Case Study Cyber Physical Production System Using AM

INTERNAL GEAR PUMP

User Manual for Developing Internal Gear Pump
Components with 3D Printing using LUA Script in IceSl

OPTIMAL GEAR SOLUTION

Dinesh Thirumurugan (00819285)
Anchana Radhakrishnan Nair (00812715)
Alekh Phadnis (00812991)
Jerin Abraham James (00813298)

Research Advisor:
Prof. Dr. -Ing. Stefan Scherbath

TECHNISCHE HOCHSCHULE DEGGENDORF

June 06, 2021

Table of Contents

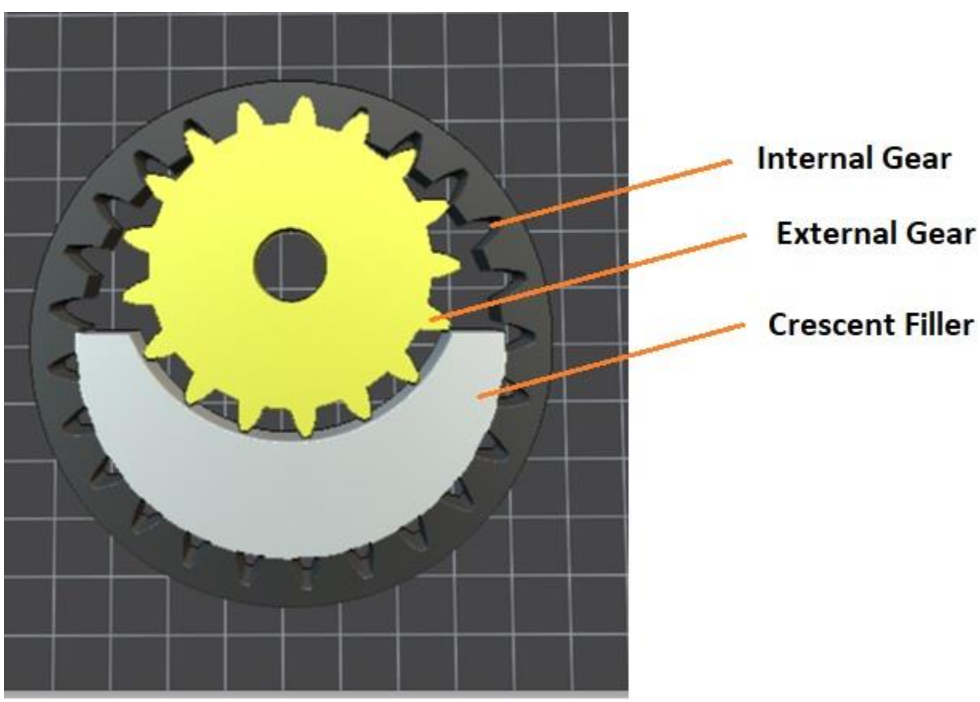
Introduction

The user manual helps to develop the Internal Gear Pump components using Lua scripts in IceSl for 3D printing. This document guides the users to develop important components required to put together an Internal Gear Pump assembly with customized parameters.

This user manual is structured with an objective for the user to be able to flexibly interpret various parameters, analyze the performance, and also able to reproduce the entire demonstrated parametrical model in the Lua platform. Here also included in the documentation are the instructions for extracting necessary '.stl' files and G-codes allowing for cross-platform implementations and 3D printing of the components.

The demonstrated model [Fig1] includes modelling of three main components of the internal gear pump namely:

1) External Gear
2) Internal Gear
3) Crescent



[Fig 1]: *Internal Gear Pump -Components*

Script Overview

The multi-functional script 'Internal_Gear_Pump_LuaScript.lua' by Optimal Gear Solutions allows the users to generate the entire assembly together with their inter-dependencies as well as individual components. The .lua file can be read/edited using text editors supported by IceSL Slicer or with IceSL-forge.

Note: The Internal Gear Pump modelled in this manual perceives the involute spur gear profile.

The scripting structure can be segregated in following sections:

1. Parameter Definitions
2. Gear functions
3. Crescent Functions
4. Miscellaneous functions

Parameter Definition

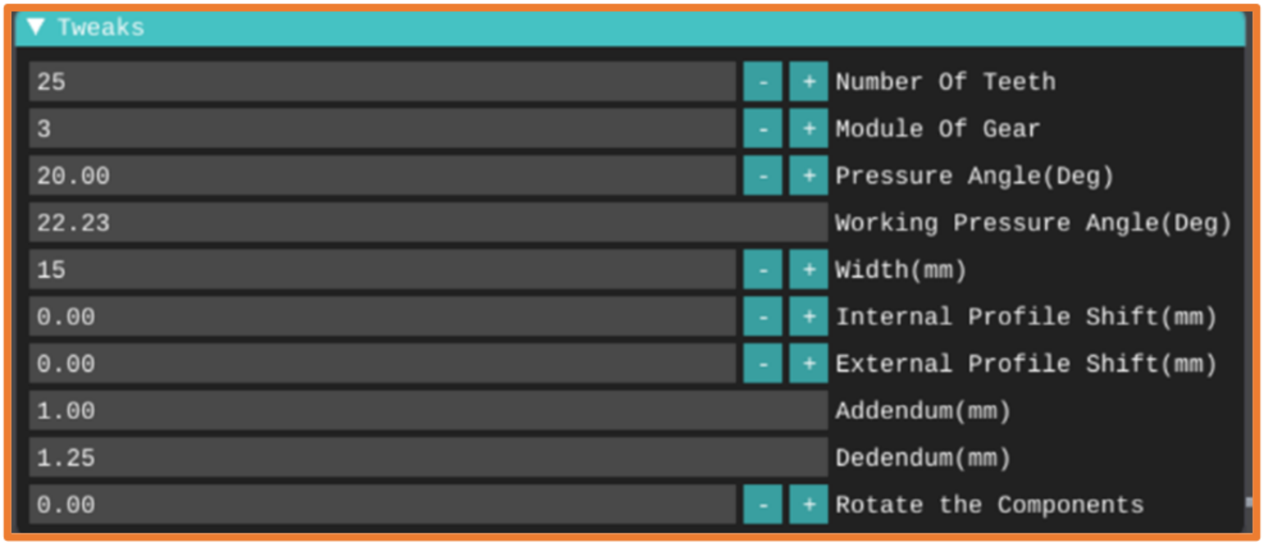
The section further sub-divides the parameters into two parts:

- Input Parameters: Defined by the user.
- Interdependent Parameters: Derived from formulas and input parameters.

Input Parameters

Defining the base parameters, is considered as the first and essential step of modelling. The Tweak box [Fig2] accepts the input values from users for developing custom models for simulation which provides real-time modification and analysis.

The default values in the Tweak box can be altered by modifying the below section [Fig 3] in the Lua script.




```
-------------------------Passing parameters to user interface-----------------------------------------------
z_n=ui_numberBox("Number Of Teeth",25);                          -- Nnumber of teeth, ideal: from 25
m=ui_numberBox("Module Of Gear",3);                              -- Gear Module, Ideal Input: 3
alpha_t=ui_scalarBox("Pressure Angle(Deg)",20,0.1);              -- Pressure angle
alpha_awn=ui_scalarBox("Working Pressure Angle(Deg)",22.23,0);   -- Working Pressure angle
width=ui_numberBox("Width(mm)",15);                              -- Width/Thickness of the gear
x_coef_int=ui_scalarBox("Internal Profile Shift(mm)",0.0,0.1);   -- Profile shift factor for internal gear
x_coef_ext=ui_scalarBox("External Profile Shift(mm)",0,0.1);     -- Profile shift factor for external gear
h_a_coef_p=ui_scalarBox("Addendum(mm)",1,0);                     -- Addendum height factor
h_f_coef_p=ui_scalarBox("Dedendum(mm)",1.25,0);                  -- Dedendum height factor
rotation= ui_scalarBox("Rotate the Components",0,4)*18;          -- Rotation
```

[Fig 3]: *Tweak Box code Modification*

Inter-Dependent Parameters

The input parameters aid in deriving the inter-dependent parameters that are significant for gear profile modelling. The derived parameters are utilized in the generation of gear-pair geometry as represented in the below figure [Fig 4].

```
-- Base formulae calculations required for the gear profile

-- Pitch Diameter:
    d_p = z_t * m_t                             -- Pitch Diamter
    r_p = d_p / 2                               -- Pitch radius


-- Base Diameter:
    d_b = d_p * math.cos(alpha_t_rad)           -- Base diameter of gear
    r_b = d_b / 2                               -- Base radius


-- Addendum and Dedendum
    h_a = m_t* h_acoef;                         -- Addendum

    h_f = m_t* h_dcoef                          -- Dedendum

-- Tip Diameter / Addendum Diameter
    d_a = m_t*z_t + 2*x_coef*m_t + 2*h_a;       -- Addendum diameter


    r_a = d_a / 2                               -- Addendum radius
    --h_a1 = m_t + c                            -- Addendum height

-- Root Diameter / Dedendum Diameter
    d_f = m_t*z_t+ 2*x_coef*m_t - 2*(h_f)       -- Root diamter
    r_f = d_f / 2                               -- Root_radius
    --h_f = m_t + c                             -- Root height
```

```
-- True Involute Diameter:

    d_TIF = math.sqrt(math.pow(d_p * math.sin(alpha_t_rad) -
            2 *(h_a - (m_t * x_coef) - h_f *(1 - math.sin(alpha_t_rad))), 2) +   d_b * d_b);   -- True involute diameter
    r_TIF = d_TIF / 2;                                                                          -- True form radius
```

```
--- Tooth thickness calculation on the form circle:

-- Tooth thickness on the pitch Circle

    S_0 = m_t*((math.pi/2)+ 2*x_coef* math.tan(alpha_t_rad));

-- Involute function

    inv_a = math.tan(alpha_t_rad) - alpha_t_rad;

-- Tooth thickness on the form circle

    alpha_f = math.acos((d_p*math.cos(alpha_t_rad))/d_TIF);             -- Involute angle along form circle
    inv_Ff  = math.tan(alpha_f) - alpha_f;                              -- Involute function
    s_f     = d_TIF*((S_0/d_p)+inv_a - inv_Ff)+(x_coef*math.tan(alpha_f));  -- Thickness of the gear teeth along form circle
    omega   = (s_f)/(0.5*d_TIF);                                        -- Angle swept along form circle for corresponding thickeness
```

[Fig 4]*: Inter-Dependent Parameters*

Note: The formulation of the inter-dependent parameters was implemented with reference to the document: [[1] Section 2.1]

Gear-pair implementations

The following sections provide the users with the scripting information on the external and internal gear modelling.

External Gear

The different functions and iterations for the external gear generation are discussed below. The functions defining the gear profile are:

1. Gear Profile: The main function to generate the complete gear-tooth profile.
2. Involute Angle: Generates the involute angle for the gear-tooth.
3. Tooth Involute: Generates the involute tooth curve.
4. Rotate Points: Represents the rotational matrices.
5. Mirror Points: Mirroring object profiles.
6. External Gear: Returns the table containing the XY points for involute profile.
7. Circle: Generates the circle of required radius.
8. Extrude: Extrusion of object along Z-axis.

Furthermore, the emitting and generation of the bore for external gear will be briefed.

Main Function: Gear Profile

```
------------------------Function for the calculation for the required Internal and External Gear Profiles----
function gear_profile(z, m, alpha_t, x_coef, h_a_coef, h_f_coef, width)
    local inv_xy = {}
-- Definition of the input parameters and calculation of the other base parameters for the involute profile.
    m_t = m;                                      -- Module of gear
    alpha_t_rad = alpha_t*math.pi/180;           -- Pressure angle
    z_t = z;                                     -- Number of teeth
    x_coef = x_coef;                             -- Profile shift co-efficient
    --x_coef_ext = 0;                            -- Profile shif coeff external gear
    c = 0.167 * m_t;                            -- Clearance of tooth
    h_acoef = h_a_coef;                          -- Addendum coefficient
    h_dcoef = h_f_coef;                          -- Dedendum coefficient
```

```
-- Base formulae calculations required for the gear profile

-- Pitch Diameter:
    d_p = z_t * m_t                        -- Pitch Diamter
    r_p = d_p / 2                          -- Pitch radius


-- Base Diameter:
    d_b = d_p * math.cos(alpha_t_rad)      -- Base diameter of gear
    r_b = d_b / 2                          -- Base radius


-- Addendum and Dedendum
    h_a = m_t* h_acoef;                    -- Addendum

    h_f = m_t* h_dcoef                     -- Dedendum

-- Tip Diameter / Addendum Diameter
    d_a = m_t*z_t + 2*x_coef*m_t + 2*h_a;  -- Addendum diameter


    r_a = d_a / 2                          -- Addendum radius
    --h_a1 = m_t + c                       -- Addendum height

-- Root Diameter / Dedendum Diameter
    d_f = m_t*z_t+ 2*x_coef*m_t - 2*(h_f)  -- Root diamter
    r_f = d_f / 2                          -- Root_radius
    --h_f = m_t + c                        -- Root height
```

```lua
-- True Involute Diameter:

    d_TIF = math.sqrt(math.pow(d_p * math.sin(alpha_t_rad) -
            2 *(h_a - (m_t * x_coef) - h_f *(1 - math.sin(alpha_t_rad))), 2) +   d_b * d_b);   -- True involute diameter
    r_TIF = d_TIF / 2;                                                                      -- True form radius

--- Tooth thickness calculation on the form circle:

-- Tooth thickness on the pitch Circle

    S_0 = m_t*((math.pi/2)+ 2*x_coef* math.tan(alpha_t_rad));

-- Involute function

    inv_a = math.tan(alpha_t_rad) - alpha_t_rad;

-- Tooth thickness on the form circle

    alpha_f  = math.acos((d_p*math.cos(alpha_t_rad))/d_TIF);            -- Involute angle along form circle
    inv_Ff   = math.tan(alpha_f) - alpha_f;                             -- Involute function
    s_f      = d_TIF*((S_0/d_p)+inv_a - inv_Ff)+(x_coef*math.tan(alpha_f));  -- Thickness of the gear teeth along form circle
    omega    = (s_f)/(0.5*d_TIF);                                       -- Angle swept along form circle for corresponding thickeness

-- Function for stariing and ending of involute between two radius

    tooth_ang = (((math.pi * m_t / 2) + 2 * m_t * x_coef * math.tan(alpha_t_rad)) /
                r_p + 2 * math.tan(alpha_t_rad) - 2 * alpha_t_rad)


    res = 30;                                                           -- Defining iterating points

-- Iterating the points for the involute points and iterating teeth around the circle

    for i = 1, z_t do
        th = 2 * math.pi                                                -- Iteration angle of the teeth around the gear diameter
        th1 = involute_angle(r_b,r_b);                                  -- Start angle to define the start of the involute curve
        th2 = involute_angle(r_b, r_a);                                 -- End asngle to determine the end of the involute curve


-- Defining iterations for the involute teeth profile

      for j = 1, res do

        inv_xy[#inv_xy + 1] = rotate_points(th*i/z_t, tooth_involute(r_b, (th1 +(th2-th1) * j / res)))          -- The one side of the involute points are obtained

      end
      for j = res, 1, -1 do

        inv_xy[#inv_xy + 1] =
        rotate_points(th*i/z_t, rotate_points(omega, tooth_mirror(tooth_involute(r_b, (th1 + (th2 - th1)* j / res)))))  -- The second side of the involute points are obtained

      end
```

[Fig 5]: Main function for Generating Gear Profile

a) Iterations

This section deals with the iterations to generate the involute gear tooth profile for the gear object generation.

Involute Tooth Profile Iterations: Iterative loops defining involute tooth curves. [Fig 6]

- Inner Loop – Iteration of 'j' from 1 to n providing single points for XY points of the involute tooth curve.
- Outer Loop – Iteration of 'th' from 0 to 360 degrees for generating 'z_t' number of teeth around the gear.

```lua
-- Iterating the points for the involute points and iterating teeth around the circle

    for i = 1, z_t do
        th = 2 * math.pi                                            -- Iteration angle of the teeth around the gear diameter
        th1 = involute_angle(r_b,r_b);                              -- Start angle to define the start of the involute curve
        th2 = involute_angle(r_b, r_a);                             -- End asngle to determine the end of the involute curve

-- Defining iterations for the involute teeth profile

        for j = 1, res do

            inv_xy[#inv_xy + 1] = rotate_points(th*i/z_t, tooth_involute(r_b, (th1 +(th2-th1) * j / res)))          -- The one side of the involute points are obtained

        end
        for j = res, 1, -1 do

            inv_xy[#inv_xy + 1] =
            rotate_points(th*i/z_t, rotate_points(omega, tooth_mirror(tooth_involute(r_b, (th1 + (th2 - th1)* j / res)))))  -- The second side of the involute points are obtained

        end

    end
```

[Fig 6]: Involute Tooth Profile Iterations

b) Sub Functions

i. Function Involute Angle - This function is intended for calculating the involute angle (Inv_alpha). The radius of the Addendum circle and base circle are passed as an input for obtaining corresponding involute angle. [Fig 7]

```lua
-----------Function defining angle between corresponding points-------

function involute_angle(r_p1, r_p2)
    return (math.sqrt((r_p2 * r_p2 - r_p1 * r_p1) / (r_p1 * r_p1)))
end
```

[Fig 7]: Function Involute Angle

ii. Function Involute Profile – The base radius (start of the involute profile) and the corresponding pressure angle is passed to this function to obtain the required xy points of the involute tooth curve. [Fig 8]

```lua
----------Function definition for Parametric equations for the involute profile points--------------------------

function tooth_involute(base_radius, inv_alpha)                          -- inv_alpha = Involute angle
    return v(base_radius*(math.sin(inv_alpha) - inv_alpha*math.cos(inv_alpha)),
            base_radius*(math.cos(inv_alpha) + inv_alpha* math.sin(inv_alpha)))
end
```

[Fig 8]: Function Involute Profile

iii. Function Mirror – This function is intended for inverting profile points with respect to Y-Axis. This function takes in xy points of a single side profile and provides with points for the mirrored profile. [Fig 9]

```
-----Function defining for mirroring or inverting profile points-
-- coord -Co-ordinates
function tooth_mirror(coord)
    return v(-coord.x, coord.y)
end
```

[Fig 9]: Function Mirror

iv. Function Rotational Matrix – The rotational matrix is represented in this function. The rotation angle and the object coordinates are passed as an input giving out the rotated coordinates for the object. [Fig 10]

```
----------Function defining the rotational matrix--------------------------------------------

function rotate_points(angle, coord)                                     -- angle = angle, coord = co-ordinates
    return v(math.cos(angle) * coord.x + math.sin(angle) * coord.y, math.cos(angle) * coord.y - math.sin(angle) * coord.x)
end
```

[Fig 10]: Function Rotational Matrix

v. External Gear - The function 'f(x)= gear(x)' in [fig] takes in the user defined input values which in turn brings about the required 'XY' points for creating the external

```
---------Function for External Gear Formation----------------------------------------------
function external_gear()

  externalGear = gear({z=z_n-8;m=m;alpha_t=alpha_t;x_coef=x_coef_ext;h_a_coef=h_a_coef_p;h_f_coef=h_f_coef_p;width=width})

end
external_gear()
```

gear profile. [Fig 11]

[Fig 11]: Function Calling - External Gear

vi. Circle: The function accepts the radius 'r' as input and gives XY coordinates for generating a circular profile.[Fig 12]

```
----Function defining the parametrical equation for the circle-----
-- r -radius of circle
function circle(r)
  local x, y = 0, 0
  local XY={}
  for i = 1, 360 do
    local angle = i * math.pi / 180
    XY[i] = v(x + r * math.cos( angle ), y + r * math.sin( angle ))

  end
  return XY
end
```

[Fig 12]: Function Circle

vii. Extrude: Determines the extrusion of bore for the external gear.

External gear formation

The external gear is emitted by cumulatively utilizing the functions as stated in the figure below. The object is translated to the meshing distance as elaborated in the 'Meshing' section.[Fig 13]

```
---------------Emitting the External Gear---------------------------------------------------

bore_externalGear= extrude(circle(addOn_distance), 0, v(0,0,width), v(1,1,1), 20)              -- Bore formation of external gear
emit(rotate(rotation,Z)*translate(0,meshDistance,0)*difference(externalGear,bore_externalGear),100) -- External Gear Formation
```

[Fig 13]: Extrude Function

Note: The value of teeth difference between Internal and External Gear has been considered to be 8. This has been determined in accordance with different conditional interferences in the internal and external gear pair [1].

```
z_1 = z_n-8;                    -- Number of teeth External Gear
```

Internal Gear

The same set of functions described for the modelling of the external gear are reused for the generation of the internal gear.

Internal gear formation

The external gear is emitted as stated above and in turn subtracted from the circle which results in the required Internal Gear formation.[Fig 14]

```
--------------Function for Internal Gear Formation-----------------------------------------------
function internal_gear()

  internalGear = gear({z=z_n;m=m;alpha_t=alpha_t;x_coef=x_coef_int;h_a_coef=h_a_coef_p;h_f_coef=h_f_coef_p;width=width})

end
internal_gear()
```

```
-------------------------Emitting the Internal Gear---------------------------------------------------
radius=((z_n)*m/2);                                                          -- Radius for outer cylinder
outer_circle= extrude(circle(radius+addOn_distance), 0, v(0,0,width), v(1,1,1), 20)    -- Outer circle for internal gear formation

emit(difference(outer_circle,internalGear),128);                             -- Internal gear formation
```

[Fig 14]: Function emit – Internal Gear

Meshing

The center distance(a_x) calculation helps to derive meshing distance essential for the gear pairs. The working pressure angle (alpha_awn) is one of the required input parameters for meshing and is obtained from the interpolation of the Involute function table [2].[Fig 15]

```
--Calculation for the centre distance between gears using the profile shift coefficients--

-- Involute function -Operating pressure angle
x_coef_int = x_coef_int;    -- Profile shift coefficient internal gear
x_coef_ext = x_coef_ext;    -- Profile shift coefficient External gear
z_2 = z_n;                  -- Number of teeth Internal Gear
z_1 = z_n-8;                -- Number of teeth External Gear

alpha_rad = alpha_t*math.pi/180                                              -- Preassure angle
inv_a = math.tan(alpha_rad) - alpha_rad;                                     -- Involute function
inv_aw = ((2*math.tan(alpha_rad) * (x_coef_int - x_coef_ext))/(z_2 - z_1)) + inv_a;   -- Involute function working pressure angle

-- Working preassure angle
alpha_aw = alpha_awn*math.pi/180;                                            -- Working pressure angle

-- Centre distance incremental factor
y_c = ((z_2 - z_1) * (math.cos(alpha_rad) - math.cos(alpha_aw)))/ (2* math.cos(alpha_aw));

-- Center distance between Gears
a_x = (((z_2 - z_1)/2)+y_c) * m;                                              -- Center distance

meshDistance = a_x;                                                          -- Center distance is assagined for cresant calculation
```

[Fig 15]: Meshing – Centre Distance Between Gears

Crescent Filler Formation

The crescent is obtained by creating a difference of two cylinders with addendum radius and clearance of internal and external gear respectively. The resultant cylinder formed by the external gear is translated to the length equivalent to the meshing distance with respect to the Y-axis. The sharp edges of the so formed crescent filler are flattened by subtracting it from the cube generated as shown below. [Fig 16]

```
--------------------------------------------------------Crescent Filler---------------------------------------------------------

crescent_innerRadius=r_a;                                                                                    -- Radius of the external gear for the cylinder
crescent_clearanceIn=c;                                                                                      -- Clearance of the external gear for the cylinder

crescent_translation=meshDistance;                                                                           -- Translation for crescent formation

crescent_cylBottom=translate(0,crescent_translation,0)*ccylinder(crescent_innerRadius+crescent_clearanceIn,width);       --Inner circle for internal gear formation

crescent_cubeRight=translate(crescent_innerRadius+crescent_clearanceOut,meshDistance+m*2,0)*
                   cube(crescent_rootRadius+crescent_clearanceOut, crescent_rootRadius+crescent_clearanceOut,width+0.1)     -- Cube to remove sharp edges of the crescent
crescent_cubeLeft=translate(-(crescent_innerRadius+crescent_clearanceOut),meshDistance+m*2,0)*
                   cube(crescent_rootRadius+crescent_clearanceOut, crescent_rootRadius+crescent_clearanceOut,width+0.1)     -- Cube to remove sharp edges of the crescent
crescent_Main=translate(0,0,width/2+0.1)*difference(crescent_cylTop,crescent_cylBottom)                     -- Crescent Filler Formation with sharp edges
crescent_Full=rotate(rotation,Z)*intersection(difference(crescent_Main,crescent_cubeRight),
                   difference(crescent_Main,crescent_cubeLeft))                                              -- Crescent Filler Formation without sharp edges
```

[Fig 16]: Crescent Filler Formation

Slicing and G-Code

The figure below demonstrates the steps for slicing (to generate '.stl' files) and G-Code generation for the objects emitted in IceSl Slicer. [Fig 17]

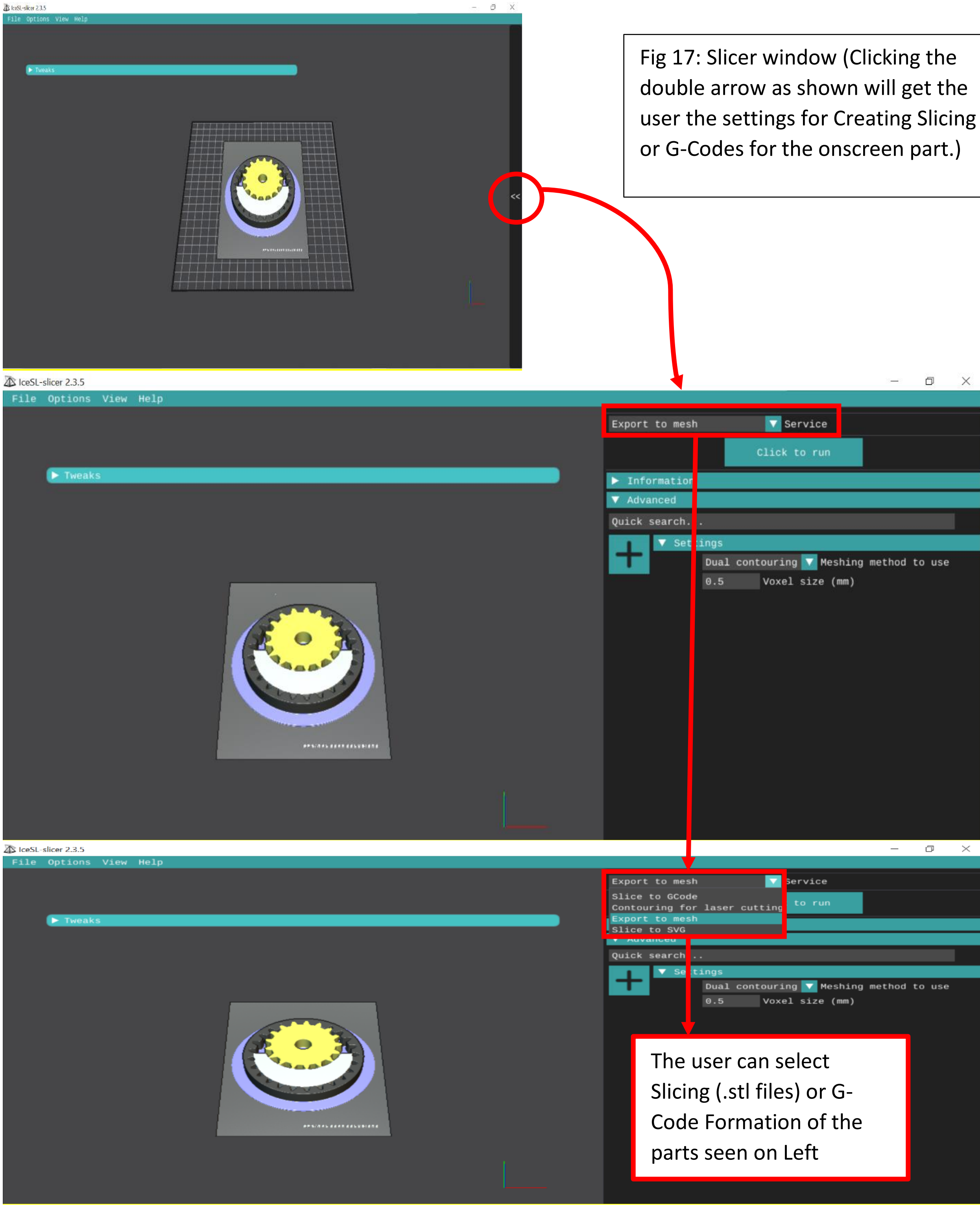


Fig 17: Slicer window (Clicking the double arrow as shown will get the user the settings for Creating Slicing or G-Codes for the onscreen part.)

Components List

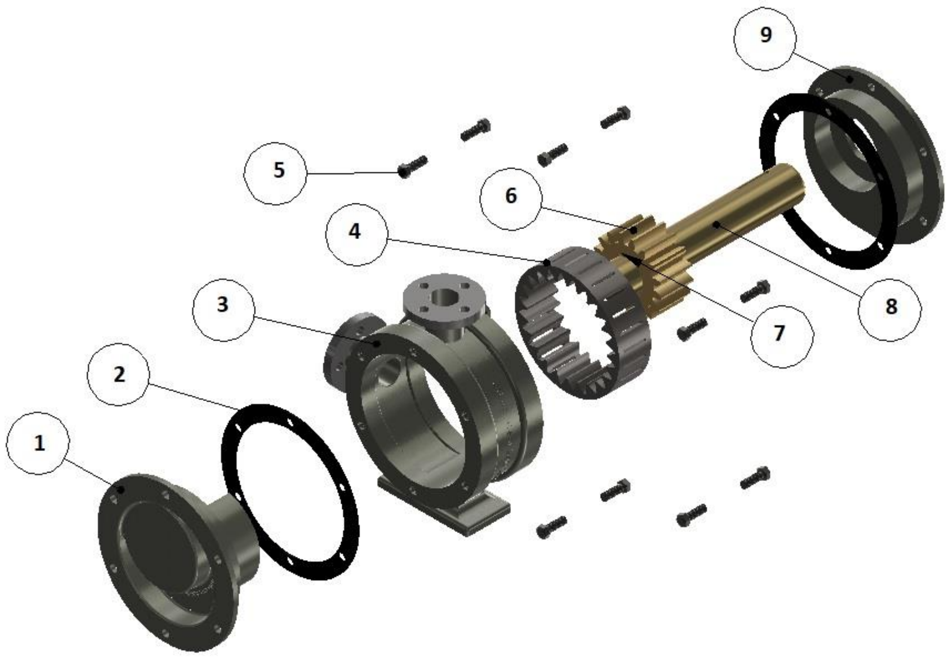


Fig 18: Internal Gear Pump Exploded View

No.	Name Of Component	Quantity
1	Flange Head with Crescent	1
2	O-Ring/ Rubber Seal	2
3	Casing	1
4	Internal Gear	1
5	M6 Bolt	12
6	External Gear	1
7	Key	1
8	Shaft	1
9	Tail Flange	1

Assembly Guidance

The Internal Gear, External Gear and Crescent Filler can be 3D printed as per the user defined parameters with the instructions stated above. This allows the users to generate the part numbers stated in the above table: 1: Crescent without Flange, 4: Internal Gear and 6: External Gear.

To assemble the Internal Gear Pump users also require the remaining parts as stated below:

- Part number 1 (Only Flange head),
- Part number 2 (O-Rings/Rubber Seal Rings into 2 quantities),
- Part number 3 (Casing with Inlet and Outlet ports),
- Part number 5 (Bolts) fitting into the casing threading provided,
- Part number 7,8 (Shaft with key) fitting the bore of External Gear,
- Part number 9 (Tail flange).

The user needs to drill the holes in the 3D printed Internal Gear as per the Inlet and Outlet port discharge which can be again calculated using the technical documentation.[1]

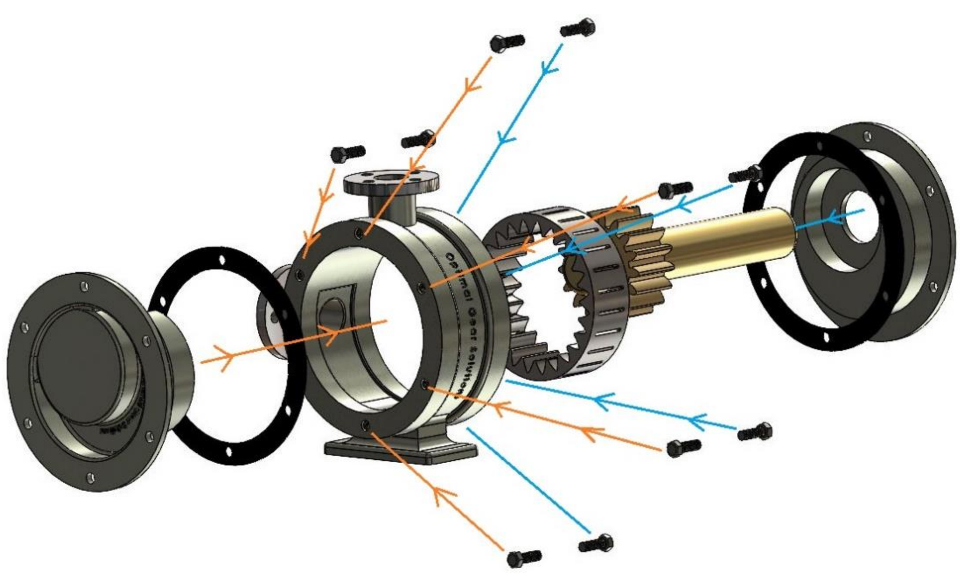


Fig 19: Internal Gear Pump Assembly

- Accumulating all the parts of the pump, the user can start assembling the parts in the direction as shown in the above picture. The orange lines present the fitting of parts on one side of
- Further ahead, user should fit the Internal Gear inside the cylinder casing with the help of lubrication followed by the external gear which is pre-fitted with the shaft and then at last the crescent should slide in between the remaining gaps of Internal Gear and External Gear.
- When printed using the Lua Script provided by Optimal Gear Solutions, the 3D printed parts should mesh properly and should have no problems mating in a perfect way to provide a seamless output.
- Then the upper/front side casing of the pump should also be sealed properly for starting the working of the Internal Gear Pump. The user should follow the above steps to assemble the parts and should take care of the limits. The user should also take care of the values provided for printing the model and should refer to the necessary documents if required.

References

[1] Radhakrishnan Nair, A., Phadnis, A., Thirumurugan, D., & Abraham James, J. (2021). Internal gear pump. Optimal Gear Solutions.

[2] KOHARA, G.I.C., 2007. Numerical Formulas and Tables Version: July 2018. – [Online; Stand 2 July, 2012].