REVISTA ELECTRÓNICA SOBRE TECNOLOGÍA, EDUCACIÓN Y SOCIEDAD
ISSN: 2448 - 6493
CTES

Artículos científicos

Diseño de cuerpos sólidos con FreeCAD y su uso en dinámica de fluidos computacional

Solid bodies design with FreeCAD and its use in computational fluid dynamics

Mario A. Aguirre-López
Universidad Autónoma de Chiapas, México
marioal1906@gmail.com
https://orcid.org/0000-0002-5191-3462

José Roberto Cantú-González
Universidad Autónoma de Coahuila, México
roberto.cantu@uadec.edu.mx
https://orcid.org/0000-0001-5616-2947

Filiberto Hueyotl-Zahuantitla
Cátedra CONACyT-FCFM UNACH, México
fhueyotl@conacyt.mx
https://orcid.org/0000-0002-5527-7141

Resumen

El diseño de estructuras en las simulaciones de aerodinámica computacional es una etapa muy importante en la que se requiere de programas especializados que suelen ser bastante caros. En este contexto, las aplicaciones libres y/o de código abierto como FreeCAD son cada vez más utilizadas. En este trabajo se muestran características destacables de FreeCAD que permiten modelar morfologías con diferentes grados de complejidad. Se muestra el procedimiento para diseñar múltiples objetos mediante líneas de código en Python, en lugar de utilizar la interfaz gráfica, reduciendo el tiempo de cómputo de manera significativa. Se presentan dos ejemplos del uso de las estructuras diseñadas en simulaciones de dinámica de fluidos computacional con OpenFOAM y su visualización con ParaView, ambos también de libre acceso. En esta contribución se resalta el potencial de la sinergia de recursos de libre acceso y/o de código abierto, en este caso FreeCAD+OpenFOAM+ParaView+Python.

Palabras clave: FreeCAD, dinámica de fluidos computacional, Open FOAM, modelación por código, ParaView.

REVISTA ELECTRÓNICA SOBRE TECNOLOGÍA, EDUCACIÓN Y SOCIEDAD
ISSN: 2448 - 6493
CTES

Abstract

The design of structures in computational aerodynamics simulations is a very important stage, in which specialized software are required, usually quite expensive. In this context, free access and/or open-source applications such as FreeCAD are increasingly used. This work shows outstanding features of FreeCAD that allow modeling morphologies with different degrees of complexity. The procedure for designing multiple objects is shown through Python scripting, instead of using the graphical interface, reducing the computing time significantly. This is shown by two examples of the use of the designed structures in computational fluid dynamics simulations with OpenFOAM and its visualization with ParaView, both also freely accessible. This contribution highlights the potential of the synergy of free-access and/or open-source resources, in this case FreeCAD+OpenFOAM+ParaView+Python..

Keywords: FreeCAD, computational fluids dynamics (CFD), Open FOAM, code modeling, ParaView.

Fecha Recepción: Junio 2021 **Fecha Aceptación:** Diciembre 2021

Introducción

FreeCAD es un modelador paramétrico de código abierto con arquitectura modular para diseñar objetos de la vida real de cualquier tamaño. Es ampliamente utilizado de manera profesional por arquitectos, ingenieros mecánicos, diseñadores de interiores, programadores, así como por estudiantes, docentes e investigadores de diversas áreas de la ciencia. Es un recurso disponible para descarga en compatibilidad a los sistemas operativos Mac, Windows y Linux (The FreeCAD Team, 2019), donde se encuentran de manera explícita las instrucciones para su instalación.

Entre las principales funciones de FreeCAD se pueden mencionar el dibujo 2D y modelado 3D, se pueden crear objetos 3D a partir de modelos 2D y viceversa, cuenta con herramientas CAD (*Computer Aided Design*, diseño asistido por computadora), herramientas modernas de análisis de elemento finito y herramientas de importación y exportación de datos en los formatos más usuales. Sus herramientas tienen características similares a otros softwares como CATIA o SolidWorks, pero son de libre acceso.

Una de las características importantes de FreeCAD es que se pueden integrar módulos externos con su código fuente, estos módulos son llamados *workbenches* y amplían de manera casi ilimitada sus posibilidades. Una lista amplia de *workbenches* y su descripción se puede encontrar en los recursos que comparte la organización FreeCAD (The FreeCAD team, 2022). A manera de muestra podemos mencionar dos problemas de mecánica de

fluidos: el primero es DesignSPHysics, que es una interfaz gráfica para DualSPHysics, un solver de dinámica de fluidos que usa el formalismo SPH (Smoothed particle hydrodynamics); el segundo es Cfd, el cual provee una interfaz gráfica de OpenFOAM para realizar simulaciones de dinámica de fluidos computacional (CFD, Computational Fluid Dynamics).

OpenFOAM es un software libre y de código abierto con gran reputación en CFD. Es ampliamente utilizado en varias áreas de ingeniería y ciencias para simular flujos tan complejos como aquellos que involucran reacciones químicas, turbulencia, transferencia de calor, fenómenos acústicos, de mecánica de sólidos y electromagnéticos. OpenFOAM está disponible para sistemas operativos como Mac, Windows y Linux, su descarga e instrucciones de instalación son de libre acceso (OpenCFD Ltd, 2022).

Este documento hace énfasis en la importancia del software libre como opción para modelar-resolver problemas que impliquen el uso de un modelador de objetos como FreeCAD y un solver de CFD como OpenFOAM. En un acercamiento preliminar a la aplicación de estos softwares encontramos los ejemplos puntuales de: modelado de instrumentos musicales y su análisis de elemento finito (Torres et al., 2018), estudios de la rotación de hélices en un fluido estratificado (Jacobs, 2020), flujos supersónicos a través de una boquilla (Welnitzová & Molnár, 2016) y comparación de modelos de turbulencia en flujos en un túnel de viento (Bibhab Kumar Lodh et al., 2017).

Ahora bien, en la utilización de FreeCAD para propósitos de diseño y modelación de los cuerpos y lo relacionado a simulaciones asociadas a CFD, se identifican interesantes aplicaciones en diversas áreas de la ingeniería, una clasificación donde destacan ciertas de sus ramas se presenta a continuación:

- Ingeniería química: FreeCAD ha sido empleado en la optimización de geometrías de espaciadores de ósmosis retardada por presión (Benjamin et al., 2022); también en el diseño de la línea central de la dirección de la tubería en un estudio de transferencia de masa y calor, donde se analiza el efecto de golpe de ariete inducido por condensación de contacto directo en tuberías de vapor (Pham & Choi, 2021); a su vez, en un estudio comparativo exhaustivo en 3D de campo completo de velocimetría por resonancia magnética (MRV) y CFD del flujo de gas a través de estructuras monolíticas catalíticas regulares, FreeCAD es utilizado para generar un objeto uniforme a partir de combinar el monolito con el cuerpo del reactor (Mirdrikvand et al., 2021); finalmente, en procesos de hermanamiento digital para reactores de tanque agitado/operaciones de unidades de separación, se

utiliza este software para la geometría de ciertos experimentos (Oblak et al., 2020).

- Ingeniería civil: en un estudio para mejorar el desempeño de un vertedero tipo laberinto rectangular, FreeCAD es utilizado para modelar la geometría de los modelos probados (Said & Ouamane, 2021); mientras que, en una simulación de amortiguación de un cilindro hidráulico lineal, este software fue utilizado para realizar cálculos de momentos de inercia y la determinación de centro de masa (Algar et al., 2021).
- Ingeniería de procesos y evaluación de riesgos: en un estudio de una explosión ventilada de polvo de almidón de maíz, FreeCAD es utilizado para desarrollar la geometría CAD del contendedor (Huang et al., 2022).
- Ingeniería energética: aquí FreeCAD es utilizado para crear el modelo y llevar a cabo cálculos estructurales en el diseño de convertidores de energía de olas (Giannini et al., 2022); adicionalmente ha sido utilizado para generar el modelo y el mallado de la geometría del sistema en un estudio de consumo de energía por capacidad de refrigeración debida a la introducción de un flujo de aire dentro de una habitación (Ho et al., 2022).
- Ingeniería aeronáutica: en esta rama de la ingenieria FreeCAD se hace presente en el diseño de perfiles aerodinámicos, tanto en un análisis detallado del perfil NACA 0012 (Sadadiwala, 2021), como en un estudio de formación de ondas de choque oblicuas (Hasan & Nik Mohd, 2021). Un ejemplo adicional se presenta en el desarrollo de modelos alternos de redes neuronales para la optimización de formas aerodinámicas (Zhang et al., 2021).
- Ingeniería de gestión educativa: aquí FreeCAD en conjunto con CFD y Realidad virtual (RV) / Realidad aumentada (RA) es utilizado como modelador 3D para desarrollar herramientas educativas para la ingeniería y los negocios (Solmaz & Van Gerven, 2021).

Mencionada la diversidad de las aplicaciones antes expuestas, conviene agregar que este trabajo describe las características más importantes de FreeCAD en su aplicación como software modelador de objetos para su uso en aerodinámica computacional. A diferencia de la mayoría de los trabajos de aerodinámica computacional, donde el enfoque se acentúa en las pruebas aerodinámicas o en el software CFD, este trabajo se centra en la creación de objetos en FreeCAD mediante un script de Python, enfocándose principalmente en la modelación de cuerpos con morfología sencilla para una fácil comprensión de usuarios principiantes en FreeCAD.

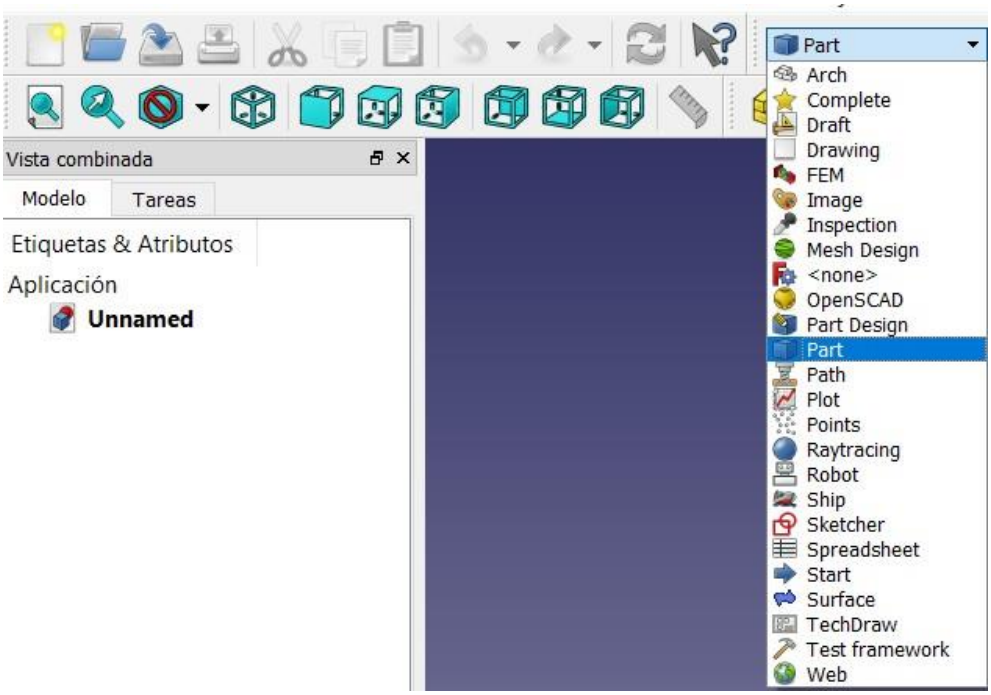
Finalmente es pertinente exponer que basado en el aporte de otras herramientas educacionales interactivas (Ramirez-Cortes et al., 2012), (Yadav & Patel, 2018), (Cantú-González et al., 2020) y (Rhee et al., 2021), y coincidiendo en su enfoque didáctico, el presente trabajo tiene el propósito de proveer una guía metodológica en el uso del software FreeCAD, la misma podrá ser de gran beneficio para los usuarios de la academia y las organizaciones que valoran el software libre.

La estructura siguiente a la presente sección de introducción está divida de la siguiente manera: como primer punto se muestra la sección de "Características del FreeCAD", la cual detalla las características y herramientas a utilizar en las pruebas numéricas, en seguida se presentan "Método" que describe los pasos a seguir en el uso de las herramientas; posteriormente "Aplicación Metodológica" ejemplifica el método con pruebas numéricas para dos casos de estudio; enseguida tenemos la sección de "Resultados" que proporciona un recuento general de los ejemplos presentados y sus salidas, luego "Discusión" resume una interpretación de los resultados y confronta los objetivos iniciales, por último, se presentan "Conclusión" y "Futuras Líneas de Investigación".

Características de FreeCAD

La parametrización de objetos es la principal característica con la que FreeCAD se identifica: *"Your own 3D parametric modeler"* (The FreeCAD Team, 2019), que puede traducirse como "Tu propio modelador paramétrico 3D". Esta característica permite establecer un diseño base y modificarlo fácilmente al ir hacia atrás en el historial del modelo o diseño comenzado y cambiar sus parámetros, obteniendo diseños semejantes en función de sus parámetros, sin la necesidad de crear uno nuevo desde cero. Para ello, FreeCAD cuenta con el escenario Part, ver Figura 1.

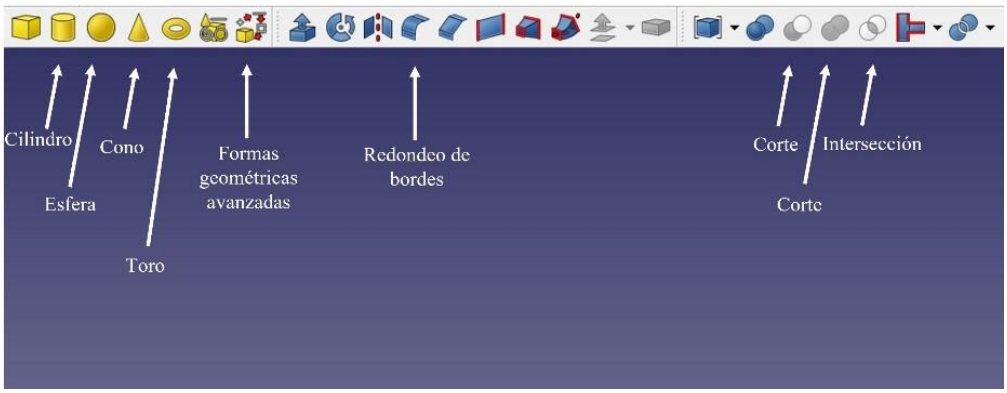
Figura 1. Escenario Part en la interfaz de usuario.



Fuente: (The FreeCAD Team, 2019)

Este escenario consiste en un panel de funciones con las que se pueden diseñar formas primitivas geométricas en 2D y 3D, así como formas simples en 3D como esferas, cilindros, prismas, conos, toros, entre otros. De igual manera, se tienen funciones para crear formas paramétricas avanzadas, redondear los bordes de un objeto o bien, unir, cortar o intersectar dos o más formas, ver Figura 2.

Figura 2. Panel de funciones disponibles dentro del escenario Part

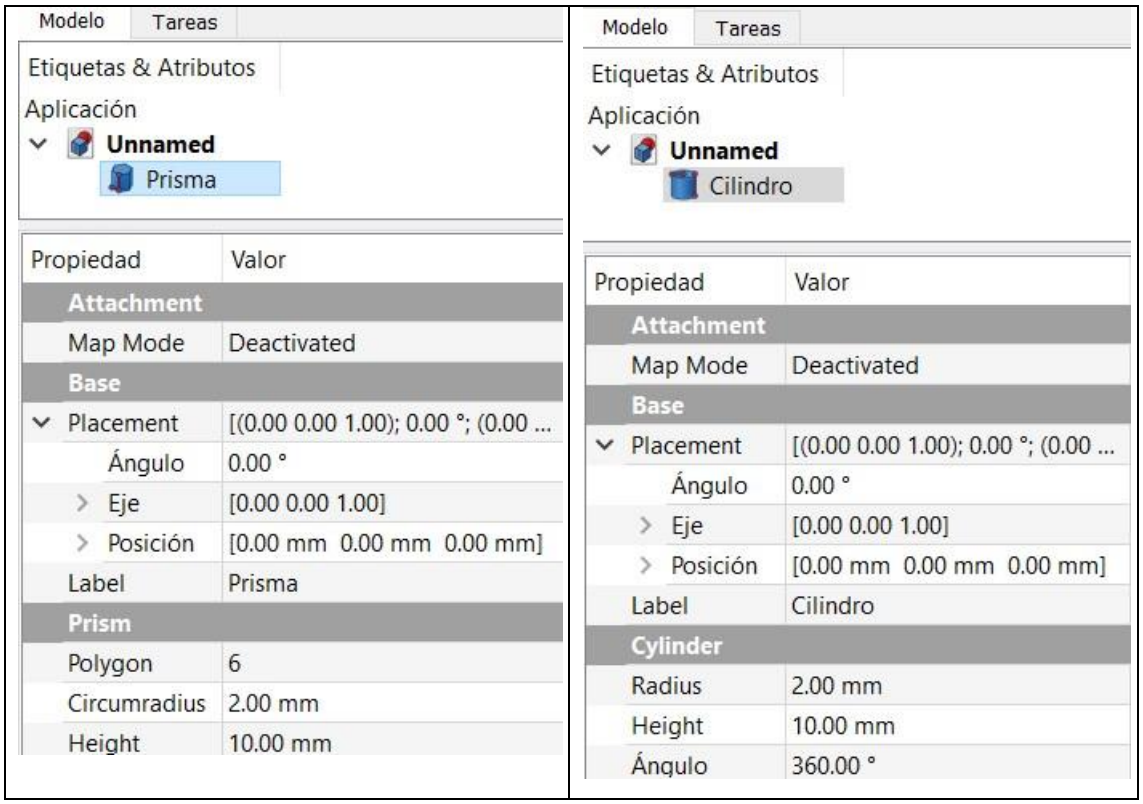


Fuente: Adaptación a partir de (The FreeCAD Team, 2019)

La modificación de parámetros está programada de acuerdo con el tipo de forma a añadir. A manera ilustrativa, la función *Primitivas geométricas / Prisma* tiene como parámetros: número de lados de la base del polígono, circunradio en el que estará inscrita la base y altura del prisma, ver Figura 3 (a); a su vez, la función Cilindro tiene como parámetros:

altura, radio y ángulo del cilindro (p.ej. 360º equivale a un cilindro completo), ver Figura 3 (b).

Figura 3. Menú de parámetros de las funciones
(3a: izquierda) Prisma y (3b: derecha) Cilindro



Fuente: (The FreeCAD Team, 2019)

Por otro lado, hay parámetros generales que comparten todas las funciones para creación de formas, los cuales se refieren a la ubicación del objeto: posición, eje y ángulo.

Hasta este punto, las funciones y herramientas mencionadas han sido mostradas para su aplicación desde la interfaz de usuario, sin embargo, otra característica de FreeCAD muy útil para la modelación paramétrica es la programación por código. Esto es posible mediante la herramienta *Macro*, la cual permite grabar todas las acciones realizadas durante una sesión de FreeCAD para una posterior ejecución, incluyendo la apertura y cierre de un documento, ver "Ejemplo 1" mostrado en la Figura 4.

Figura 4. Ejemplo1

```
1 # -*- coding: utf-8 -*-
2
3 # Macro Begin: C:\Users\mario\AppData\Roaming\FreeCAD\Macro\assadas.FCMacro ++++++++++++
4 import FreeCAD
5 import Part
6 import Mesh
7
8 #exec(open('C:/Program Files/FreeCAD 0.18/data/Mod/Start/StartPage/LoadNew.py').read())
9 #App.setActiveDocument("Ejemplo1")
10 #App.ActiveDocument=App.getDocument("Ejemplo1")
11 #Gui.ActiveDocument=Gui.getDocument("Ejemplo1")
12 App.ActiveDocument.addObject("Part::Cylinder","Cylinder")
13 App.ActiveDocument.ActiveObject.Label = "Cilindro"
14 App.ActiveDocument.recompute()
15 #Gui.SendMsgToActiveView("ViewFit")
16 FreeCAD.getDocument("Ejemplo1").getObject("Cylinder").Height = '1 mm'
17 FreeCAD.getDocument("Ejemplo1").getObject("Cylinder").Angle = '40 deg'
18 FreeCAD.getDocument("Ejemplo1").getObject("Cylinder").Angle = '360 deg'
19
20 __objs__=[]
21 __objs__.append(FreeCAD.getDocument("Unnamed1").getObject("Cylinder"))
22 Mesh.export(__objs__,u"C:/Users/mario/Documents/ejemplo1.stl")
23
24 del __objs__
25 App.closeDocument("Ejemplo1")
26 #App.setActiveDocument("")
27 #App.ActiveDocument=None
28 #Gui.ActiveDocument=None
29 # Macro End: C:\Users\mario\AppData\Roaming\FreeCAD\Macro\assadas.FCMacro ++++++++++++++
```

Fuente: Elaboración propia

La Figura 4 muestra un ejemplo de un *Macro* realizado en la interfaz gráfica de FreeCAD ejecutando el documento “Ejemplo1”, el cual se describe a continuación:

→ Creación de un cilindro (líneas 12-14)

→ Modificación de la altura del cilindro (línea 16)

→ Modificación del ángulo de la base del cilindro (línea 17-18)

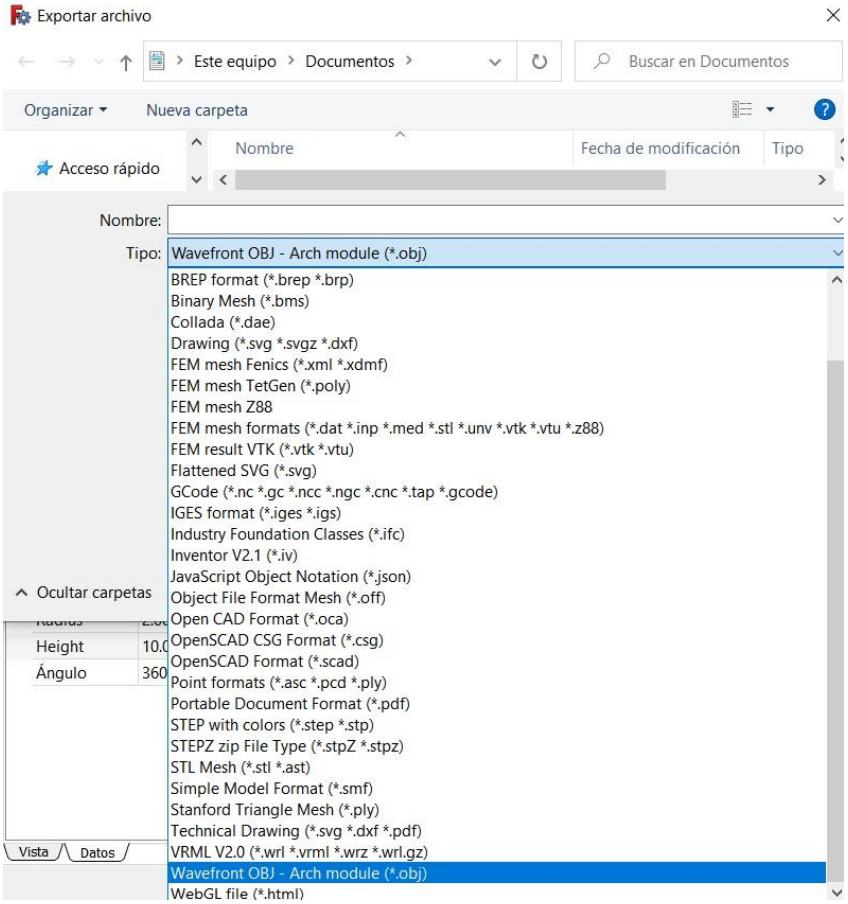
→ Exportación del cilindro en formato STL mesh (líneas 20-22)

→ Cierre del documento (líneas 24-25).

El usuario tiene acceso al código del *Macro* gracias a la característica de código abierto de FreeCAD. Además, el código está programado en Python, un lenguaje de programación que también está disponible para descarga (Python Software Foundation, 2019), esta ventaja nos permite copiar las líneas de código requeridas para ejecutar desde la línea de comandos de una terminal o bien desde un *script*.

Por último, como se puede ver en el menú de la Figura 5, FreeCAD cuenta con más de 30 formatos diferentes de exportación de las formas generadas, entre los que destacan más de cinco tipos de malla de elemento finito (*Finite Element Mesh* (FEM)), *STereoLithography* (STL mesh), *Wavefront OBJ*, y extensiones CAD.

Figura 5. Tipos de formato de exportación de un mallado



Fuente: Elaboración propia

Método

Una vez conocidas las principales características de FreeCAD, aquí se describe el procedimiento necesario para programar un mallado tipo STL de una forma 3D desde un *script* de Python, y su implementación en OpenFOAM para realizar una simulación CFD. A continuación, se describe el procedimiento:

Paso 0. Identificar las funciones en FreeCAD a utilizar para diseñar la forma deseada

Paso 1. Crear el *script* en Python

-> 1.1. Apertura del documento

-> 1.2. Diseño de la forma

-> 1.3. Exportación de la forma

Paso 2. Ejecutar el *script* desde la terminal

Paso 3. Implementar el objeto generado a OpenFOAM

-> 3.1. Importación del mallado STL mediante la utilería SurfaceFeatureExtractDict

-> 3.2. Implementación en la utilería SnappyHexMeshDict

-> 3.3. Simulación CFD

En la estructura anterior, el Paso 0 es opcional debido a que puede omitirse si se tiene el

conocimiento previo de las líneas de código (en Python) necesarias para diseñar la forma deseada; en caso contrario, la identificación se hace por medio de la función *Macro*, como fue descrito en la sección anterior.

La escritura del *script* generador de la forma abarca el Paso 1, el cual está dividido en tres subpasos; a manera ilustrativa, el código para generar un archivo "*.stl" se muestra en la Figura 4, recordando que se debe añadir el comando #!/usr/bin/env python en la primera línea del *script* para activar el ambiente de Python, y abrir un nuevo documento de FreeCAD con el comando App.newDocument("nombreArchivo"), en caso de que no se tenga abierto el documento. El subpaso a modificar para crear objetos diferentes es el 1.2. Para realizar el Paso 2, se debe teclear el nombre del archivo a ejecutar con la ruta correcta (p. ej. /home/usuario/prueba1/nombre_script.py).

La implementación en OpenFOAM del objeto STL para las simulaciones CFD (Paso 3) involucra dos subpasos, 3.1 y 3.2. Ambos subpasos requieren tener instalado el software OpenFOAM, los detalles técnicos asociados exceden al propósito de este trabajo, pero son disponibles para beneficio del usuario (OpenCFD Ltd, 2022).

Aplicación Metodológica

Se presentan dos casos de estudio para ilustrar los resultados numéricos que se obtienen al implementar el método mostrado en la sección anterior. Dichos casos de estudio son: una caja cuadrada (5 mm x 5 mm) con una altura de (1.5 mm), y la misma caja pero con un corte circular de 1mm de radio en una de sus esquinas.

Las Figuras 6 (a) y 6 (b) muestran los *scripts* de Python para los dos casos de estudio junto con su objeto "*.stl" correspondiente. De forma que, para el primer caso de la caja sin alteraciones, se presenta la activación del ambiente Python (línea 1), librerías (líneas 4-10) y parámetros (líneas13-15) a utilizar, apertura del documento FreeCAD (línea 18), creación (líneas 21-23), modificación (líneas 26-30) y exportación del objeto (líneas 33-35), y cierre del documento (líneas 38-39).

Figura 6. Scripts generador y objeto de estudio resultantes:
(a) caja sin alteraciones y (b) caja cortada por un cilindro

```python
#!/usr/bin/env python # ambiente Python

## Librerías a utilizar
import sys
import math
import FreeCAD
import Part
import Mesh
import numpy as np
from FreeCAD import Base

## Parámetros
Largo = 5          #eje x (en milimetros)
Ancho = 5          #eje y (en milimetros)
Alto  = 1.5        #eje z (en milimetros)

## Apertura del documento
App.newDocument("Objeto1")

## Creación del objeto
App.ActiveDocument.addObject("Part::Box","Objeto")
App.ActiveDocument.ActiveObject.Label = "Objeto"
App.ActiveDocument.recompute()

## Modificación del objeto
FreeCAD.getDocument("Objeto1").getObject("Objeto").Length = str(Largo)+'mm'
FreeCAD.getDocument("Objeto1").getObject("Objeto").Width = str(Ancho)+'mm'
FreeCAD.getDocument("Objeto1").getObject("Objeto").Height = str(Alto)+'mm'
FreeCAD.getDocument("Objeto1").getObject("Objeto").Placement = App.Placement(App.Vector(0,0,0),App.Rotation(App.Vector(0,0,1),0))
App.ActiveDocument.recompute()

## Exportación del objeto
__objs__=[]
__objs__.append(FreeCAD.getDocument("Objeto1").getObject("Objeto"))
Mesh.export(__objs__,u"Objeto1.stl")

## Cierre del documento
del __objs__
App.closeDocument("Objeto1")
```

(a)

```python
## Parámetros
Largo        = 5          #eje x (en milimetros)
Ancho        = 5          #eje y (en milimetros)
Alto         = 1.5        #eje z (en milimetros)
Cortar       = 'si'       #realizar corte
levelAnchura = 4.5        #anchura a la que se corta el objeto (en milimetros)
Radio        = 1          #angulo de corte (en grados)

## Apertura del documento
App.newDocument("Objeto1")

## Creación del objeto
App.ActiveDocument.addObject("Part::Box","Base")
App.ActiveDocument.ActiveObject.Label = "Base"
App.ActiveDocument.recompute()

## Modificación del objeto
FreeCAD.getDocument("Objeto1").getObject("Base").Length = str(Largo)+'mm'
FreeCAD.getDocument("Objeto1").getObject("Base").Width = str(Ancho)+'mm'
FreeCAD.getDocument("Objeto1").getObject("Base").Height = str(Alto)+'mm'
FreeCAD.getDocument("Objeto1").getObject("Base").Placement = App.Placement(App.Vector(0,0,0),App.Rotation(App.Vector(0,0,1),0))
App.ActiveDocument.recompute()

if "si" == Cortar:
    App.ActiveDocument.addObject("Part::Cylinder","corte")
    App.ActiveDocument.ActiveObject.Label = "corte"
    App.ActiveDocument.recompute()

    FreeCAD.getDocument("Objeto1").getObject("corte").Radius = str(Radio)+'mm'
    FreeCAD.getDocument("Objeto1").getObject("corte").Height = str(Alto)+'mm'
    FreeCAD.getDocument("Objeto1").getObject("corte").Placement = App.Placement(App.Vector(0,levelAnchura,0),App.Rotation(App.Vector(0,0,1),0))
    App.ActiveDocument.recompute()

    App.activeDocument().addObject("Part::Cut","Objeto")
    App.activeDocument().Objeto.Base = App.activeDocument().Base
    App.activeDocument().Objeto.Tool = App.activeDocument().corte
    App.ActiveDocument.recompute()
else:
    App.activeDocument().addObject("Part::MultiFuse","Objeto")
    App.activeDocument().Objeto.Shapes = [App.activeDocument().Base,App.activeDocument().Base,]
    App.ActiveDocument.recompute()
```

(b)

Fuente: Elaboración propia

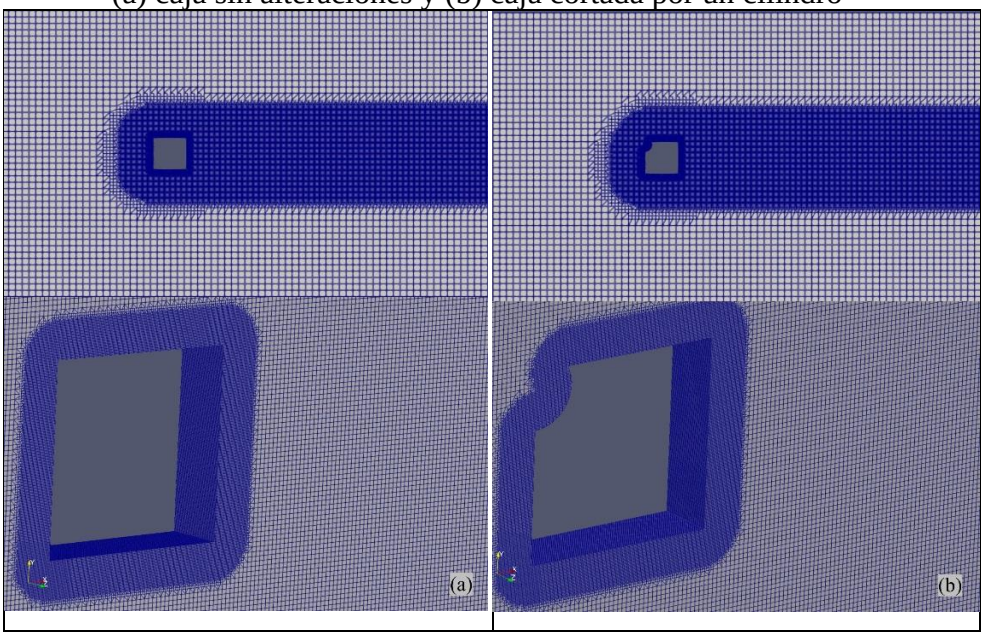
Para el caso de la Figura 6 (b) al generar la caja "cortada" se muestran solo las modificaciones necesarias a partir del *script* del primer caso: parámetros (líneas 2-7), creación por partes del objeto (líneas 13-15 para la caja, 25-27 para el cilindro y 34-37 para unirlos) modificaciones (líneas 18-22 para la caja y 29-32 para el cilindro), con opción para realizar el corte o no hacerlo (líneas 24 y 38) de acuerdo a la entrada en uno de los parámetros. El resto del *script* (ambiente Python, creación del documento, exportación del objeto y cierre del documento) es similar al caso de la caja sin corte. De forma general, sintetizando podemos decir que en este caso se utilizó la función *Cut* entre dos objetos, la parte llamada "Base" y la parte llamada "corte"; esta última consiste en un cilindro trasladado en el eje “y” y con la misma orientación de la caja "Base", de acuerdo

a los parámetros ingresados en el *script.* Esta es una manera muy sencilla y efectiva de aumentar la complejidad del cuerpo de estudio, pudiendo aplicar el mismo procedimiento con diferentes formas y medidas de acuerdo con el cuerpo de estudio requerido, siempre que éste pueda ser parametrizado y formado por la unión o corte de las formas existentes en FreeCAD. Más aún, al diseñar desde un *script* por código, el tiempo necesario para generar objetos parametrizados es menor que si se emplea la interfaz gráfica, y es más eficiente a medida que aumenta la complejidad del objeto.

Resultados

En seguida las Figuras 7 (a) y 7 (b) muestran los mallados (refinamiento en función de la forma del objeto) obtenidos al utilizar las utilerías SurfaceFeatureExtracDict y SnappyHexMeshDict en OpenFOAM, donde se muestra un mallado más refinado alrededor del objeto y su estela (lado derecho del objeto), con un número aproximado de 2,300,000 celdas en el dominio computacional. En este punto se resalta la vinculación existente entre ambos softwares para realizar el diseño de la malla en función del objeto generado. De hecho, esta característica de adaptación entre FreeCAD y OpenFOAM permiten diseñar la malla computacional, la cual generalmente requiere mayor refinamiento conforme aumenta la complejidad del objeto a estudiar, sea un proceso mucho más sencillo siguiendo la secuencia de pasos descritos en la sección anterior.

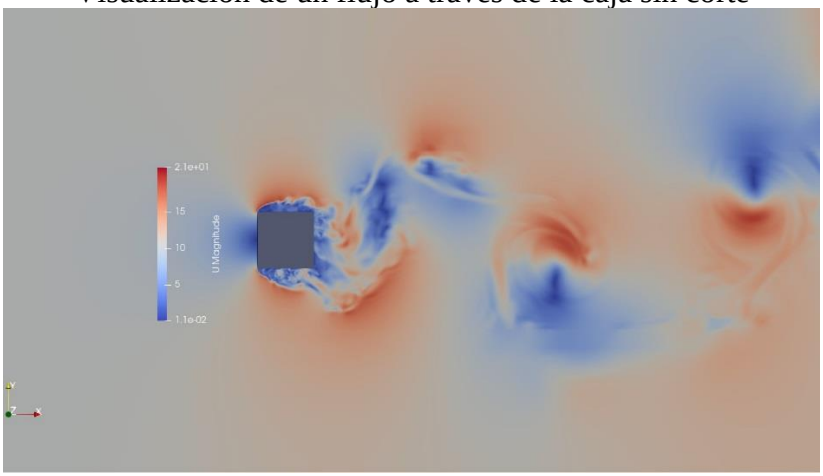
Figura 7. Mallas generadas para ambos objetos con su respectivo zoom: (a) caja sin alteraciones y (b) caja cortada por un cilindro



Fuente: Elaboración propia

La visualización de un flujo a través de la caja sin corte se muestra en la Figura 8, donde se observa el mapa de la magnitud de la velocidad en un instante de tiempo para un número de Reynolds 21,400, teniendo como velocidad de entrada U=10 m/s. Cabe mencionar que para la visualización de las Figuras 7 y 8 se utilizó el software ParaView, de igual forma la integración con ParaView no requiere de realizar nuevamente una importación del objeto "*.stl", sino que directamente de OpenFOAM, se importan los resultados de las variables (velocidad, presión, etc.) en todo el dominio computacional. De esta manera, la integración FreeCAD → OpenFOAM →ParaView resulta una poderosa combinación de herramientas para estudios CFD.

Figura 8. Mapa de velocidad en un instante de tiempo.
Visualización de un flujo a través de la caja sin corte



Fuente: Elaboración propia

Discusión

Haciendo un resumen del presente trabajo, se destaca la presentación de una guía metodológica para la creación de objetos en FreeCAD y su importación en OpenFOAM vía *script* de Python para su uso en simulaciones CFD. Tal guía se describió a manera de manual o tutorial, detallando sus características y funciones necesarias para elaborar el *script*. Se destaca también, que FreeCAD ofrece un enfoque ilimitado de atención a las necesidades en su ramo, al integrar las *workbenches* con su código fuente. De todo lo anterior, queda de manifiesto que este trabajo excede al propósito original expuesto al inicio del documento cuando planea describir las características más importantes de FreeCAD en su aplicación como software modelador de objetos para su uso en aerodinámica computacional.

Conclusión

Basado en la explicación aportada en las secciones de "Método" y "Aplicación Metodológica", este procedimiento de diseño por código tiene ventajas tanto en el tiempo de cómputo como en la organización al realizar estudios paramétricos o simulaciones con objetos dinámicos. Además, se mostró la compatibilidad existente entre FreeCAD y OpenFOAM para la simulación de flujo en aerodinámica computacional. Esto se destacó también mediante la programación por código desde Python, obteniendo un procedimiento más efectivo y fácil de implementar.

Futuras Líneas de Investigación

Como trabajo a futuro se propone realizar manuales similares para OpenFOAM y ParaView, lo que en definitiva podrán ser de gran apoyo para potenciales usuarios.

Referencias

Algar, A., Freire, J., Castilla, R., & Codina, E. (2021). Simulation of Hydraulic Cylinder Cushioning. *Sustainability*, *13*(2), 494. https://doi.org/10.3390/su13020494

Benjamin, J., AL Mashrafi, S., Tejada-Martinez, A., Diaz-Elsayed, N., Arias, M. E., & Zhang, Q. (2022). Optimizing pressure retarded osmosis spacer geometries: An experimental and CFD modeling study. *Journal of Membrane Science*, *647*, 120284. https://doi.org/10.1016/j.memsci.2022.120284

Bibhab Kumar Lodh, Ajoy Kumar Das, & Navtej Singh. (2017). Numerical Comparison of RANS and LES Turbulence Model for Wind Flow Over a Cube in a Turbulent Channel using OPENFOAM. *INTERNATIONAL JOURNAL of ENGINEERING RESEARCH & TECHNOLOGY (IJERT)*, *06*(02). (February 2017). https://doi.org/10.17577/ijertv6is020303

Cantú-González, J. R., Guardado-García, M. del C., & Illescas, G. (2020). Monitoreo de la variabilidad del proceso mediante Gráficas XR, una guía de aplicación estadística. *Revista Electrónica Sobre Tecnología, Educación Y Sociedad*, *7*(14), 203–217. https://doi.org/https://www.ctes.org.mx/index.php/ctes/article/view/726/868

Giannini, G., Rosa-Santos, P., Ramos, V., & Taveira-Pinto, F. (2022). Wave energy converters design combining hydrodynamic performance and structural assessment. *Energy*, *249*, 123641. https://doi.org/10.1016/j.energy.2022.123641

Hasan, Z. H., & Nik Mohd, N. A. R. (2021, November). *Mixed Compression Scramjet Air Intake Aerodynamics with Sharp and Blunt Cowls: A Computational Fluid*

Dynamics Analysis. ICASA 2021- Conference on Aerospace and Aviation 2021, Bandung. https://www.researchgate.net/profile/Nik-Ahmad-Ridhwan-Nik-Mohd/publication/359055972_Mixed_Compression_Scramjet_Air_Intake_Aerodynamics_with_Sharp_and_Blunt_Cowls_A_Computational_Fluid_Dynamics_Analysis/links/6225b6c89f7b32463415080e/Mixed-Compression-Scramjet-Air-Intake-Aerodynamics-with-Sharp-and-Blunt-Cowls-A-Computational-Fluid-Dynamics-Analysis.pdf

Ho, X., Ho, W. S., Wong, K. I., Hassim, M., Hashim, H., Muis, Z., Yunus, A., Hoh, G., & Ling, T. (2022). Study of Fresh Air Supply Vent on Indoor Airflow and Energy Consumption in an Enclosed Space. *CHEMICAL ENGINEERING TRANSACTIONS*, *83*, 2021. https://doi.org/10.3303/CET2183032

Huang, C., Bloching, M., & Lipatnikov, A. N. (2022). A vented corn starch dust explosion in an 11.5 m3 vessel: Experimental and numerical study. *Journal of Loss Prevention in the Process Industries*, *75*, 104707. https://doi.org/10.1016/j.jlp.2021.104707

Jacobs, C. T. (2020). Modelling a Moving Propeller System in a Stratified Fluid Using OpenFOAM. *Fluids*, *5*(4), 217. https://doi.org/10.3390/fluids5040217

Mirdrikvand, M., Sadeghi, M., Pesch, G. R., Dreher, W., & Thöming, J. (2021). Full-Field Comparison of MRV and CFD of Gas Flow through Regular Catalytic Monolithic Structures. *Processes*, *9*(3), 566. https://doi.org/10.3390/pr9030566

Oblak, B., Babnik, S., Erklavec-Zajec, V., Likozar, B., & Pohar, A. (2020). Digital Twinning Process for Stirred Tank Reactors/Separation Unit Operations through Tandem Experimental/Computational Fluid Dynamics (CFD) Simulations. *Processes*, *8*(11), 1511. https://doi.org/10.3390/pr8111511

OpenCFD Ltd. (2022). *OpenFOAM® - Official home of The Open Source Computational Fluid Dynamics (CFD) Toolbox*. Www.openfoam.com; OpenCFD Ltd. https://www.openfoam.com/

Pham, T. Q. D., & Choi, S. (2021). Numerical analysis of direct contact condensation-induced water hammering effect using OpenFOAM in realistic steam pipes. *International Journal of Heat and Mass Transfer*, *171*, 121099. https://doi.org/10.1016/j.ijheatmasstransfer.2021.121099

Python Software Foundation. (2019, May 29). *Welcome to Python.org*. Python.org; Python Software Foundation. https://www.python.org/

Ramirez-Cortes, J. M., Alarcon-Aquino, V., Gomez-Gil, P., Diaz-Mendez, A., Ibarra-Bonilla, M., & García-Enriquez, I. (2012). Interactive educational tool for

compensators design in MATLAB® using frequency response analysis. *Computer Applications in Engineering Education*, *22*(4), 699–707. https://doi.org/10.1002/cae.21562

Rhee, D. W., Pendse, J., Chan, H., Stern, D. T., & Sartori, D. J. (2021). Mapping the Clinical Experience of a New York City Residency Program During the COVID-19 Pandemic. *Journal of Hospital Medicine*, *16*(6). https://doi.org/10.12788/jhm.3623

Sadadiwala, V. (2021). A Detailed Study: CFD Analysis of NACA 0012 at Varying Angles of Attack. *International Journal for Research in Applied Science and Engineering Technology*, *9*(VII), 2330–2336. https://doi.org/10.22214/ijraset.2021.36843

Said, M. B., & Ouamane, A. (2021). Performance of rectangular labyrinth weir – an experimental and numerical study. *Water Supply*. https://doi.org/10.2166/ws.2022.005

Solmaz, S., & Van Gerven, T. (2021). Automated integration of extract-based CFD results with AR/VR in engineering education for practitioners. *Multimedia Tools and Applications*. https://doi.org/10.1007/s11042-021-10621-9

The FreeCAD Team. (2019). *FreeCAD: Your own 3D parametric modeler*. Freecadweb.org. https://www.freecadweb.org/

The FreeCAD team. (2022, March 3). *External workbenches - FreeCAD Documentation*. Wiki.freecad.org. https://wiki.freecad.org/External_workbenches

Torres, J. A., Naranjo, F. J., Torres, D. T., & Quiroz, J. (2018). Vibrational behaviors studied through experiments and simulations using free licensing cross-platform software. *Revista Mexicana de Física E*, *64*(1 Jan-Jun), 92–99. https://doi.org/10.31349/revmexfise.64.92

Welnitzová, D., & Molnár, V. (2016). High speed flow verification using open source CFD software. *AIP Conference Proceedings*, *1768*(020038). https://doi.org/10.1063/1.4963060

Yadav, D., & Patel, R. (2018). Interactive educational tool for the design of compensators using frequency response analysis. *International Journal of Electrical Engineering Education*, *55*(1), 44–61. https://doi.org/10.1177/0020720917750958

Zhang, X., Xie, F., Ji, T., Zhu, Z., & Zheng, Y. (2021). Multi-fidelity deep neural network surrogate model for aerodynamic shape optimization. *Computer Methods in*

Applied Mechanics and Engineering, *373*, 113485. https://doi.org/10.1016/j.cma.2020.113485