

PAPER • OPEN ACCESS

Computerized system based on FreeCAD for geometric simulation of the oil and gas equipment thread turning

To cite this article: V B Kopei *et al* 2019 *IOP Conf. Ser.: Mater. Sci. Eng.* **477** 012032

View the [article online](#) for updates and enhancements.

Computerized system based on FreeCAD for geometric simulation of the oil and gas equipment thread turning

V B Kopei, O R Onysko and V G Panchuk

Ivano-Frankivsk National Technical University of Oil and Gas, Department of Computerized Machine-Building Production, Karpatska str., no. 15, 76019 Ivano-Frankivsk, Ukraine

E-mail: vkopey@gmail.com

Abstract. This paper describes the principles of the computerized system for creating geometric models of threaded joints with various geometric deviations, which arise during the thread turning, and demonstrates the use of these models to determine the dependencies of reliability indicators on these deviations by the finite element method. Using the developed system, based on Python and FreeCAD, axisymmetric finite element models of drilling tool joints and sucker rod couplings with deviations of the profile angle and the thread pitch were created and analyzed.

1. Statement of the problem

Requirements for reliability of threaded connections of oil and gas equipment are constantly increasing. It is possible to determine the dependencies of reliability indicators on different geometric parameters by finite-element (FE) modeling of the stress-strain state [1-3]. But for conducting of such studies it is necessary to construct a parametric geometric model of a threaded connection.

One of the main parameters that affects the reliability of threaded connections is their geometric accuracy. Geometric accuracy depends on the machining errors of thread turning (machine errors, load induced errors, thermal growth errors, error caused by tool wear and other errors). The purpose of this work is the development of the computerized system, which is designed to build models of threaded joints with various geometric deviations arising during the thread turning, and the demonstration of the use of these models to determine the dependencies of reliability indicators on these deviations. This rather complicated geometric problem can be solved with the help of modern geometric modeling kernels, in particular Open CASCADE Technology, on which FreeCAD is based.

2. Description of the system

The computerized system being developed is based on the open-source software Python 2.7 and FreeCAD 0.17. Python is a high-level general-purpose programming language. FreeCAD is a parametric 3D CAD based on Open CASCADE Technology 7.2.0 geometry kernel with Python API. Open CASCADE Technology is a software development kit intended for development of applications dealing with 3D CAD data. Note that if the FreeCAD capabilities for creating a model are not enough, you can use the PythonOCC library [4], in which the Open CASCADE kernel is fully available.

The developed system has the ability to create the geometrical models of threads with different computational complexity and adequacy (Table 1). The most adequate models for the FE simulation of the stress-strain state are 3D models with helical threads. But they have the highest computational



complexity for geometric and FE simulation. Planar geometric models are often used to create axisymmetric FE models [5].

Table 1. Computational complexity and adequacy of thread models

3D real thread (helical)	3D quasi-thread (circular grooves)	2D planar
high	medium	low

We describe the program using the drilling tool joint ZN-80 GOST 5286 model as an example. This is the joint with the Z-66 GOST 28487 thread (2 3/8 REG API Spec 7 equivalent). First import the necessary modules.

```
import FreeCAD as App
import FreeCADGui as Gui
import Part
import numpy as np
from dims import d
```

The functions in the `FreeCAD` module allow working with FreeCAD documents. `FreeCADGui` module contains everything related to the Graphical User Interface and the 3D views [6]. The `Part` module is the direct connection between FreeCAD and the Open CASCADE kernel. With this module, you can create BREP (boundary representation) topological shapes (Vertex, Edge, Wire, Face, Shell, Solid, Compsolid, Compound) [6]. The `numpy` module provides the routines for fast operations on multidimensional arrays. The `dims` module contains an object `d` which attributes are equal to different parameters of the joint. If you are not using an internal Python interpreter, you must specify the path to the FreeCAD modules.

```
import sys
sys.path.append("e:\\FreeCAD 0.17x64\\bin")
```

We can use parametric FreeCAD sketches to simplify the development of the models. The sketches of tools, workpieces of pin and box are shown on the Figure 1.

The function `rebuildSketch(dim, sk)` can be used to rebuild the parametric sketch. The `dim` parameter is a dictionary with pairs (sketch constraint ID, dimension value), and the `sk` parameter is the name of a sketch. The function calls `setDatum(k, v)` method for each pair in `dim` to set value `v` of a distance or angle constraint with ID `k`, rebuilds the document and returns the face.

```
def rebuildSketch(dim, sk):
    doc=App.getDocument("Sketches") #get the document object by name
    sketch=doc.__getattr__(sk) #get the sketch object by name
    # for each pair (sketch constraint ID, dimension value)
    for k,v in dim.items():
        sketch.setDatum(k,v) # set dimension value
    doc.recompute() # rebuild the document
    w=sketch.Shape.Wires[0] # get the first wire
    f=Part.Face(w) # create the face
    return f
```

The `revolve` function creates the solid shape by rotating the face `f` around the Y axis.


```
def revolve(f):
    return f.revolve(App.Vector(0,0,0), App.Vector(0,1,0))
```

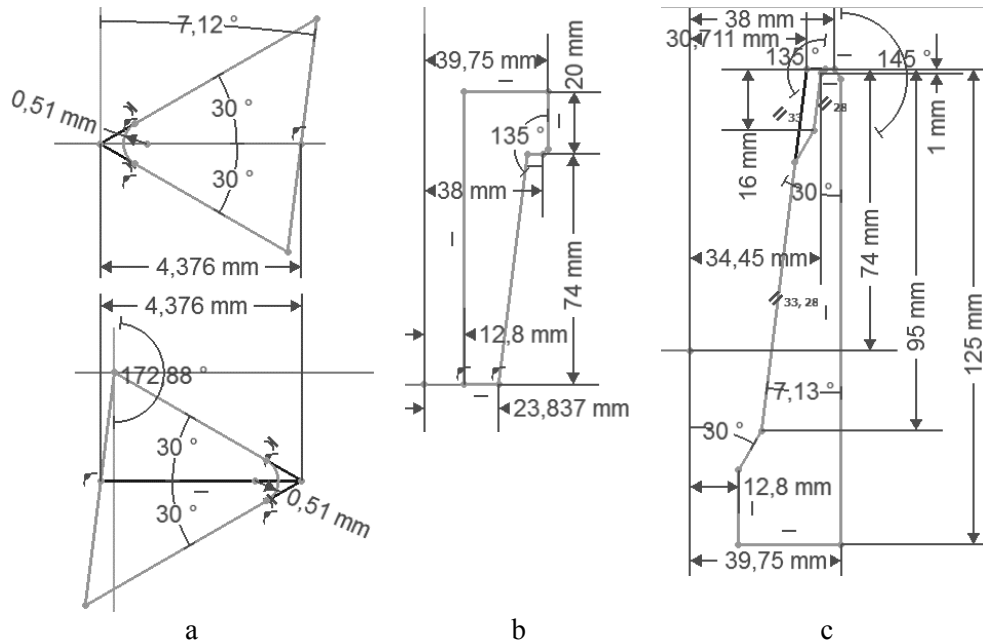


Figure 1. Parametric sketches of tools (a), pin workpiece (b) and box workpiece (b)

To create the ideal helix, use the `makeLongHelix` function from the `Part` module. To avoid errors when building long threads, do not use the `makeHelix` function. It is necessary to rotate the helix so that its axis coincides with the `Y` axis. These operations are executed by the function `helix(r, h, p, fi)`, where `r` is the radius, `h` is the height, `p` is the pitch, `fi` is the cone angle (rad).

```
def helix(r, h, p, fi):
    w=Part.makeLongHelix(p, h, r, np.degrees(fi))
    w.rotate(App.Vector(0,0,0), App.Vector(1,0,0), -90)
    return w
```

The thread on the pin (or box) is created by the `makeThread(f, h, s)` function, where `f` is a tool face, `h` is a helix wire, `s` is a workpiece solid.

```
def makeThread(f, h, s):
    s2=h.makePipeShell(f.Wires, True, True) # the helical solid
    s=s.cut(s2) # Boolean cut operation
    return s
```

With the help of these functions, it is possible to create nominal solids of the pin or box (Figure 2).

```
App.open(u"D:/3/Sketches.FCStd") # open the document with sketches
fA0=rebuildSketch(dim={6:d.fi, 9:d.H}, sk='Sketch') # tool face
fA0.translate(d._v2) # move the tool to the starting position
sA0=revolve(rebuildSketch(dim={20:d.fi, 16:d.d3/2}, sk='Sketch001'))
# workpiece solid
hA0=helix(r=d._r, h=d.l4-12, p=d.P, fi=d.fi) # helix wire
hA0.translate((0,d.l3-d.l4,0)) # move the wire
```



```
sA1=makeThread(fA0, hA0, sA0) # pin with thread
```

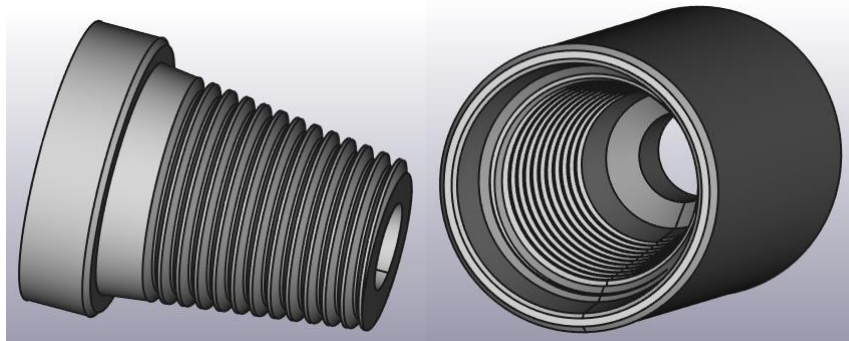


Figure 2. The 3D model of the drilling tool joint ZN-80 (GOST 5286)

Now simulate threads with different deviations. To simulate various machining errors in the program, it is possible to change the values of the geometric parameters of the workpiece and tool and the parameters of the tool path relative to the workpiece. The trajectory of the relative movement of the workpiece and the tool is a conical helical curve with parametric equations (1). In this equations $a = \tan(\varphi)p/(2\pi)$, $b = p/(2\pi)$, t - parameter (rad), r - start radius, φ - cone angle (rad), p - pitch.

$$\begin{aligned} x &= (r+at)\cos(-t), \\ z &= (r+at)\sin(-t), \\ y &= bt. \end{aligned} \quad (1)$$

The function `helixPoints(r, h, p, fi, n)` returns an array of points of the conical helix. Parameter r is the smaller radius, h - height, p - step, fi - cone angle, n - the number of points in one turn. Here t, x, y, z - numpy arrays.

```
def helixPoints(r, h, p, fi, n):
    a=np.tan(fi)*p/(2*pi)
    b=p/(2*pi)
    k=h/p # number of turns
    N=int(n*k) # total number of points
    t=np.linspace(0,h/b,N)
    x=(r+a*t)*np.cos(-t)
    z=(r+a*t)*np.sin(-t)
    y=b*t # helix axis
    return zip(t,x,y,z)
```

You can simulate various systematic or random machining errors by using the helix equation with deviations (2), where x_0, y_0, z_0 - coordinates of points of a nominal helix, $A_x, A_y, A_z, \Delta x, \Delta y, \Delta z$ - constants or random variables. For nominal thread $A_x = A_y = A_z = 1, \Delta x = \Delta y = \Delta z = 0$. For example, the constant A_x can be used to simulate the spindle radial run-out, and the variable Δy - to simulate the pitch error. To simulate the random machining errors, the variables $A_x, A_y, A_z, \Delta x, \Delta y, \Delta z$ must be random variables with some distributions.

$$\begin{aligned}
 x &= A_x x_0 + \Delta x, \\
 z &= A_z z_0 + \Delta z, \\
 y &= A_y y_0 + \Delta y.
 \end{aligned}
 \tag{2}$$

The function `perror(points)` returns the points coordinates of the helix with the deviations, where Δx , Δy , Δz are random variables with a triangular distribution. The `points` parameter is the points of the nominal helix. The user can modify this function to simulate other errors.

```
def perror(points):
    epoints=[] # points of the helix with the deviations
    d=0.2 # triangular distribution parameter
    for t,x,y,z in points:
        x=x+np.random.triangular(-d,0,d)
        y=y+np.random.triangular(-d,0,d)
        z=z+np.random.triangular(-d,0,d)
        epoints.append((t,x,y,z))
    return epoints
```

The thread creation algorithm is based on the approximation of the tool movement trajectory by splines or polylines. To approximate a helix with deviations by a polyline, use the `helix2(points)` function, where `points` are the points of the helix with deviations.

```
def helix2(points):
    pts=[(x,y,z) for t,x,y,z in points]
    h=Part.makePolygon(pts) # polyline wire
    return h
```

It is possible to create a thread with deviations in different ways. For example, the `makePipeShell` function can be used to create a helical body using a spline (or polyline) and a list of tool profiles at each point. But the most simple and universal way is to use shape operations "loft" and "cut" for each point `helix2(points)` (Figure 3). The function `makeThread2(f, h, s)` creates such thread by using `copy`, `rotate`, `translate`, `makeLoft` and `cut` functions. The parameter `f` is the face of the tool, `h` is the helix wire, `s` is the workpiece solid. Calculations can take a long time if there are many points.

```
def makeThread2(f, h, s):
    w=f.Wires[0] # tool wire
    points=[(v.X,v.Y,v.Z) for v in h.Vertexes] # helix points
    W=[] # profiles at different points of the helix
    for x,y,z in points: # for each point
        w_=w.copy() # tool wire copy
        # tool rotation angle relative to the workpiece
        t=np.arctan2(-z,x)
        # rotate the tool around an Y axis
        w_.rotate(App.Vector(0,0,0),App.Vector(0,1,0),np.degrees(t))
        w_.translate(App.Vector(x,y,z)) # move the tool to x,y,z
        W.append(w_.copy()) # append the tool profile to the list W
    if len(W)>1:
        # create a solid shape using two adjacent profiles
        st=Part.makeLoft([W[-2],W[-1]],True)
```



```
s=s.cut(st) # Boolean cut for workpiece and shape st
return s,W
```

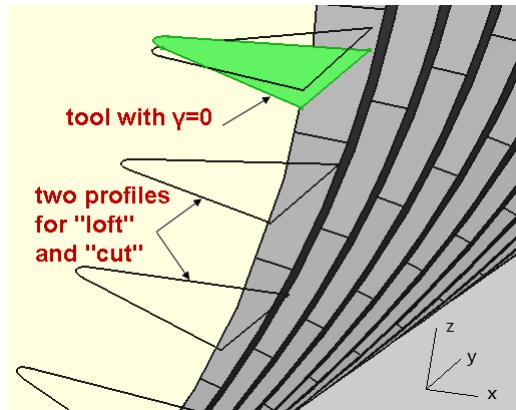


Figure 3. Creating threads with deviations

You can also simulate the slope of the rake face of the cutting tool and nonzero rake angle γ . To do this, rotate the face by an angle γ around the line, that parallel to the helix axis and passes through the middle of the profile (Figure 3). The following code creates a pin with the thread that has deviations.

```
fA3.rotate(App.Vector(0,0,0),App.Vector(0,1,0),-10) # rake angle
points=perror(points) # points of a helix with deviations
hA3=helix2(points) # helix with deviations
hA3.translate((0,d.l3-d.l4,0)) # move the helix
sA4,W=makeThread2(fA3, hA3, sA0) # pin with thread
```

It is possible to calculate the volume or area of the symmetrical difference (XOR or union without the intersection) for nominal and real threads. Below you can see the corresponding FreeCAD API code. The testing of the thread suitability with deviations can be realized by Boolean operations on the maximum permissible and real shapes.

```
# XY plane:
f=Part.makePlane(400,400,App.Vector(-200,-200,0),App.Vector(0,0,1))
f1=sA3.common(f).Faces[0] # axial section of a pin
f2=sA4.common(f).Faces[0] # axial section of a box
f3=f1.cut(f2).fuse(f2.cut(f1)) # Boolean XOR operation
print f3.Area # XOR area
```

Figure 4 shows the maximum deviations between nominal ($\gamma=0^\circ$, $A_x=0$, $\Delta y=0$) and real profiles when there is combination of different errors. The nominal profile is indicated by an arrow, and the XOR area is between the profiles. Deviations are almost not visible in Figure 4(a). In this case, according to the data [7], the deviation of the profile angle is not more than 0.47° .

Figure 5 shows the 3D nominal thread ($\Delta x=0$, $\Delta y=0$, $\Delta z=0$) and the 3D thread with deviations by equation 2, where Δx , Δy , Δz - random variables with a triangular distribution (with parameters $a=-0.2$ mm, $c=0$ mm, $b=0.2$ mm). Calculated XOR volume is equal 133.79 mm^3 .

3. Use of geometric models for finite element simulation

The models of threaded connections constructed with this program can be used to simulate the stress-strain state by the finite element method and justify the values of the thread tolerances. We used an open source finite element mesh generator Gmsh 2.7 [8] and a FEA software CalculiX 2.12 [9].

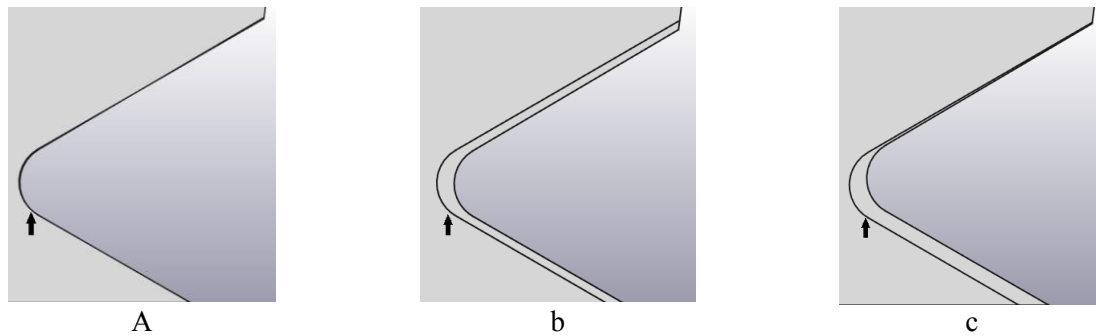


Figure 4. Deviations between nominal and real profiles when there is combination of different errors: $\gamma=-10^\circ$, $\Delta y=0$, $A_x=0$, XOR area= 0.0353 mm^2 (a); $\gamma=-10^\circ$, $\Delta y=0$, $A_x=1.01 \text{ mm}$, XOR area= 0.8412 mm^2 (b); $\gamma=-10^\circ$, $\Delta y=0.1 \text{ mm}$, $A_x=1.01 \text{ mm}$, XOR area= 0.7744 mm^2 (c)

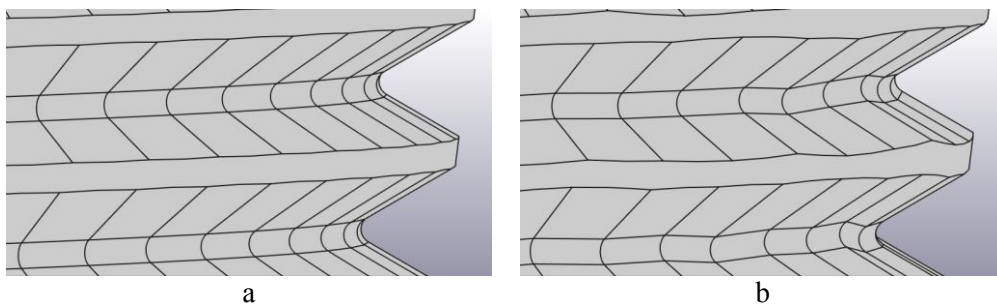


Figure 5. The nominal thread (a) and the thread with random deviations (b)

Based on the planar geometric model of the drilling tool joint ZN-80, obtained with the help of the developed program, an axisymmetric FE model is constructed. The material of the parts is steel 40XH (GOST 4543) with the Young's modulus $E=2.1e5 \text{ MPa}$, Poisson's ratio $\nu=0.28$, yield strength $\sigma_y=735 \text{ MPa}$, tensile strength $\sigma_t=882 \text{ MPa}$. Material plasticity and friction are simulated. The size of the mesh elements is 0.2 mm . Two steps of the external axial tensile load were created: $F_{\min}=0 \text{ N}$ and $F_{\max}=1 \text{ MN}$. The joint make-up was simulated by axial deformation Δ of the box shoulder. For each tool joint design, a value of Δ was set such that the contact pressure on joint shoulders was 274 MPa for F_{\max} . For the standard design $\Delta = 0.2 \text{ mm}$. For the approximate calculation of the fatigue safety factor D , the dependence of Sines [10], [11], with a material endurance limit of 207 MPa was used. Note that D values can only be used for the relative comparison of different designs, but not for calculating cyclic durability values [1]. A smaller value of D only indicates a lower cyclic durability.

The Figure 6 shows the values of equivalent von Mises stress σ_v and fatigue safety factor D as a function of the deviation of the thread profile angles of the pin from the nominal value. Deviations are determined by the formula $\Delta\varphi_{1,2} = \varphi_{1,2} - 30^\circ$, where $\varphi_{1,2}$ is the angle between the pin profile side (index 1 means unloaded side, 2 - loaded side) and the line perpendicular to the axis, 30° is the nominal value. Much larger contact stresses arise on the loaded side 2 (the bottom side in Figure 7) if the joint is subjected to an axial tensile load F_{\max} . Deviations $\Delta\varphi_{1,2}$ can be caused by a nonzero rake angle γ of the tool and machining error, caused by tool wear. The deviation of the unloaded side $\Delta\varphi_1$ of the pin profile can be caused by the wear of the left cutting edge of the tool. As a rule, the wear of the left cutting edge near the corner is greatest, and this increases the angle of the unloaded side of the thread profile φ_1 .

Negative values of the deviations are the cause of a sharp increase of σ_v values and decrease of D values (Figure 6). Areas with very small D values appear in the root of the box thread, and D values of the pin decreases insignificantly (Figure 7). Positive values of the deviations of both sides $\Delta\varphi_{1,2}$ are somewhat smaller decrease D values. The D value of the first root in the pin thread decreases most of all (Figure 7). Positive values of the deviations of the unloaded side $\Delta\varphi_1$ almost do not change the stress and fatigue strength of the joint (Figure 7). Thus, the tool wear along the left cutting edge has no influence on the fatigue strength of the joint.

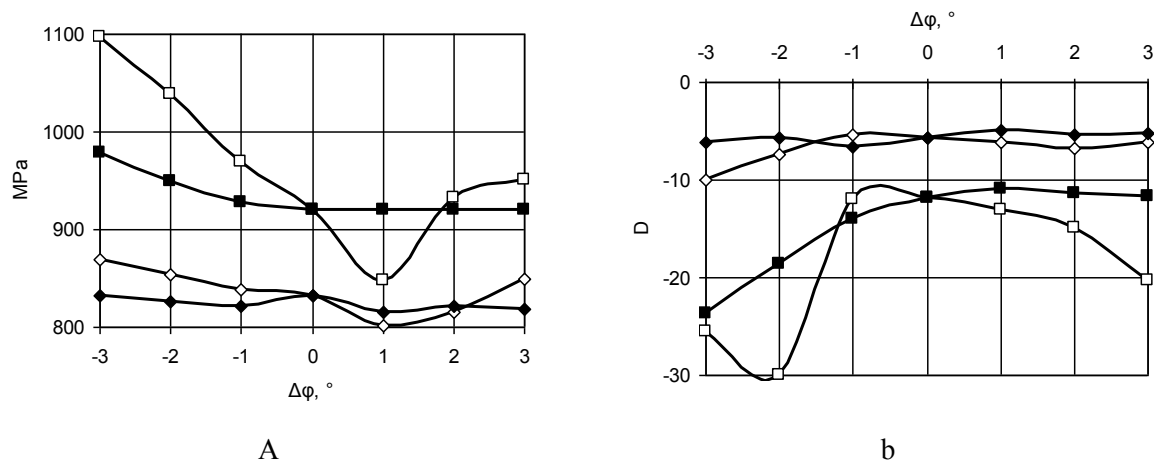


Figure 6. Von Mises stress at F_{\max} (a) and fatigue safety factor D (b) as a function of $\Delta\varphi_1$ (■, ◆) and $\Delta\varphi_{1,2}$ (□, ◇): □, ■ - maximum value in the joint; ◇, ◆ - value at the first loaded root of the pin thread

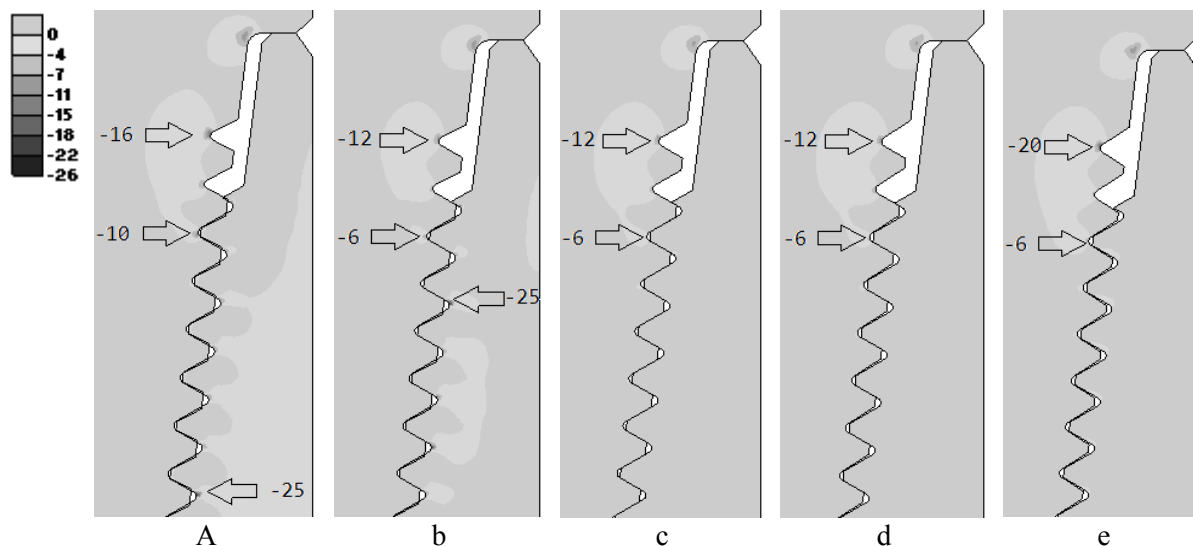


Figure 7. Distribution of the fatigue safety factor D for different $\Delta\varphi_{1,2}$ values: $\Delta\varphi_{1,2} = -3^\circ$ (a); $\Delta\varphi_1 = -3^\circ$ (b); $\Delta\varphi_{1,2} = 0^\circ$ (c); $\Delta\varphi_1 = 3^\circ$ (d); $\Delta\varphi_{1,2} = 3^\circ$ (e)

Using the developed program axisymmetric FE models of 19 mm GOST 13877 (3/4 in. API Spec 11B equivalent) sucker rod couplings were also constructed. According to GOST 13877, the pin thread must be rolled, and the box thread can be rolled or manufactured by another method, for example, by a single-point cutting tool. The material of the parts is steel 40 (GOST 1050) with

$E=2.1\text{e}5$ MPa, $\nu=0.28$, $\sigma_y=314$ MPa, $\sigma_t=559$ MPa. Material plasticity and friction are simulated. The size of the mesh elements is 0.2 mm. Two steps of the external axial tensile load were created, which form in the rod body the stress $P_{\min} = 0$ MPa and $P_{\max} = 170$ MPa. The joint make-up was simulated by axial deformation Δ of the box shoulder. For each coupling design, a value of Δ was set such that the contact pressure on joint shoulders was 312 MPa for P_{\max} . For the standard design $\Delta = 0.12$ mm.

In Figure 8 the von Mises stress and fatigue safety factor D as a function of angle deviation $\Delta\varphi = \Delta\varphi_1 = \Delta\varphi_2$ of box thread profile are shown. The index 1 denotes the unloaded profile side and 2 - the loaded one. It is noticeable that negative $\Delta\varphi$ values cause a slight decrease of stress values and an increase of D values in the first loaded root of the pin thread. But contact stresses grow in the zone of the minimum diameter of the box thread (Figure 9(a)). Positive $\Delta\varphi$ values sharply reduce the D values in the first root of the pin thread, but almost do not change the von Mises stress values in it for P_{\max} .

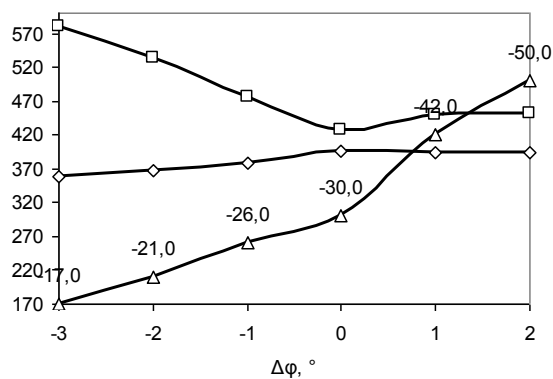


Figure 8. Von Mises stress at P_{\max} (□, ◇) and fatigue safety factor D (Δ) as a function of $\Delta\varphi$: □ - maximum value in the joint; ◇, Δ - value at the first loaded root of the pin thread

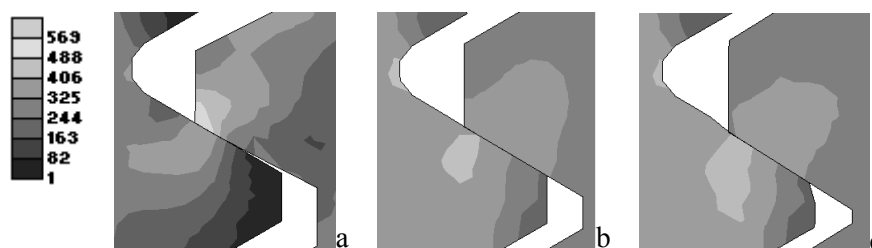


Figure 9. Von Mises stress distribution (MPa) in the first thread of sucker rod coupling at P_{\max} for different $\Delta\varphi_{1,2}$ values: -3° (a); 0° (b); 2° (c)

The Figure 10 shows the von Mises stress and fatigue safety factor D as a function of Δp - the magnitude of decrease the box thread pitch on the each thread turn (starting from the first). Positive Δp values decrease the pitch of the box thread, and negative values increase the pitch. Positive Δp values are the cause of a sharp decrease of D values due to non uniform load to each thread (Figure 11, 12). Stresses, which are shown in the Figure 12(a), may occur after multiple make-up of the joint as a result of plastic deformation and wear of the first threads. Negative Δp values equalize loads in the thread, somewhat reduce the von Mises stresses in the first loaded root of the pin thread (Figures 10-12) and somewhat reduce the area with negative D values (Figure 13). Therefore, you should avoid

$\Delta p > 0$ but try to provide $\Delta p < 0$. For single-point threading on a CNC machine, this can be achieved by increasing the feed by the Δp value per one revolution of the spindle.

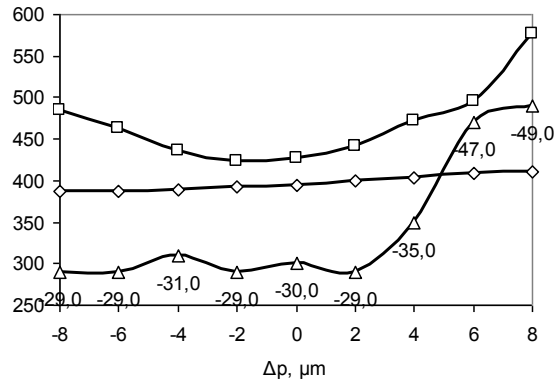


Figure 10. Von Mises stress at P_{max} (\square , \diamond) and fatigue safety factor D (Δ) as a function of Δp : \square - maximum value in the joint; \diamond , Δ - value at the first loaded root of the pin thread

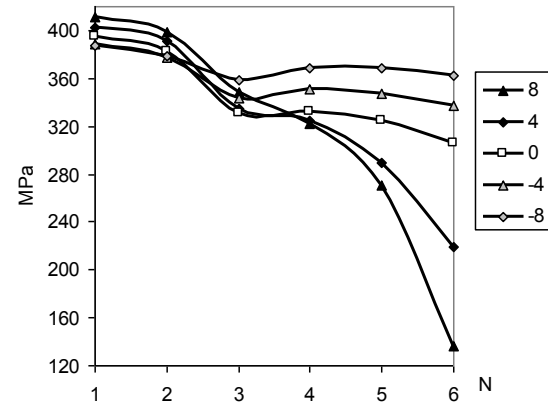


Figure 11. The values of the von Mises stress at P_{max} in the root N of the pin thread for different Δp values (μm)

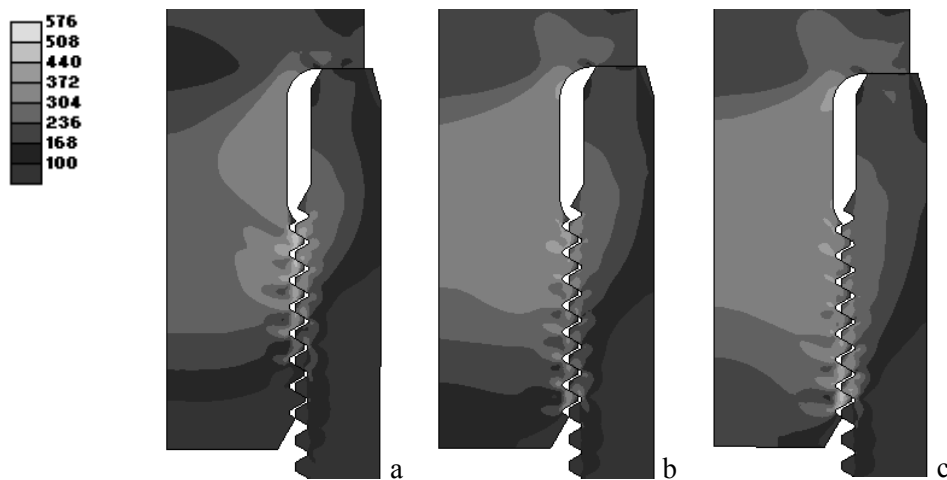


Figure 12. Von Mises stress distribution (MPa) in the sucker rod coupling at P_{max} for different Δp values: $8 \mu\text{m}$ (a); $0 \mu\text{m}$ (b); $-8 \mu\text{m}$ (c)

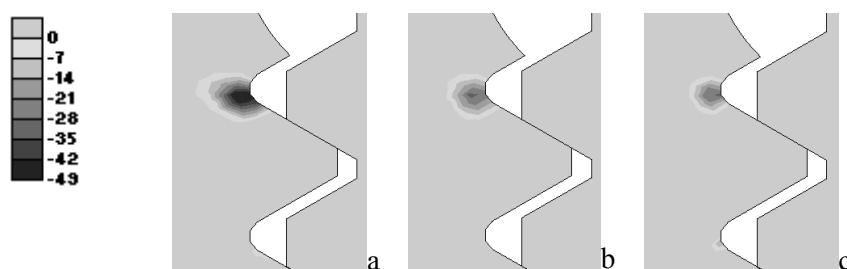


Figure 13. Distribution of the fatigue safety factor D in the first thread of sucker rod coupling for different Δp values: $8 \mu\text{m}$ (a); $0 \mu\text{m}$ (b); $-8 \mu\text{m}$ (c)

4. Conclusions

The developed system can be used to construct 3D and 2D geometric models of threaded connections with deviations, justify their tolerances and optimize the geometric parameters with additional software for finite element analysis. You can download the source code by the following address: <https://github.com/vkopey/Thread-turning-simulator>

References

- [1] Hoffman E L 1997 *Finite element analysis of sucker rod couplings with guidelines for improving fatigue life* Sandia report SAND97-1652 (Albuquerque, New Mexico: Sandia National Laboratories) p 66
- [2] Mackerle J 2003 Finite element analysis of fastening and joining: A bibliography (1990–2002) *International Journal of Pressure Vessels and Piping* **80** 253–71
- [3] Galle T, Van Wittenberghe J, Camelia Julia F, De Waele W and De Baets P 2013 Effect of load flank angle modifications on the structural integrity of buttress threaded connections *Sustainable Construction and Design* **4**(1)
- [4] ***pythonOCC, 3D CAD/CAE/PLM development framework for the Python programming language Available from: <http://www.pythonocc.org>
- [5] Kopei V B 2010 Skinchenno-elementnyi analiz ta optymizatsiia rizbovykh ziednan *Visnyk Natsionalnoho tekhnichnoho universytetu Ukrainy «Kyivskiy politekhnichnyi instytut». Seriya mashynobuduvannia* **58** 42-7 in Ukrainian
- [6] ***FreeCAD Documentation Available from: http://www.freecadweb.org/wiki/Main_Page
- [7] Onysko O R 2017 Pro funktsionalnu zalezhnist velychyny polovynnoho kuta profilu zamkovo narizi vid velychyn perednoho kuta, kuta nakhyly ta polovynnoho kuta profilu rizalnoi kromky riztsia *Visnyk natsionalnoho universytetu «Lvivska politekhnika». Optymizatsiia vyrobnychkh protsesiv i tekhnichniy kontrol u mashynobuduvanni ta pryladobuduvanni* **867** 19-28 (in Ukrainian)
- [8] Geuzaine C and Remacle J F 2009 Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities *International Journal for Numerical Methods in Engineering* **79**(11) 1309-31
- [9] Dhondt G 2004 *The finite element method for three-dimensional thermomechanical applications* (Wiley) p 362
- [10] Sines G 1955 *Failure of materials under combined repeated stresses with superimposed static stresses* Technical Note 3495 (Washington: National Advisory Committee for Aeronautics) p 70
- [11] Sines G 1959 Behavior of metals under complex static and alternating stresses, in *Metal Fatigue* eds G Sines and J L Waisman (New York: McGraw-Hill) 145-69