling

Vijaysing Rathod[1]*, Atul Karanjkar[2], and Nandkishor Sawai[3]

[1]*Chhatrapati Shivaji Maharaj University, Panvel, Navi Mumbai - 410221, Maharashtra, India; vtrathod8291@gmail.com*

[2]*Department of Mechanical Engineering, Chhatrapati Shivaji Maharaj University, Panvel, Navi Mumbai - 410221, India*

[3]*Sandip Institute of Technology and Research Centre, Nashik - 422213, Maharashtra, India*

Abstract

This article presents a Python-based parametric Computer-Aided Design (CAD) system coupled with the AutoCAD API for the automation of compound die design. The system seeks to minimize the duration allocated to repeated operations, therefore promoting innovation. The graphical user interface facilitates the straightforward entry of design parameters and the automated production of precise manufacturing drawings. Advanced approaches, including Finite Element Analysis (FEA), topology optimization, and machine learning, are utilized to improve product design, reduce material consumption, and forecast component lifespan under operational stressors. The technology markedly decreases design time relative to conventional methods, rendering it optimal for sectors emphasizing bulk customization and batch production. Moreover, it enhances Time-To-Market (TTM) by providing an efficient process from design to manufacturing, while reducing errors and material waste. This tool has the potential for extensive utilization in mechanical design, especially within small and medium-sized enterprises.

Keywords: *API Based CAD, GUI Based CAD Modeling, Knowledge Based System, Python Programming*

1.0 Introduction

Compound dies are crucial in the sheet metal sector for producing perforated blanks; yet their design process is laborious and necessitates considerable knowledge. Traditional approaches to selecting die components are intricate, labour-intensive, and susceptible to inaccuracies, primarily dependent on the expertise of seasoned designers. Designers are required to consult manuals, apply formulas, and conduct several computations, so complicating the process, particularly in light of the industry's scarcity of proficient professionals.

The fabrication of progressive dies is notably complex, requiring significant expertise and manual labour. Notwithstanding the support of CAD/CAM technology in drafting and analysis, the completion of designs remains contingent upon human involvement. The elevated expense of these systems renders them unattainable for several Small and Medium-Sized Organizations (SMEs), especially in developing areas. An immediate necessity exists for an automated system that streamlines the design process, diminishes dependence on expert personnel, and aids SMEs by optimizing the selection and design of die components. An ideal system would provide intelligent

*Author for correspondence

support, automating essential processes in progressive die design.

Kumar and Singh[1] created a Knowledge-Based System (KBS) to improve the die design process in the stamping sector. This system employs Artificial Intelligence (AI) and manufacturing algorithms to automate the selection of progressive die components. Kashid and Kumar[2] created an advanced system for selecting compound die components, consisting of eight specialized modules. These modules employ domain expertise and "IF-Then" production rules to automate the selection of components. Kumar and Singh[3] presented an economical framework for a specialized knowledge-based system designed to enhance progressive die design. The system has modules, each tailored to improve activities associated with die design, emphasizing user-friendliness. Hambli *et al.*[4] devised a technique for ascertaining the ideal distance between the punch and die by employing the Lemaitre damage model to analyze fracture initiation and propagation in sheet metal. Their methodology integrated predictive finite element modeling with neural network modeling to evaluate essential blanking parameters. Nee and Foong[5] emphasized the intricate and laborious process of progressive die development, which conventionally depends on proficient designers. Bin Ab Kadir *et al.*[6] concentrated on the creation of a micro blanking dieset designed for industrial applications. They utilized solid works to construct the dieset and applied Finite Element Analysis (FEM) to evaluate its performance in the blanking process. Kumar and Singh[7] created an automated system for progressive die design, comprising 27 knowledge-based modules utilizing production rule-based methodologies. These modules are intuitive and integrate effortlessly into AutoCAD's prompt area, facilitating the automation of essential activities such as analyzing sheet metal component designs, generating strip layouts, selecting die components, and executing modeling, assembly, and material selection. Shaheen *et al.*[8] performed an examination of compound dies utilizing ANSYS software to ascertain the principal parameters affecting effective cutting tool design. The research concentrated on identifying the optimal piercing punch based on its efficacy in reducing burr height in the finished product. The findings indicated that larger sheet metals diminished burr heights, whereas thinner sheets augmented them. Kadarno *et al.*[9] devised an innovative punching technique to enhance the fatigue durability of ultra-high strength steel sheets. This method utilized a tapered punch and a graduated die to expand the aperture and augment the edge thickness of the hole. The punch angle and die step height were tuned to attain the specified thickness. Kumar[10] performed a study to enhance die design parameters in the blanking process by the application of a genetic algorithm. The study sought to augment tool durability, elevate production rates, and enhance overall results. The results verified that attaining optimal parameter levels markedly enhances both production efficiency and product quality. Skampardonis *et al.*[11] devised an advanced progressive die featuring two stations for the fabrication of a complicated metallic component through the integration of three separate manufacturing techniques. The design of the die's components was predicated on a synthesis of mathematical computations and empirical evidence. Shaheen *et al.*[12] refined compound die types, piercing punches, and double cutting settings through a comprehensive methodology that integrated finite element technology, the Taguchi technique, regression analysis, and analysis of variance. The study evaluated various punch cutting edges—flat, chamfered, flat edge with a concave hemisphere, and convex-shaped—to fill knowledge gaps regarding compound dies in stamping operations. Mehta and Rus[13] investigated a development environment for creating mechanical structures of folded plastic robots through scripting, drawing inspiration from origami due to its cost-effectiveness and accessibility. Their instruments sought to facilitate the dissemination of open-source designs and improve the reutilization of modular components. Machado *et al.*[14] underscored the significance of open-source hardware for scientific apparatus, stressing the necessity of incorporating fundamental elements such as source files and comprehensive documentation. This methodology enables people to analyze, reproduce, and alter designs. Tezzele *et al.*[15] examined the PyGeM toolbox, a python-based application intended for the parameterization and deformation of 3D objects derived from CAD files or 3D meshes. The toolbox comprises several morphing techniques, including free-form deformation, radial basis function interpolation, and inverse distance weighting. Salunkhe and Kumar[16] emphasize the transformative influence of AI/ML techniques on the design process, especially concerning sheet metal components and press equipment. They contend that contemporary designers require sophisticated abilities and substantial industrial expertise. Zuo and Xie[17] presented a python-based methodology comprising 100 lines of code to enhance 3D topology optimization. This code employs the Abaqus scripting interface, facilitating access to intricate Finite Element Analysis (FEA). Their methodology utilizes Bi-directional Evolutionary Structural Optimization (BESO) technology, emphasizing compliance reduction while maintaining volume limitations. Agrawal *et al.*,[18] found that component optimization is typically done without addressing how the optimized components will fit together in an assembly. Thus, fitting and compatibility concerns may develop when these optimized components are assembled. Their study used CAD-based optimization to identify the optimized component's packing space boundaries. Mukundakrishnan *et al.*[19] suggested a methodology for generating 3-D geometric models via programming approaches, specifically aimed at simulation-based design. Their research highlights the use of the Python API for executing mathematical operations unsupported by the kernel and for interfacing with other Python-based studies. Mathur *et al.*,[20] examined mechanical process working parameter graphical representation progression. They addressed parametric Computer-Aided Design (CAD), which defines a family of objects with a unique final shape for each parameter value. GUI-based CAD tools are popular and user-friendly, however the authors noted that GUI procedures lack a semantic definition, leaving them prone to parameter value changes.

The authors effectively combined the interactive elements of a Graphical User Interface (GUI) with the resilience of programming in their study. Their methodology, which prioritizes program synthesis via examples, produces code that precisely mirrors the user's selections made through the GUI. The trials undertaken proved the method's efficacy in swiftly synthesizing functioning and resilient sub-programs. User studies demonstrated that the GUI-based interface substantially surpassed programming-only interfaces in performance. The viability and efficacy of their parametric CAD system were evaluated through its application in the design of compound punching and blanking dies. Although previous research has concentrated on computer-assisted methods for selecting die components, such as single-operation, progressive, and deep drawing dies, there is a lack of literature specifically addressing the application of a graphical user interface for this procedure. This study aims to address that gap by creating a GUI-based CAD modeling method for complex dies.

2.0 Architecture of the Parametric Modelling system

2.1 Overview

The proposed parametric modelling system is depicted in Figure 1. The API for AutoCAD has been loaded with the algorithm that has been scripted in Python. The Tkinter GUI module has been integrated into the application so that it may offer the user a simplistic interface that is easy to navigate. For the purpose of showing the output manufacturing drawing of the compound die components, the CAD modelling software package has been incorporated into the programme through the utilisation of the Python-AutoCAD Application Program Interface (API). The user is presented with a Graphical User Interface (GUI) to input the values of design parameters for die design. Inputs such as the sheet material to be punched, the shape of the die block and punch hole, and dimensional inputs such as the diameter of the punch hole and spring as well as assembly allowances are all taken into consideration. In addition to this, mechanical properties are also taken into account throughout the design calculation process. These inputs are saved as separate values within the programme as separate variables. The design is carried out using the inputs that have been provided, and the dimensions of the features and components of the compound die are subsequently obtained. The user will be asked to provide additional inputs for the drawing of the die in a manner that is appropriate for the shape of the die and punch that they have chosen. After ensuring that all of the requirements have been met and the process of die design has been concluded, the die plates are drawn on the AutoCAD environment. With the sketch, dimensions are added to all features of each die plate to make it suitable for production.

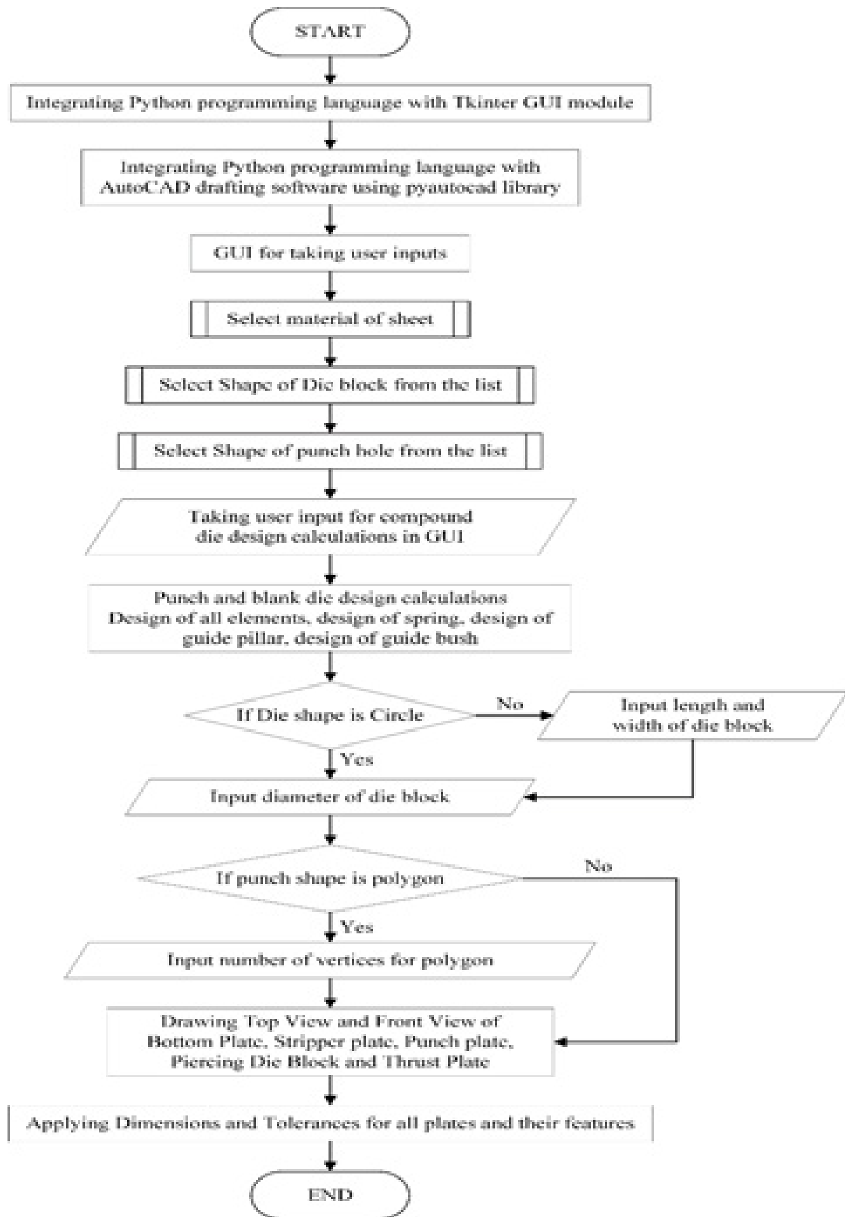

START
Integrating Python programming language with Tkinter GUI module
Integrating Python programming language with AutoCAD drafting software using pyautocad library
GUI for taking user inputs
Select material of sheet
Select Shape of Die block from the list
Select Shape of punch hole from the list
Taking user input for compound die design calculations in GUI
Punch and blank die design calculations Design of all elements, design of spring, design of guide pillar, design of guide bush
If Die shape is Circle
No
Input length and width of die block
Yes
Input diameter of die block
If punch shape is polygon
No
Yes
Input number of vertices for polygon
Drawing Top View and Front View of Bottom Plate, Stripper plate, Punch plate, Piercing Die Block and Thrust Plate
Applying Dimensions and Tolerances for all plates and their features
END

Figure 1. Sequential steps in script development and CAD modeling for compound die design.

3.0 Development of a System for the Parametric Design of Components

3.1 Overview of Pre-Processing Software

3.1.1 Autocad and General CAD Software

AutoCAD is an automatic computer aided drawing tool used for both two- and three-dimensional modeling. An outstanding characteristic of this software is its integration of NURBS (Non-Uniform Rational B-Splines) modeling capabilities, which enhances its usefulness. AutoCAD allows for the creation of NURBS curves and surfaces, with the ability to save models in different formats. The DXF file format enables the swift retrieval of data related to NURBS curves. Nevertheless, managing intricate NURBS surface data can present more difficulties. AutoCAD offers two main methods for curve modeling: one involves fitting the curve to predetermined data points, while the other comprises defining the curve's shape using control points. To replicate curves in AutoCAD, one can utilize control points, manipulate the degree (with a maximum limit of 10), and edit the position or value of these control points.

3.2 Software Abstraction

Python was chosen as the programming language because it is particularly good at handling mathematical calculations and analysing challenging scientific and technical issues. It features a noticeably clear, simple, and succinct syntax together with a lot of expressive power. The application imports the Pyautocad module and utilizes the Python-AutoCAD API for generating compound die drafting. The program is made up of three blocks: user interface, design computation and component sketching in the CAD software. To streamline the design calculations, users are required to submit essential specifications, like the material employed for the die plates and the precise configuration of the punch. These calculations yield precise dimensional values that are crucial for creating detailed representations of the components. Users with rudimentary proficiency can effortlessly input data, such as selecting materials and setting precise external measurements of components, without requiring comprehension of the underlying code. This approach ensures flexibility and inclusiveness in the design process for a diverse range of consumers. AutoCAD is known for its ability to support industry-standard parametric CAD formats and its robust export capabilities. Furthermore, the software supports both Python script-based modeling and GUI modeling.

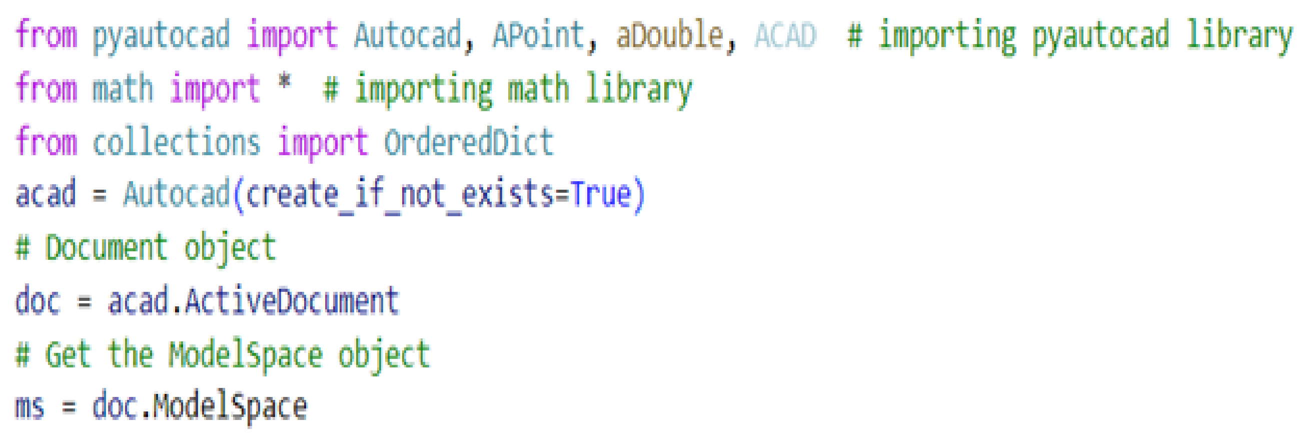
3.3 System Implementation

3.3.1 Importing Libraries (Modules)

Executable instructions and function definitions are both included in modules. The initial state of the module is set using these statements. Modules possess the capacity to incorporate other modules, enabling the exchange of functions and data among various sections of a program. Although it is advisable, it is not obligatory to include all import lines at the beginning of a module. In order to simplify operations related to coordinates, one can import the Pyautocad package, which enables the use of AutoCAD as the main Automation object in combination with aPoint and aDouble. The graphic shows how Python is connected to AutoCAD via AutoCAD. To do coordinate operations, you can also use predefined functions like APoint and aDouble. A common python module used for mathematical operations is called math. User must import the module using import math under this module for utilizing various mathematical functions available in the library.

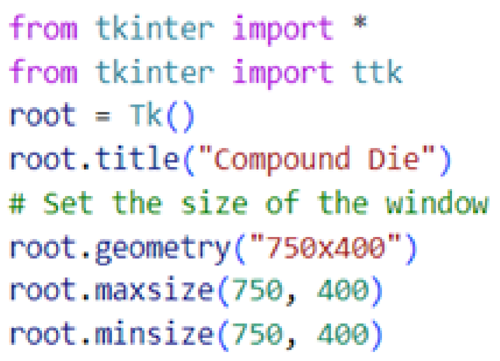
3.4 Interface Model

The most important goal of this paper is to outline the script of the GUI. The Tkinter Module is used to design the GUI panel. This Tkinter Module is imported into the python interface. Figure 3 shows the script for importing



```
from pyautocad import Autocad, APoint, aDouble, ACAD  # importing pyautocad library
from math import *  # importing math library
from collections import OrderedDict
acad = Autocad(create_if_not_exists=True)
# Document object
doc = acad.ActiveDocument
# Get the ModelSpace object
ms = doc.ModelSpace
```

Figure 2. Importing the Pyautocad library.

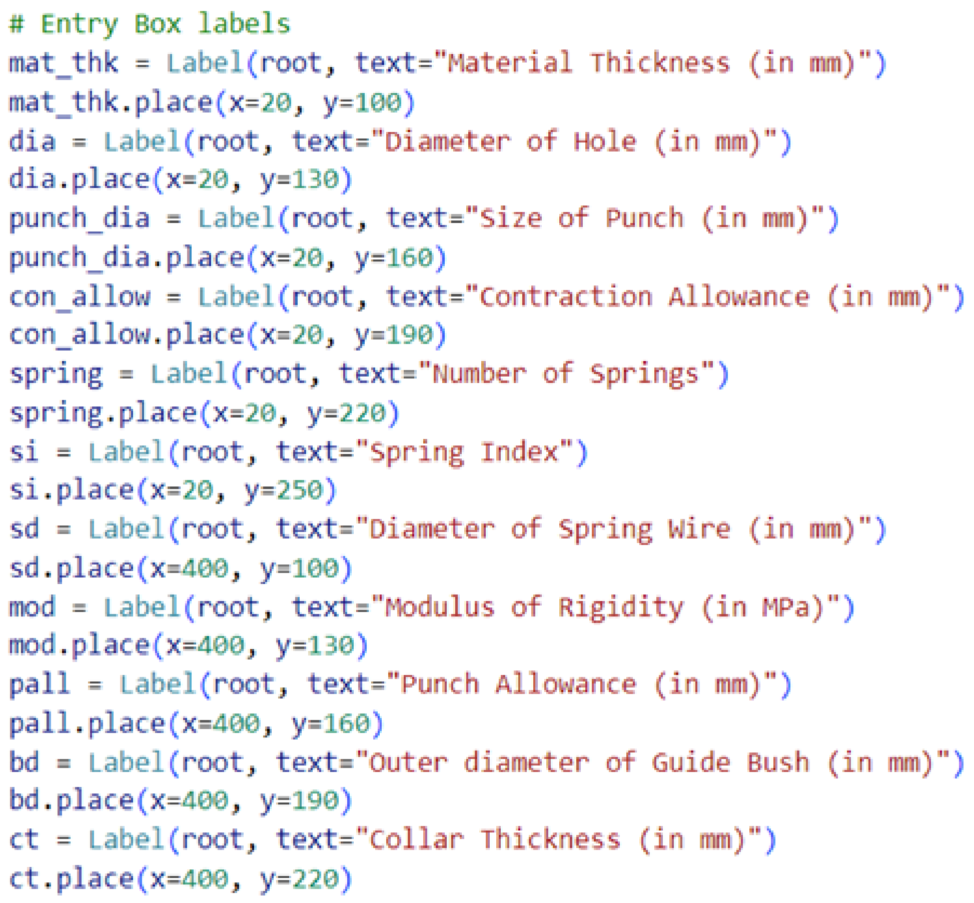


```
from tkinter import *
from tkinter import ttk
root = Tk()
root.title("Compound Die")
# Set the size of the window
root.geometry("750x400")
root.maxsize(750, 400)
root.minsize(750, 400)
```

Figure 3. Importing the Tkinter module library.

The next step is to define the values in the boxes that were created in the previous step. This is done by inserting entry command to ask the user for the input. The user inputs the values, and these values are then fed to the underlying variables in the main code. The script for asking the users for an input in the GUI window is shown in Figure 5.

The GUI module automatically assigns the values to the design variables in the main code and the code is executed as shown in [Previous paper]. the design algorithm then creates the design of the bottom, stripper,



```
# Entry Box labels
mat_thk = Label(root, text="Material Thickness (in mm)")
mat_thk.place(x=20, y=100)
dia = Label(root, text="Diameter of Hole (in mm)")
dia.place(x=20, y=130)
punch_dia = Label(root, text="Size of Punch (in mm)")
punch_dia.place(x=20, y=160)
con_allow = Label(root, text="Contraction Allowance (in mm)")
con_allow.place(x=20, y=190)
spring = Label(root, text="Number of Springs")
spring.place(x=20, y=220)
si = Label(root, text="Spring Index")
si.place(x=20, y=250)
sd = Label(root, text="Diameter of Spring Wire (in mm)")
sd.place(x=400, y=100)
mod = Label(root, text="Modulus of Rigidity (in MPa)")
mod.place(x=400, y=130)
pall = Label(root, text="Punch Allowance (in mm)")
pall.place(x=400, y=160)
bd = Label(root, text="Outer diameter of Guide Bush (in mm)")
bd.place(x=400, y=190)
ct = Label(root, text="Collar Thickness (in mm)")
ct.place(x=400, y=220)
```

Figure 4. Script for designing the box labels.

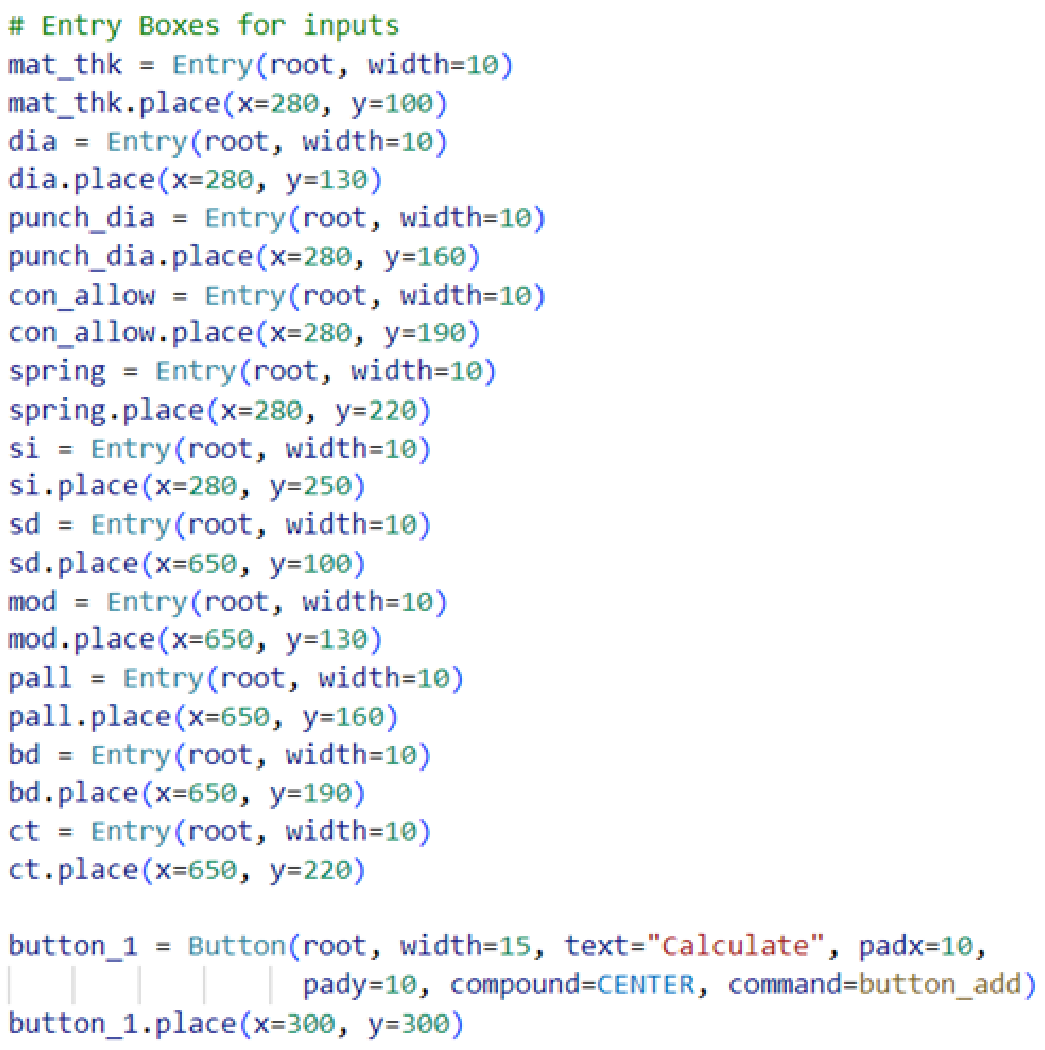
the Tkinter Module. The dimensions of the GUI widow are defined after importing this library.

The next step is to define the box labels containing the design parameters of the flange coupling. The Figure 4 shows the script to design the box labels. Here, individual boxes are defined using x and y coordinates.

punch and thrust plates as well as the piercing die block as shown in Figure 5.

3.4.1 Selecting the Sheet Material

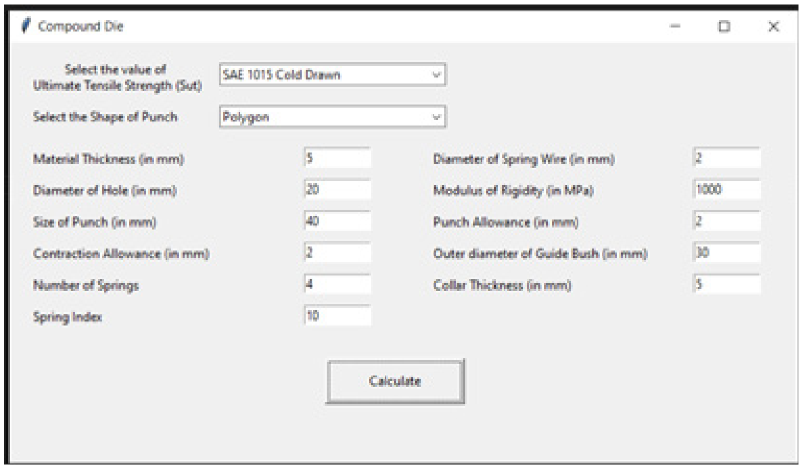
Figure 7 shows the GUI with list of materials for selecting the required material of the sheet. The Ultimate tensile



```
# Entry Boxes for inputs
mat_thk = Entry(root, width=10)
mat_thk.place(x=280, y=100)
dia = Entry(root, width=10)
dia.place(x=280, y=130)
punch_dia = Entry(root, width=10)
punch_dia.place(x=280, y=160)
con_allow = Entry(root, width=10)
con_allow.place(x=280, y=190)
spring = Entry(root, width=10)
spring.place(x=280, y=220)
si = Entry(root, width=10)
si.place(x=280, y=250)
sd = Entry(root, width=10)
sd.place(x=650, y=100)
mod = Entry(root, width=10)
mod.place(x=650, y=130)
pall = Entry(root, width=10)
pall.place(x=650, y=160)
bd = Entry(root, width=10)
bd.place(x=650, y=190)
ct = Entry(root, width=10)
ct.place(x=650, y=220)

button_1 = Button(root, width=15, text="Calculate", padx=10,
                  pady=10, compound=CENTER, command=button_add)
button_1.place(x=300, y=300)
```

Figure 5. Script for asking the user for inputs.



Compound Die

Select the value of Ultimate Tensile Strength (Sut): SAE 1015 Cold Drawn

Select the Shape of Punch: Polygon

Material Thickness (in mm)	5	Diameter of Spring Wire (in mm)	2
Diameter of Hole (in mm)	20	Modulus of Rigidity (in MPa)	1000
Size of Punch (in mm)	40	Punch Allowance (in mm)	2
Contraction Allowance (in mm)	2	Outer diameter of Guide Bush (in mm)	30
Number of Springs	4	Collar Thickness (in mm)	5
Spring Index	10		

Figure 6. Layout of GUI.

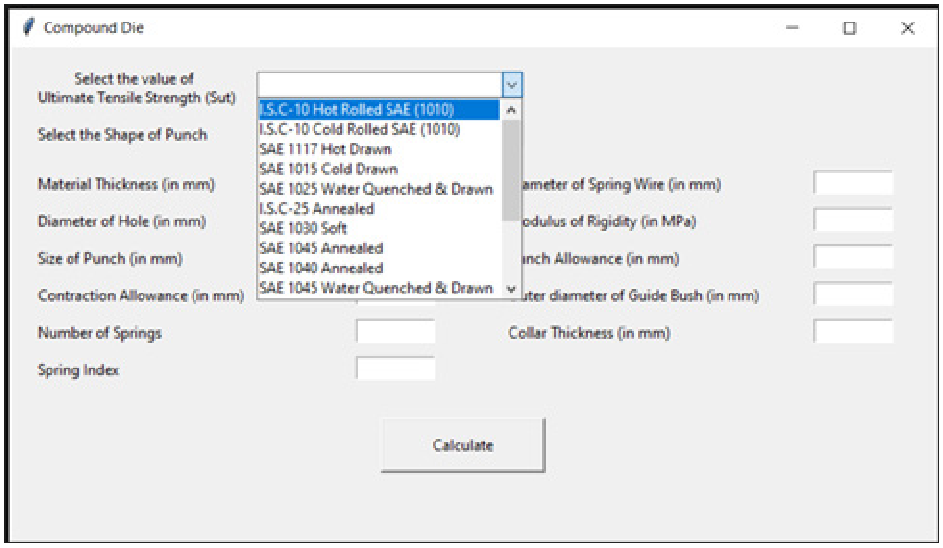

Compound Die
Select the value of Ultimate Tensile Strength (Sut)
I.S.C-10 Hot Rolled SAE (1010)
I.S.C-10 Cold Rolled SAE (1010)
SAE 1117 Hot Drawn
SAE 1015 Cold Drawn
SAE 1025 Water Quenched & Drawn
I.S.C-25 Annealed
SAE 1030 Soft
SAE 1045 Annealed
SAE 1040 Annealed
SAE 1045 Water Quenched & Drawn
Select the Shape of Punch
Material Thickness (in mm)
Diameter of Hole (in mm)
Size of Punch (in mm)
Contraction Allowance (in mm)
Number of Springs
Spring Index
ameter of Spring Wire (in mm)
odulus of Rigidity (in MPa)
nch Allowance (in mm)
uter diameter of Guide Bush (in mm)
Collar Thickness (in mm)
Calculate

Figure 7. Material selection in GUI.

strength is an important parameter while designing the die. The cutting force required for the punching operation is directly proportional to the ultimate tensile strength. Thus, increase in the ultimate tensile strength of the sheet material will increase the cutting force required for the operation. The most commonly sheet materials used in the industry are included in the list to allow the user to select the exact material to be actually used for the operation and get accurately designed die for that material.

3.4.2 Select the Shape of Punch Hole

Figure 8 shows the list of punch shapes for selecting the required punch shape. Based on the selection of the shape, the user is asked for inputs required for the selected shape.

3.5 Methodology for Die Design

3.5.1 Calculation of Force and Cutting Clearance

The cutting is calculated based on shear strength, thickness and the punch diameter as given below,

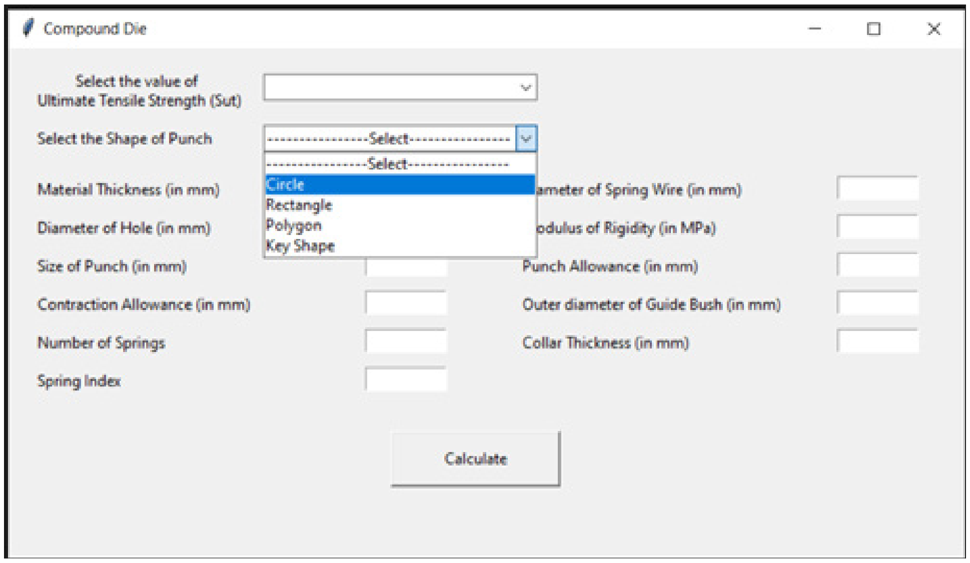
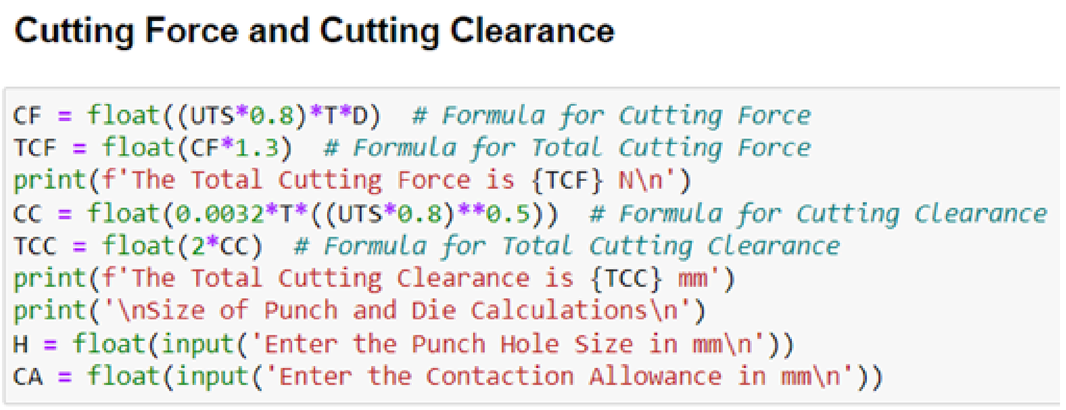

Compound Die
Select the value of Ultimate Tensile Strength (Sut)
Select the Shape of Punch
----------------Select----------------
----------------Select----------------
Circle
Rectangle
Polygon
Key Shape
Material Thickness (in mm)
Diameter of Hole (in mm)
Size of Punch (in mm)
Contraction Allowance (in mm)
Number of Springs
Spring Index
ameter of Spring Wire (in mm)
odulus of Rigidity (in MPa)
Punch Allowance (in mm)
Outer diameter of Guide Bush (in mm)
Collar Thickness (in mm)
Calculate

Figure 8. Selection of punch shape in GUI.


Cutting Force and Cutting Clearance

```
CF = float((UTS*0.8)*T*D)  # Formula for Cutting Force
TCF = float(CF*1.3)  # Formula for Total Cutting Force
print(f'The Total Cutting Force is {TCF} N\n')
CC = float(0.0032*T*((UTS*0.8)**0.5))  # Formula for Cutting Clearance
TCC = float(2*CC)  # Formula for Total Cutting Clearance
print(f'The Total Cutting Clearance is {TCC} mm')
print('\nSize of Punch and Die Calculations\n')
H = float(input('Enter the Punch Hole Size in mm\n'))
CA = float(input('Enter the Contaction Allowance in mm\n'))
```

Figure 9. Code for calculating the cutting clearance.

$F_c = \tau \times t_s \times d_p$

Where, F_c is cutting force required in N; τ is shear strength in MPa; t_s is sheet thickness in mm and d_p is diameter of punch

In the design technique, τ = 0.8σut of the material. Apart from this stripping element, inertia element and friction elements are utilized in the computation.

The cutting clearance can be estimated by utilizing the following correlation.

$C_c = 0.032 \times t_s$

$C_{tc} = 2 \times C_c$

where, C_c is cutting clearance and C_{tc} is total cutting clearance.

3.5.2 Size of Punch and Die

To pierce the hole:

$P_s = H_s + CA$

Where, P_s is punch size in mm; H_s is hole size in mm and CA is contraction allowance in mm.

$D_s = P_s + C_{tc}$

Where, D_s dies block size

To Blank the hole

$P_s = H_s - CA$

and $D_s = P_s - C_{tc}$

The total cutting force required is given by,

$F_{TC} = 1.3 \times F_C$

Where, F_{TC} is total cutting force in N

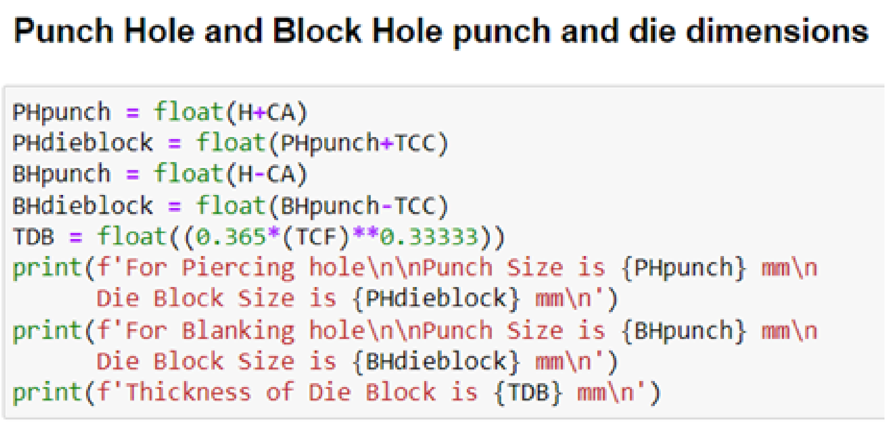
The thickness of die block is given by,

$D_t = 0.365 \times \sqrt[3]{(F_{TC})}$

Where, D_t is thickness of die block

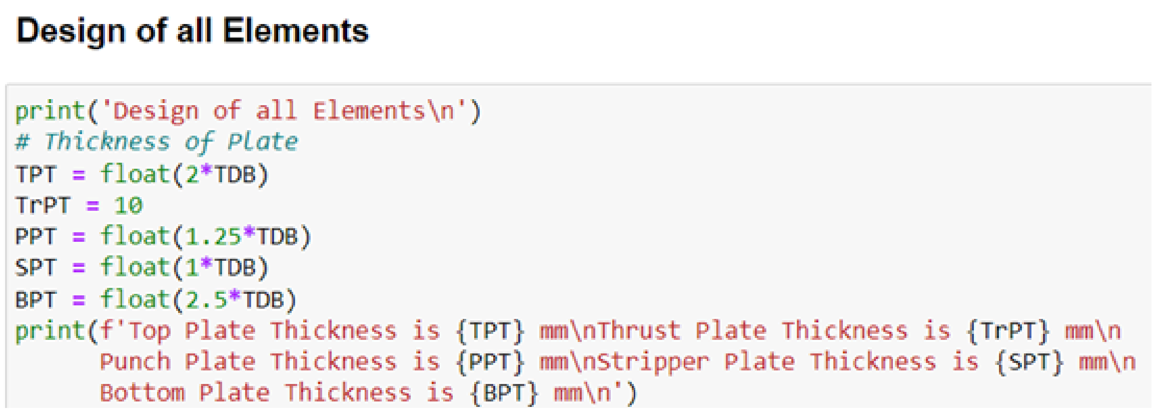
3.5.3 Design of All Elements

The size of all the compound die elements can be determined by considering the thickness of the die block.


Punch Hole and Block Hole punch and die dimensions

```
PHpunch = float(H+CA)
PHdieblock = float(PHpunch+TCC)
BHpunch = float(H-CA)
BHdieblock = float(BHpunch-TCC)
TDB = float((0.365*(TCF)**0.33333))
print(f'For Piercing hole\n\nPunch Size is {PHpunch} mm\n
      Die Block Size is {PHdieblock} mm\n')
print(f'For Blanking hole\n\nPunch Size is {BHpunch} mm\n
      Die Block Size is {BHdieblock} mm\n')
print(f'Thickness of Die Block is {TDB} mm\n')
```

Figure 10. Code for evaluation of dimensions of punch and die.



Design of all Elements

```
print('Design of all Elements\n')
# Thickness of Plate
TPT = float(2*TDB)
TrPT = 10
PPT = float(1.25*TDB)
SPT = float(1*TDB)
BPT = float(2.5*TDB)
print(f'Top Plate Thickness is {TPT} mm\nThrust Plate Thickness is {TrPT} mm\n
      Punch Plate Thickness is {PPT} mm\nStripper Plate Thickness is {SPT} mm\n
      Bottom Plate Thickness is {BPT} mm\n')
```

Figure 11. Code for calculating thickness of all plates.

The thickness of upper plate it 2 times of the die block thickness

The thickness of the thrust plate is 10 mm.

The thickness of the punch plate is 1.25 times of the die block thickness.

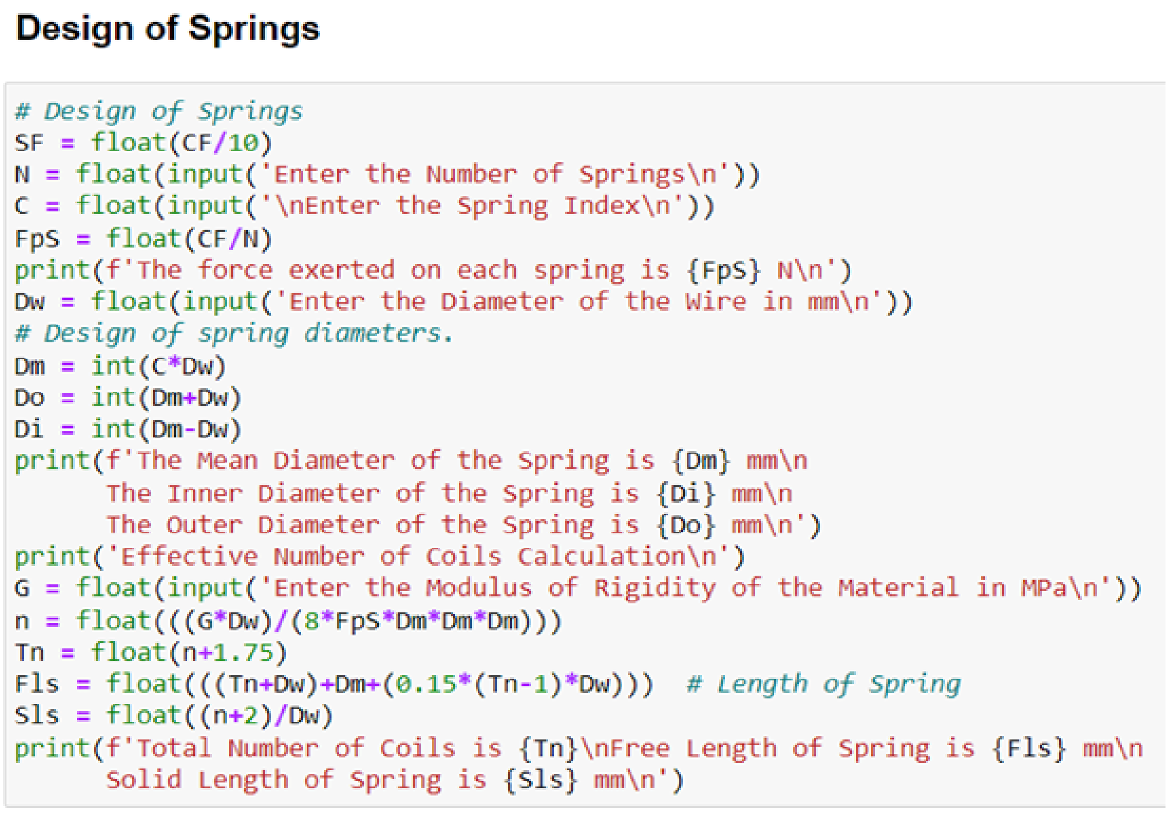
The thickness of the punch plate is 1.25 times of the die block thickness.

The thickness of the stripper plate is same as that of the die block thickness.

The thickness of the bottom plate is 2.5 times of the die block thickness.

3.5.4 Design of Springs

Based on the cutting force calculated, spring force can be calculated as



Design of Springs

```
# Design of Springs
SF = float(CF/10)
N = float(input('Enter the Number of Springs\n'))
C = float(input('\nEnter the Spring Index\n'))
FpS = float(CF/N)
print(f'The force exerted on each spring is {FpS} N\n')
Dw = float(input('Enter the Diameter of the Wire in mm\n'))
# Design of spring diameters.
Dm = int(C*Dw)
Do = int(Dm+Dw)
Di = int(Dm-Dw)
print(f'The Mean Diameter of the Spring is {Dm} mm\n
      The Inner Diameter of the Spring is {Di} mm\n
      The Outer Diameter of the Spring is {Do} mm\n')
print('Effective Number of Coils Calculation\n')
G = float(input('Enter the Modulus of Rigidity of the Material in MPa\n'))
n = float(((G*Dw)/(8*FpS*Dm*Dm*Dm)))
Tn = float(n+1.75)
Fls = float(((Tn+Dw)+Dm+(0.15*(Tn-1)*Dw)))  # Length of Spring
Sls = float((n+2)/Dw)
print(f'Total Number of Coils is {Tn}\nFree Length of Spring is {Fls} mm\n
      Solid Length of Spring is {Sls} mm\n')
```

Figure 12. Code for design calculations of springs.

Total Spring Force = Cutting Force / 10

Further, the force exerted on each spring is calculated as

Force exerted on each spring = Cutting Force / Number of Springs

The spring diameters are then calculated using the following formulae

Mean diameter of spring = Spring Index × Diameter of wire

Outer diameter of Spring = Mean diameter of spring + Diameter of wire

Inner Diameter of the Spring = Mean diameter of spring - Diameter of wire

Later the effective number of coils are calculated using the following formulae

$$\text{Effective Number of Coils} = \frac{\text{Modulus of Rigidity} \times \text{Diameter of wire}}{8 \times \text{Force Exerted on each spring} \times (\text{Mean diameter of spring})^3}$$

$$\text{Total Number of Coils} = \text{Effective number of coils} + 1.75$$

$$\text{Free Length of Spring} = (\text{Total Number of Coils} + \text{Diameter of wire}) + \text{Mean diameter of spring} + (0.15 \times (\text{Total Number of Coils} - 1) \times \text{Diameter of wire})$$

$$\text{Solid Length of Spring} = \frac{\text{Effective number of coils} + 2}{\text{Diameter of wire}}$$

3.5.5 Punch length

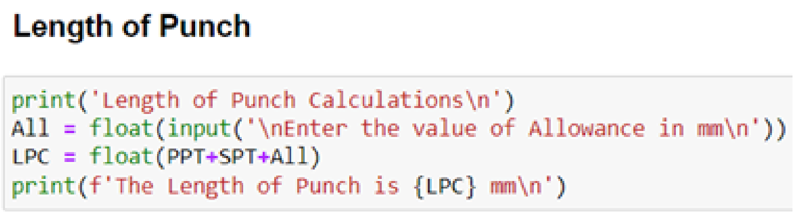
The length of the punch is evaluated with the help of thickness of plate used for punching, thickness of stripper plate, and the additional allowance.

Where, is Punch length; is punch plate thickness and is Stripper plate thickness

3.5.6 Design of Guide Pillar and Bush

The guidance pillar is defined by several particular criteria, which are outlined as follows:

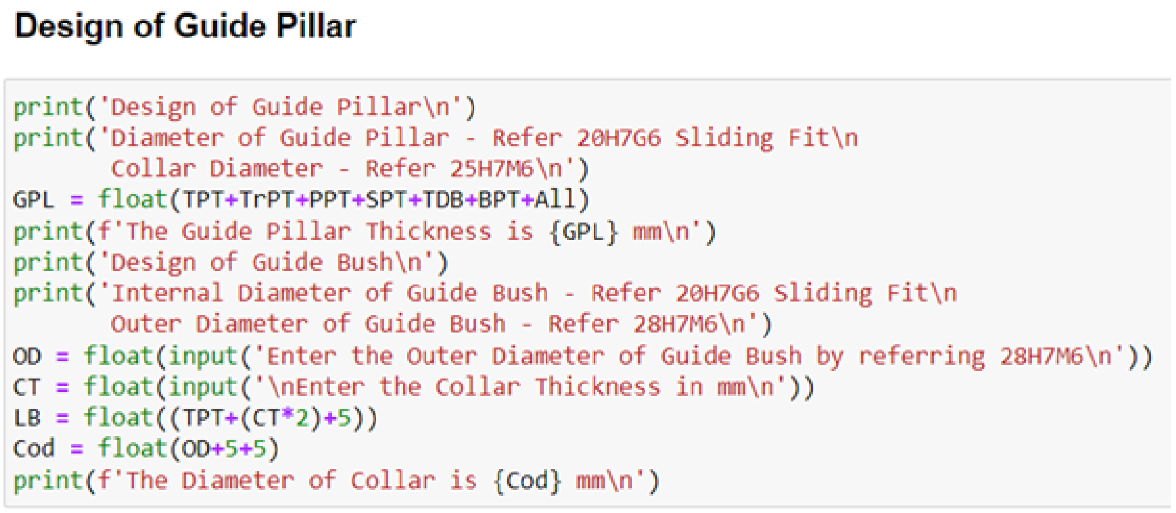
- In order to provide seamless movement between the guide pillar and the guide bush, a hole and shaft combination with a H7g6 designation is utilized.



Length of Punch

```
print('Length of Punch Calculations\n')
All = float(input('\nEnter the value of Allowance in mm\n'))
LPC = float(PPT+SPT+All)
print(f'The Length of Punch is {LPC} mm\n')
```

Figure 13. Code for punch length evaluation.



Design of Guide Pillar

```
print('Design of Guide Pillar\n')
print('Diameter of Guide Pillar - Refer 20H7G6 Sliding Fit\n
       Collar Diameter - Refer 25H7M6\n')
GPL = float(TPT+TrPT+PPT+SPT+TDB+BPT+All)
print(f'The Guide Pillar Thickness is {GPL} mm\n')
print('Design of Guide Bush\n')
print('Internal Diameter of Guide Bush - Refer 20H7G6 Sliding Fit\n
       Outer Diameter of Guide Bush - Refer 28H7M6\n')
OD = float(input('Enter the Outer Diameter of Guide Bush by referring 28H7M6\n'))
CT = float(input('\nEnter the Collar Thickness in mm\n'))
LB = float((TPT+(CT*2)+5))
Cod = float(OD+5+5)
print(f'The Diameter of Collar is {Cod} mm\n')
```

Figure 14. Code for calculating dimensions of guide pillar and guide bush.

- In order to provide a secure press, fit between the guiding pillar collar and the bottom plate, a hole and shaft pair with the designation H7m6 is used.

Bush length = Upper plate thickness + (Thickness of Collar x 2) + 5

Diameter of Collar = Guide bush outside diameter + 5 + 5

3.6 CAD System Construction and Test

3.6.1 Creating Functions for Drawing in Autocad

The Pyautocad library has limited number of pre-defined functions hence; in order to create complex drawings, the programmer will need to develop the program by utilising the fundamental and simple functions in order to produce the required drawings. This results in an increase in the total length of the program, as well as increase the computational and processing time required for the program execution. It is possible to simplify the program by eliminating the stages that are performed repeatedly by defining a new function to perform that task. The user-defined functions can be invoked at any time during the programme, whenever they are needed, and they will carry out the activity or procedure that is specified.

3.6.1.1 Creating a Circular Hole Pattern (Circular Pattern)

A "for" loop is used to draw equally spaced holes in circular pattern at a set distance from the centre of the plate. The number of holes given as the input by the user is used to determine how many "for" loops are utilised. The dimension of the holes is an input that is taken into consideration by the graphical user interface, and the coordinates of these holes are calculated based on the number of holes that are necessary.

3.6.1.2 Drawing Projection Lines (Pattern Projection Lines) of Circular Hole Pattern

In the front view, the projection lines are determined by identifying the points of the pattern holes that are diametrically opposite to one another horizontally. Due to the fact that the holes cannot be seen directly from the front view, hidden lines, also known as dashed lines, are utilised so that the features can be visualised more clearly.

3.6.1.3 Drawing Polygon Punch Hole (Draw Polygon)

The flow of program varies according to the shape of the plate given as input. "If-else" conditional statements are used to run a section of code based on whether or not a particular condition is satisfied. The number of sides and size of the polygon are both provided by the user, and depending on those two factors, the coordinates of the vertices of the polygon are calculated, and then a continuous line called a "Polyline" is created to link all of the vertices in order to form the required polygon. These coordinates are also utilised in the process of drawing the projection lines that are displayed in the front view of the plate. In case of a circle, they are diametrically opposite ends.

3.6.1.4 Drawing Key Shape Punch Hole (Draw Keyshape)

The key shape is created by sketching two semi-circular arcs and two lines that connect the points where the arcs meet. The drawing maintains the key shape's horizontal orientation throughout its entirety. The user is required to select the key shape from the list as the desired shape of punch and provide the inputs for the dimensions of the key shape. The projection lines of the punch hole are produced in the front view based on the dimensions and coordinates of the key shape (using the "keyshape_projection_lines" function).

3.6.2 Drawing Die Plates

3.6.2.1 Drawing of Bottom Plate

The user needs to provide inputs like external dimensions of the bottom plate (based on the shape of plate) and distance of bolt holes from the centre. Then based on the shape of the punch hole user needs to provide the required inputs, such as input for number of sides of polygon (for polygon shape). After getting all the required inputs the top view and front view of the plate are drawn on the drawing sheet in AutoCAD keeping margins from all sides. For bottom plate the external shape, punch shape, bolt holes and guide pillar holes are drawn in the front view. AddCircle(x, y, radius) command is used to draw a circle. AddPolyline() command is used to draw a polyline or continuous line joining multiple points; this is used to draw polygons like rectangle shape of plate and punch hole. The functions created are called to draw the features in the exact locations. Along with this the front view of the plate is also drawn to mainly show the thickness of the plate and to show the depth of holes. The front view is drawn below the Top view and a gap is kept between both the views for clean and simple drawing layout. AddPolyline() command is used to draw a rectangle for the front view of the plate.

3.6.2.2 Drawing of Stripper Plate

Taking the diameter of plates as input from the user (all plates excluding the bottom plate have same diameter). Based on the previously given inputs the front view and top view of the stripper plate are drawn. The stripper plate consists of bolt holes and punch hole. The dimensions for bolt holes and number of bolt holes are taken as input from the user. Similar to the process used for bottom plate, the bolt holes and punch hole is drawn on the stripper plate. The stripper plate is drawn beside the bottom plate keeping a gap in between the two plates. The front view of the stripper plate is drawn to show the thickness of the plate, which is calculated in the design calculations.

3.6.2.3 Drawing of Punch Plate

Based on the previously given inputs the front view and top view of the punch plate is drawn. The thickness of punch plate is calculated in the program using the formulae. The punch plate has a flange on the lower side, thickness of flange is taken as 45% of the inner diameter of flange (or the diameter of punch) and width of flange is 1.5 times the width of punch plate. The punch plate has through bolt holes at the same location as the bottom and stripper plate. The bolt holes are drawn using the same process as before by calling the user defined function (circular_pattern). The top view has concentric circles for the punch hole and its flange. The front view has a flange on the lower side and has projection lines for the bolt holes and the punch hole. Similar to the bottom and stripper plate the punch is drawn of the shape and size given by the user as inputs.

3.6.2.4 Drawing of Thrust Plate

Based on the previously given inputs the front view and top view of the thrust plate is drawn. The thickness of thrust plate is calculated in the program using the formulae. The thrust plate has bolt holes and holes for the spring. The number of springs is taken as input from the user and diameter of holes is calculated by the program. The bolt holes and spring holes are drawn using the user defined function (circular_pattern). The front view has projection lines for the bolt holes and spring holes.

3.6.2.5 Drawing of Piercing Die Block

The piercing die block has bolt holes, spring holes and a counterbore hole for the punch. The punch plate has a stepped hole which has counterbore diameter (larger diameter) to fit the punch plate flange and same punch diameter. The bolt holes and spring holes are drawn using the user defined function (circular_pattern). The pitch circle diameter for the spring holes is given as parameter while calling the "circular_pattern" function. The front view has projection lines for the bolt holes, spring holes, and step hole. The height of counterbore is taken as half the thickness of the die block.

The front view of the plates allows for better visualization of the projections, holes, internal contours, depth of holes and thickness of the plate. Furthermore, the dimensions are also added to all the features of the plate from manufacturability point of view.

4.0 Discussion

The creation of a Python-based parametric CAD system, coupled with a Graphical User Interface (GUI), has demonstrated significant efficacy in automating the design process of compound dies. This method markedly decreases the duration allocated to redundant design activities, facilitating greater emphasis on innovation and the creation of bespoke components. The technology is especially beneficial in sectors necessitating bulk customization and batch production, providing an efficient approach for component design. A primary benefit of this technology is its capacity to reduce human error. Automating design parameters guarantees consistency and accuracy in the final manufacturing designs. The decrease in errors enhances the quality of the components and mitigates the danger of inconsistencies during production. The incorporation of modern methods such as Finite Element Analysis (FEA) facilitates

Die Design Representation

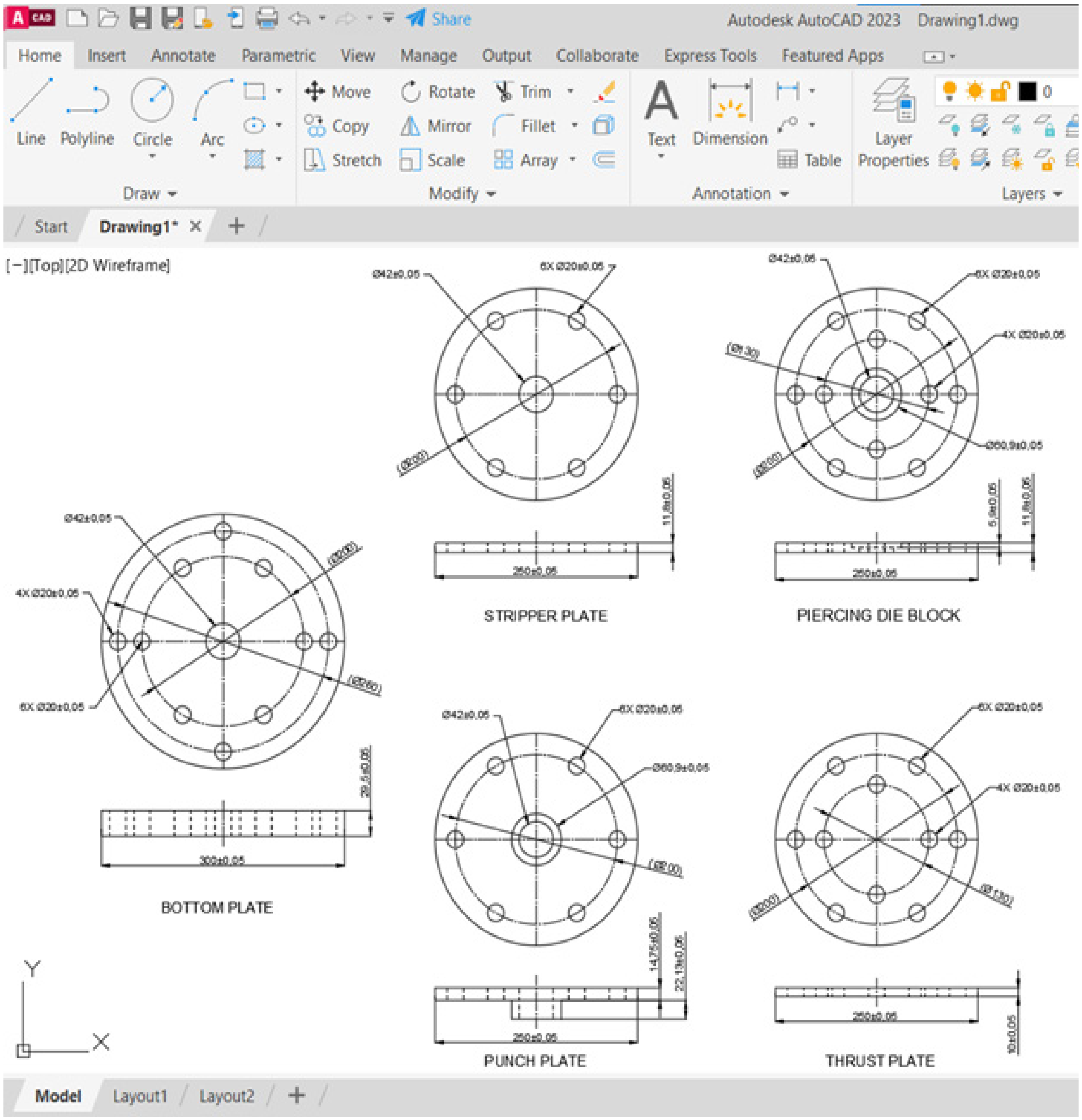


Figure 15. Case-1: Circular Die with circular punch hole.

the prediction of stress and vibration, hence improving the lifetime and reliability of the engineered components.

- The automation of the design process facilitates expedited customization and numerous iterations, rendering the approach more efficient than conventional methods.
- The intuitive GUI enables those with minimal programming expertise to utilize the system efficiently, enhancing its accessibility.
- The system's compatibility with AutoCAD and its affordability renders it especially beneficial for Small and Medium-sized firms (SMEs), allowing

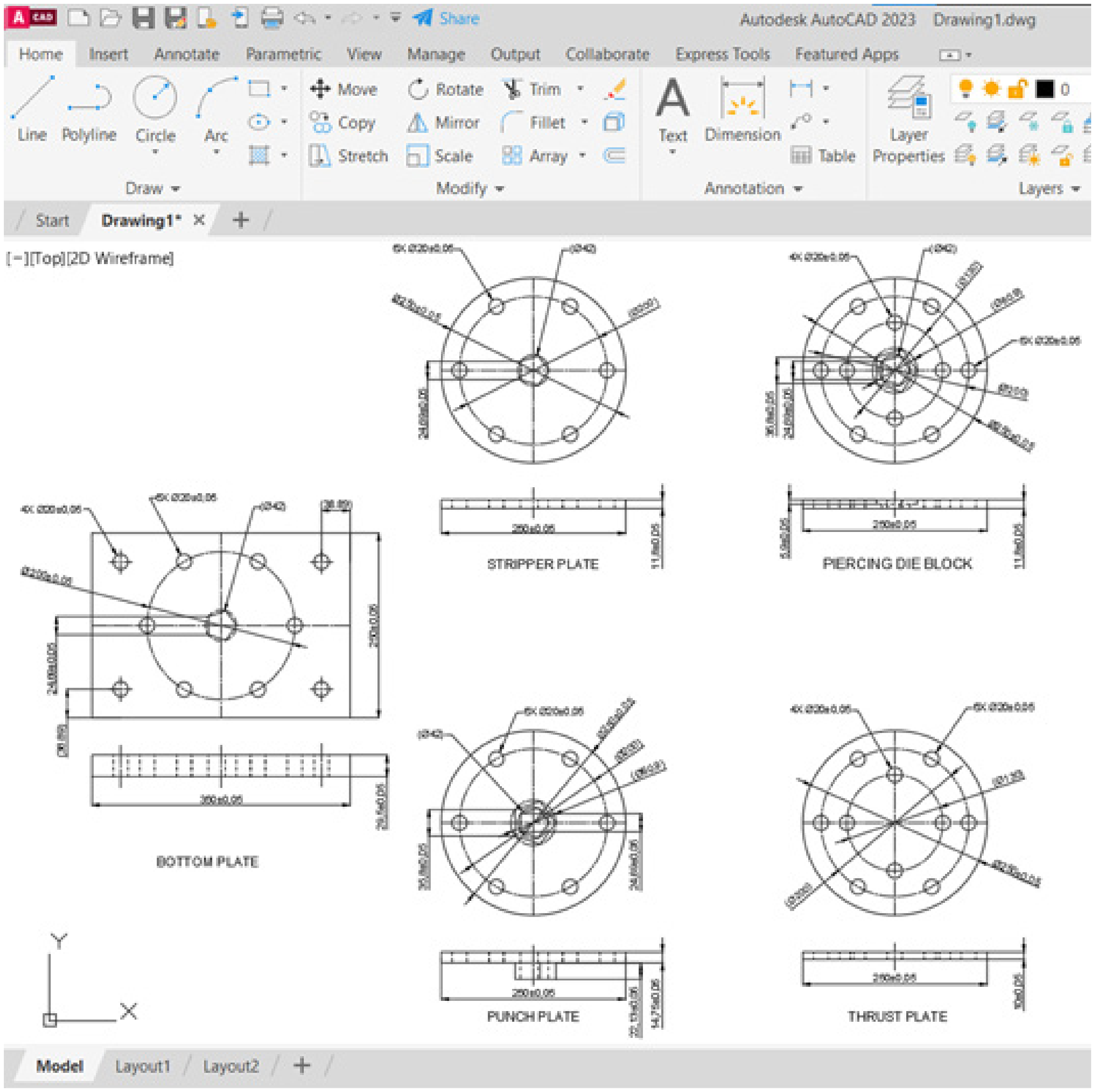


Figure 16. Case-2: Rectangular Die with pentagonal punch hole.

them to use advanced design methodologies without substantial financial outlay.

- The combination of FEA and the possibility for machine learning facilitates predictive modeling, enabling more informed design decisions that enhance component durability and performance.

5. Conclusion

This study has effectively created an automated CAD modeling system utilizing python and the AutoCAD API, specifically designed for the fabrication of compound dies. The technology enhances efficiency in designing customized components while simultaneously increasing accuracy through the automation of repetitive operations. The incorporation of a graphical user interface enhances the system's accessibility and efficiency, particularly for sectors emphasizing bulk customization and batch production. The advantages of this system are particularly significant in small and medium-sized organizations, where cost-efficiency and user-friendliness are essential considerations. The automated design process, along with the incorporation of advanced analytical methods like FEA, improves the accuracy and resilience of the produced components.

- The technology markedly reduces the design duration, facilitating a swifter time-to-market for tailored items.
- Automated computations diminish inaccuracies in design, resulting in more exact and uniform manufacturing designs.
- Subsequent advancement of this system may include machine learning methodologies to enhance the design process, yielding superior accuracy and predictive capabilities in forthcoming applications.

The effective execution of this method represents a crucial advancement in modernizing the design process for compound dies, enhancing accessibility, efficiency, and precision for various industrial applications.

6.0 References

1. Kumar S, Singh R. A knowledge-based system for selection of progressive die components. J Achiev Mater Manuf Eng. 2007; 20(1-2):475-78. https://doi.org/10.1504/IJCMSSE.2007.013846
2. Kashid S, Kumar S. An expert system for selection of components of compound die. J Adv Manuf Syst. 2014; 13(3):181-95. https://doi.org/10.1142/S0219686714500115
3. Kumar S, Singh R. A low-cost knowledge base system framework for progressive die design. J Mater Process Technol. 2004; 153:958-64. https://doi.org/10.1016/j.jmatprotec.2004.04.236
4. Hambli R, Guerin F. Application of a neural network for optimum clearance prediction in sheet metal blanking processes. Int J Adv Manuf Technol. 2003; 22(1-2):20-5. https://doi.org/10.1007/s00170-002-1437-5
5. Nee AYC, Foong KY. Some considerations in the design and automatic staging of progressive dies. J Mater Process Technol. 1992; 29(1-3):147-58. https://doi.org/10.1016/0924-0136(92)90431-Q
6. bin Ab Kadir AR, Wan NNI, bin Said MS, binti Zubir B, bin Ibrahim MZ, Krishnan P. Design, and analysis of punch and die of a micro blanking tool. Int J Recent Technol Eng. 2019; 8(4):827-33. https://doi.org/10.35940/ijrte.D7416.118419
7. Kumar S, Singh R. An automated design system for progressive die. Expert Syst Appl. 2011; 38(4):4482-9. https://doi.org/10.1016/j.eswa.2010.09.121
8. Shaheen W, Kanapathipillai S, Mathew P, Prusty BG. Optimization of compound die piercing punches and double cutting process parameters using finite element analysis. Proc Inst Mech Eng Part B J Eng Manuf. 2020; 234(1-2):3-13. https://doi.org/10.1177/0954405419855507
9. Kadarno P, Mori KI, Abe Y, Abe T. Punching process including thickening of hole edge for improvement of fatigue strength of ultra-high strength steel sheet. Manuf Rev. 2014; 1:4. https://doi.org/10.1051/mfreview/2014003
10. Kumar RM. Optimization of die design parameters in blanking operation using genetic algorithm. i-Manager's J Mech Eng. 2016; 6(2):30. https://doi.org/10.26634/jme.6.2.4890
11. Skampardonis N, Tsirkas S, Grammatikopoulos S. Design, and analysis of an industrial, progressive die for cutting and forming. 2021. https://doi.org/10.21203/rs.3.rs-262802/v1
12. Shaheen W, Kanapathipillai S, Mathew P, Prusty BG. Optimization of compound die piercing punches and double cutting process parameters using finite element analysis. Proc Inst Mech Eng Part B J Eng Manuf. 2020; 234(1-2):3-13. https://doi.org/10.1177/0954405419855507
13. Mehta AM, Rus D. An end-to-end system for designing mechanical structures for print-and-fold robots. In: Proceedings of IEEE International Conference on Robotics and Automation (ICRA); 2014; 1460-5. https://doi.org/10.1109/ICRA.2014.6907044
14. Machado F, Malpica N, Borromeo S. Parametric CAD modeling for open-source scientific hardware:

comparing OpenSCAD and FreeCAD Python scripts. PLoS One. 2019; 14(12):1-30. https://doi.org/10.1371/journal.pone.0225795

15. Tezzele M, Demo N, Mola A, Rozza G. PyGeM: Python geometrical morphing. Softw Impacts. 2021; 7:100047. https://doi.org/10.1016/j.simpa.2020.100047
16. Salunkhe S, Kumar S. Applications of artificial neural network to sheet metal work - A review. Am J Intell Syst. 2013; 2(7):168-76. https://doi.org/10.5923/j.ajis.20120207.03
17. Zuo ZH, Xie YM. A simple and compact Python code for complex 3D topology optimization. Adv Eng Softw. 2015; 85:1-11. https://doi.org/10.1016/j.advengsoft.2015.02.006
18. Agarwal D, Robinson TT, Armstrong CG. A CAD based framework for optimizing performance while ensuring assembly fit. In: Wang S, Price M, Lim M, Jin Y, Luo Y, Chen R, editors. Recent advances in intelligent manufacturing. Singapore: Springer; 2018; 1-12. https://doi.org/10.1007/978-981-13-2396-6_7
19. Mukundakrishnan B, Rajmohan N, Rajnarayan DG, Fugal S. A script-based CAD system for aerodynamic design. In: AIAA Aviation 2019 Forum. 2019. https://doi.org/10.2514/6.2019-3069
20. Mathur A, Pirron M, Zufferey D. Interactive programming for parametric CAD. Comput Graph Forum. 2020; 00(00):1-18.