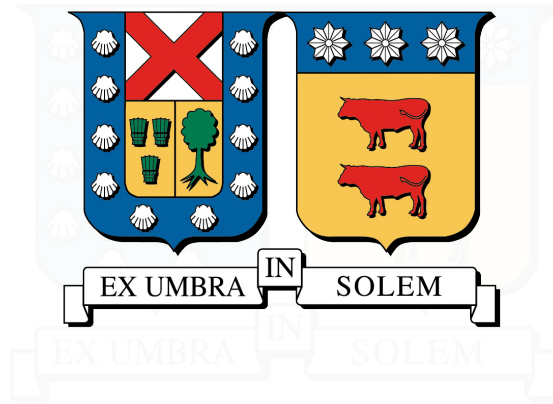


UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INGENIERÍA MECÁNICA
SANTIAGO - CHILE



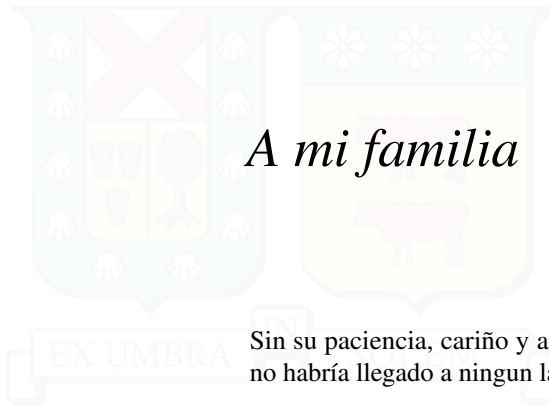
Estudio de los efectos de erosión en palas hidráulicas de turbinas, por medio de simulación computacional de un banco de pruebas utilizando OpenFOAM®

FRANCISCO JAVIER RAMÍREZ ADRIAZOLA

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL MECÁNICO

PROFESOR GUÍA : DR. RODRIGO CASSINELI PALHARINI
PROFESOR CORREFERENTE : DR. ING. JAIME NUÑEZ SEGOVIA

AGOSTO 2023



A mi familia

Sin su paciencia, cariño y apoyo incondicional
no habría llegado a ningún lado.

RESUMEN EJECUTIVO

Las turbinas Pelton son uno de los componentes fundamentales en la generación de energías renovables no convencionales (ERNC), que se ve afectado por distintos factores que disminuyen su rendimiento. Entre estos factores se encuentra el desgaste de sus palas hidráulicas debido a lo que se conoce como **slurry erosion**, en otras palabras, el impacto de partículas sólidas diluidas en un flujo de agua que choca contra estas. En este documento se hace una revisión en la literatura sobre los parámetros que influyen en este tipo de erosión de los que se destaca la velocidad y ángulo de impacto.

El objetivo principal de este documento validar como herramienta el software OpenFOAM ® para simular computacionalmente un Jet Impingement Test (JIT), y así poder estudiar como afectan ciertos parámetros en la erosión. Por otra parte, el JIT se puede considerar la prueba experimental más popular dentro de la bibliografía de este tipo de erosión, ya que ofrece un alto rango de velocidad y ángulo de impacto en un área reducida y su relativa facilidad para medir y visualizar resultados. Entonces, junto a esto, proponer un diseño conceptual de un banco de pruebas potencialmente pueda validar los datos de las simulaciones hechas y futuras.

La metodología de las simulaciones es utilizar base el artículo *CFD study of Jet Impingement Test erosion using Ansys Fluent and OpenFOAM. Alejandro Lopez , William Nicholls , Matthew T. Stickland , William M. Dempster [1]* donde se simula la salida de un flujo de una tobera hacia una placa plana de un JIT y se calculan las velocidades y ángulos de impacto promedio. Por otra parte, se aplica metodología de diseño en base a que funciones debe realizar un banco de pruebas por lo que en base a las simulaciones hechas y en la revisión bibliográfica se fue resolviendo como realizar estas funciones hasta llegar a un diseño base, junto a dos variaciones respecto a la forma en que se mezclan las partículas en el flujo transportador, en otras palabras, pre o post-mezclado (antes o después de que el fluido es presurizado).

Por medio de la versión 8 de OpenFOAM se replicaron exitosamente los resultados del autor Alejandro Lopez, aplicando el algoritmo de modelo de partícula discreta o DPM (Discrete Particle Model, que aplica el enfoque matemático Euler-Lagrange, en otras palabras simular el agua como un flujo continuo con el método de volúmenes finitos, resolviendo ecuaciones de Navier-Stokes, mientras que por otro lado las partículas se rastrea su trayecto de forma lagrangiana. Se realizaron otras simulaciones más, para visualizar el perfil de erosión en una placa plana aplicando el modelo de erosión de Finnie.

Una vez validadas las simulaciones se propone un banco de pruebas conceptual, cuyas principales características son utilizar una bomba peristáltica (que disminuya el desgaste tanto de partículas como de los componentes de la bomba), que se repliquen las condiciones de las simulaciones en una cámara de prueba transparente, la cual permite la aplicación de método PIV para adquirir datos, esto servirá para validar las simulaciones hechas y futuras. Finalmente se propone también una bomba de piscina para poder reutilizar el agua, sin reutilizar las partículas sólidas ya que estas se deforman luego de impactar.

Palabras Clave. Turbina Pelton, Erosión, JIT, Banco de pruebas, OpenFoam®, DPM, Euler-Lagrange, PIV, Finnie.

ABSTRACT

Pelton turbines are one of the fundamental components in the generation of non-conventional renewable energy (ERNC), which is affected by different factors that reduce its performance. Among these factors is the wear of its hydraulic blades due to what is known as **slurry erosion**, in other words, the impact of solid particles diluted in a water flow that crashes against them. On this document reviews the literature on the parameters that influence this type of erosion, of which speed and angle of impact stand out.

The main objective of this document is to validate the OpenFOAM® software as a tool to computationally simulate a Jet Impingement Test (JIT), and thus be able to study how it affects certain parameters in erosion. On the other hand, JIT can be considered the most popular experimental test within the bibliography of this type of erosion, since it offers a wide range of speed and angle of impact in a small area and its relative ease to measure and visualize results. So, along with this, propose a conceptual design of a potential test bench that can validate the data from the simulations made and future ones.

The methodology of the simulations is based on the article *Erosion CFD study by jet impact test using Ansys Fluent and OpenFOAM*. Alejandro Lopez, William Nicholls, Matthew T. Stickland, William M. Dempster [1] where the output of a flow from a nozzle towards a flat plate of a JIT is simulated and the average speeds and angles of impact are calculated. On the other hand, a design methodology is applied based on what functions a test bench must carry out, so that based on the simulations carried out and the bibliographic review, it is resolved how to carry out these functions until reaching a base design, together with two variations of the way the particles are mixed in the transport flow, that is, pre or post mixing (before or after the fluid is pressurized).

Through version 8 of OpenFOAM, the results of the author Alejandro Lopez were successfully replicated, applying the discrete particle model algorithm or DPM (Discrete Particle Model, which applies the mathematical approach Euler-Lagrange, in other words to simulate water as a continuous flow with the finite volume method, by solving Navier-Stokes equations, while on the other hand the particles are tracked in a Lagrangian way. Other simulations were performed to visualize the erosion profile on a flat plate applying the model of Finnie's erosion.

Once the simulations have been validated, a conceptual test rig is proposed, whose main characteristics are to use a peristaltic pump (that reduces the wear of both particles and the components of the pump), which replicates the conditions of the simulations in a transparent test chamber, that allows the application of the PIV method to acquire data, this will serve to validate the simulations made and future ones. Finally, a pool pump is also proposed to be able to reuse the water, without reusing the solid particles since they deform after impact.

Keywords. Pelton Turbine, Erosion, JIT, Test rig, OpenFOAM®, DPM, Euler-Lagrange, PIV, Finnie.

INTRODUCCIÓN

La generación de energía es un factor fundamental en el desarrollo económico de cualquier país. Junto a esto, actualmente la búsqueda de fuentes de generación de energías renovables no convencionales (ERNC) es un tema que ha tenido mayor interés esta última década que en toda la historia. Entre esas fuentes se encuentra la generación hidroeléctrica, la cual tiene distintas clasificaciones respecto al su diseño, potencia, caudal, pero todas tienen en común en que necesitan una fuente de flujo acuático del cual se pueda sacar provecho su energía potencial. Sin embargo, no existe una fuente que tenga agua en estado puro, sino que siempre vendrá mezclado con otros componentes, de los cuales se destacan partículas sólidas, como arena, cuarzo, etc.

De las múltiples dificultades que puede conllevar la operación de una central hidroeléctrica, se encuentra el problema de erosión particulada. En las referencias [2] y [3] se habla de el efecto negativo que tiene este tipo de erosión en componentes de la industria en general, de como la pérdida de material conlleva a una falla temprana en piezas fundamentales. En las referencias [4] y [5] se trata el tema de la erosión relacionado con plantas hidroeléctricas, de como la erosión afecta principalmente componentes esenciales en las turbinas y de que produce pérdidas monetarias considerables a largo plazo. Mientras que en la referencia [6] se habla de la importancia de estudiar experimentalmente (en condiciones de laboratorio) los parámetros que influyen en la erosión y los efectos que producen sobre un material.

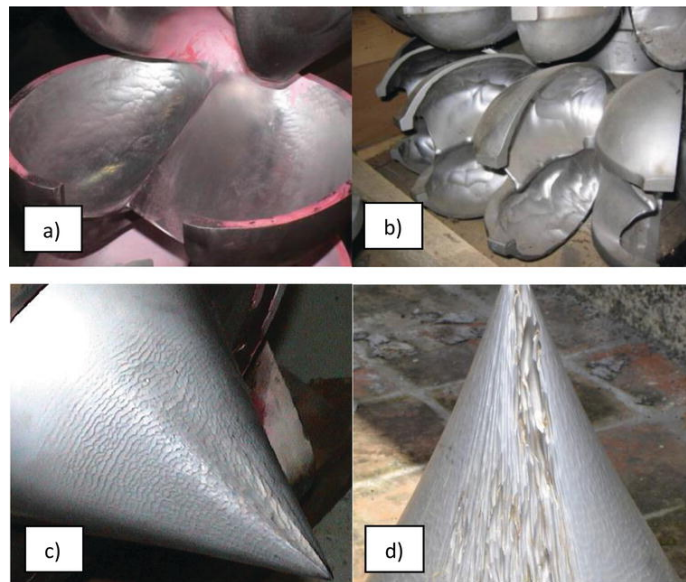


Figura 1: Ejemplo de erosión en pala e inyector de una turbina Pelton [5].

Entonces, es importante el estudio de como puede afectar este problema, especificando ciertas condiciones ya que la clasificación el fenómeno de erosión y los factores que lo afectan son bastante amplios.

En la referencia [6] se menciona también, lo común que es el estudio del fenómeno por medio de la dinámica de fluidos computacional, cuyas siglas en inglés son CFD (Computer Fluid Dynamics). El uso de CFD ayuda a la predicción de erosión en base a modelos teóricos para poder estudiar los efectos de este fenómeno en distintos materiales en diferentes condiciones evitando (hasta cierto punto) la necesidad de costosas pruebas experimentales. Sin embargo, toda simulación en CFD, sin importar cual software se utilice, siempre se requiere haya **verificación y validación**.

En la tesis [7] se define de verificación como:

”El proceso de determinar que una implementación de modelo representa acertadamente la descripción conceptual del desarrollador del modelo y a la solución de este”

Mientras que validación se define como:

”El proceso de determinar el grado con el cual un modelo es una representación precisa del mundo real”

El objetivo de verificación es tratar de cuantificar los errores en la implementación y solución de algún cálculo para estimar la certeza de este. La validación conlleva a comparar resultados obtenidos con datos experimentales o resultados de otras simulaciones. Es importante tener en cuenta las incertidumbres de los datos en la literatura. También considerar que las soluciones numéricas son siempre aproximaciones, y en la referencia [7] se nombran tres tipos de errores sistemáticos:

- **Error de modelo:** La diferencia entre el mundo real y la solución matemática.
- **Error de discretización:** La diferencia entre soluciones exactas de las ecuaciones de conservación y la solución exacta del sistema de ecuaciones discretizado.
- **Error de iteración:** La diferencia entre la solución iterativa y la solución exacta del sistema de ecuación discretizada.

En el presente trabajo, al estar relacionado con turbinas Pelton, y relacionado con flujos de agua mezclada con una cierta concentración de partículas sólidas, se busca simular computacionalmente un modelo de JIT (Jet Impingement Test), es decir una prueba de impacto de un chorro sobre un cuerpo. Se requiere utilizar un modelo que simule un flujo multifásico, específicamente un flujo continuo de líquido con partículas sólidas dispersas en este. Al mismo tiempo, en esta simulación se requiere de un modelo matemático que sirva para predecir la erosión sobre el material. Entonces, para verificación y validación se utiliza como base el estudio hecho por **Alejandro López**, respecto al modelamiento de un flujo con partículas sobre una superficie plana utilizando OpenFOAM®(Versión 2.2) y ANSYS Fluent (*CFD study of Jet Impingement Test erosion using Ansys Fluent and OpenFOAM. Alejandro Lopez , William Nicholls , Matthew T. Stickland , William M. Dempster [1]*) y la comparación de distintos modelos de erosión implementados a un mismo dominio computacional en OpenFOAM®[8]. La base teórica de estos artículos se explicará con más detalle en el capítulo 2, complementando con otros textos que se consideran relevantes para el tema.

Se busca utilizar la versión 8 del programa OpenFOAM, el cual es un software de código abierto y que no requiere licencia. En el artículo de A. Lopez, se busca ver qué tan similares son los resultados de simulaciones de JIT (jet impingement test) respecto al software de Ansys, en orden de validar el uso de este software para la simulación de flujos con partículas. Por consiguiente, este será base en la metodología computacional y en los parámetros de las simulaciones como se explicará en los capítulos 4 y 5 respectivamente.

Tal como se plantea en el paper, hay muchas teorías y modelos de la erosión. Por lo que el autor optó por enfocarse en la erosión producida por un flujo de agua mezclada con 1 % partículas sobre una placa plana. De la teoría de erosión se aplica específicamente en dos tipos de desgaste producidos por el choque de partículas sólidas: desgaste por corte y desgaste por deformación:

Desgaste por corte

- Predominante en materiales dúctiles
- Común para flujos con bajo ángulo de ataque

Desgaste por deformación

- Predominante en materiales frágiles
- Común para flujos con ángulo de ataque alto
- La deformación puede aumentar el límite de elasticidad, lo que provoca que el material se vuelva más duro y por consiguiente más frágil

Además, similar a lo que se ve en la figura 2, los parámetros más relevantes que se consideraron fueron:

- Ángulo de impacto
- Propiedades del material
- Propiedades de partículas sólidas (tamaño)

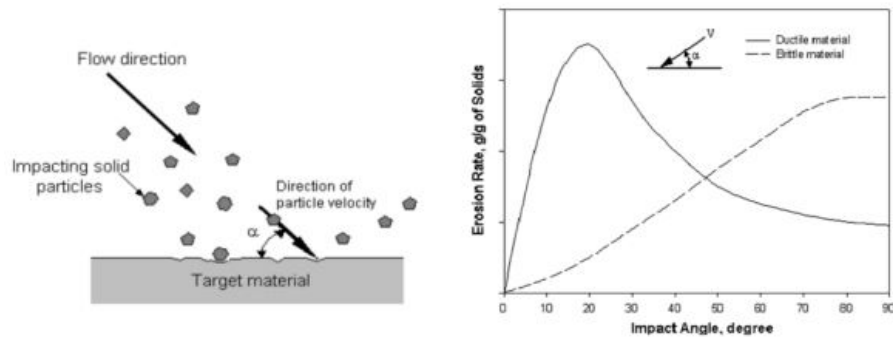


Figura 2: Modelo de desgaste de un material expuesto a un flujo

(Fuente: [9])

Al ser un flujo de agua particulado, es decir, un flujo multifásico (líquido-sólido) formado por una fase continua (agua) y por con baja concentración de material sólido, en otras palabras, fase discreta (partículas sólidas). Para la simulación computacional, el autor opta por que fuera modelado con enfoque Euler-Lagrange (Capítulo 3). Entonces, para el rastreo lagrangiano o Lagrangian Particle Tracking (2.4.1), en este estudio se considera que las partículas tienen todas la misma forma esférica, mismo diámetro reducido, mismo material (por consiguiente misma densidad), y poca concentración en relación al volumen de la fase continua. Adicionalmente, en el enfoque lagrangiano, se ven 3 tipos de acoplamientos (coupling) que dependen de la concentración de partículas en el flujo y de como estas se relacionan entre ellas y con la fase continua:

Acoplamiento unidireccional. Para baja concentración de partículas. Se puede no considerar la fuerza de las partículas

Acoplamiento bidireccional. Se incluye la fuerza de las partículas en las ecuaciones de flujo.

Acoplamiento cuatro direcciones. Para mayores concentraciones. Se considera interacción partícula-partícula.

Al ser baja concentración, se opta por acoplamiento unidireccional por lo que se requiere un balance de fuerza en base a la segunda ley de Newton para ver los efectos de la fase continua sobre la fase discreta. Consecutivamente, se desprecian las fuerzas de rotación, presión y gravitacionales entre otras, reduciendo la ecuación de equilibrio de fuerzas solamente a las producidas por esfuerzos viscosos, en otras palabras, fuerza de arrastre (drag). Se determinó que en OpenFOAM se podían llegar a resultados válidos como en ANSYS.

Luego de verificar y validar los resultados, se busca implementar este código de simulación, para otro dominio computacional de un JIT sumergido, que permita predecir el comportamiento de un flujo multifásico y el perfil de erosión sobre una superficie metálica cuyas propiedades asemejen al material que conforman una pala hidráulica de una turbina Pelton. Este dominio computacional se debe basar en la geometría de la propuesta conceptual de un banco de pruebas de erosión, hecho para este trabajo y explicado en el capítulo 7.

OBJETIVOS

Objetivo General

Simular un Jet Impingement Test (JIT) para evaluar la herramienta OpenFOAM® en aplicaciones de erosión en materiales. Proponer un diseño de banco de pruebas que potencialmente pueda generar datos experimentales y validar los datos obtenidos a través de simulaciones computacionales.

Objetivos Específicos

- Estudiar el uso del software OpenFOAM®, junto otros antecedentes bibliográficos relevantes (aparte del estudio usado como base) para poder complementar.
- Determinar que parámetros es necesario enfocarse para las simulaciones computacionales.
- Plantear que modelo matemático es adecuado para el o los escenarios que se busque simular.
- Validar la simulación numérica con datos disponibles en la literatura.

PLAN DE TRABAJO

Alcance

El estudio de los casos de simulación de JIT buscan verificar y validar si la versión 8 del software OpenFOAM puede llegar a resultados suficientemente similares a los resultados obtenidos previamente por el autor Alejandro López [1] (quien utilizó una versión más antigua de este software). Para las simulaciones solo se utilizan las herramientas, solvers, diccionarios y modelos disponibles en la versión utilizada del software, NO se implementará un modelo numérico de flujo o de erosión que requiera utilizar lenguaje C++ para poder realizarse, ya que eso va más allá del objetivo de aprendizaje de esta tesis.

Si se cuenta con esta información, se puede hacer otra simulación que cuyo dominio y condiciones de borde/iniciales representen una propuesta de banco de pruebas que sea posible construir en las instalaciones de la facultad. Además, se desea verificar también este código con datos de otras simulaciones hecha por otro(s) autor(es).

Es importante destacar que para este trabajo, NO se construirá el banco de prueba sino que proponer un diseño netamente conceptual y preliminar, en base a información disponible en la revisión bibliográfica.

Estructura de la memoria

- 2. Estado del arte:** Revisión de literatura respecto a la teoría de la erosión, explicar brevemente factores se consideran más influyentes en su tasa. Informaciones sobre banco de pruebas de este fenómeno y algunas simulaciones, específicamente lo realizado por el autor A. Lopez que se utiliza como modelo de verificación y validación.
- 3. Marco teórico:** Explicación de conceptos claves en las simulaciones de flujo multifase y modelamiento de erosión aplicadas en este informe.
- 4. Método computacional:** Introducción al software utilizado para las simulaciones junto a una explicación de como se aplicaron y ajustaron ciertos parámetros esenciales para el funcionamiento de la simulación.
- 5. Parámetros computacionales:** Continuando con lo explicado en el capítulo anterior, se da mayor detalle de dimensiones, modelos, comandos, entre otras cosas que se aplicaron para realizar las simulaciones.
- 6. Verificación y validación de modelo CFD** Comparación de los resultados obtenidos con aquellos disponibles en la literatura.
- 7. Propuesta de banco de pruebas:** Metodología de diseño. Se determine que funciones esenciales debe cumplir el banco de prueba, teniendo en cuenta que se pueda replicar los parámetros de las simulaciones y otorgue libertad de modificarlos para nuevos experimentos, junto con esto una revisión bibliográfica que permita justificar algunas de las decisiones de diseño tomadas.
- 8. Conclusiones:** Revisar, comentar sobre el cumplimiento de los objetivos de este informe. Además plantear puntos para trabajos futuros.

Índice de Contenidos

1. Estado del arte	1
1.1. Factores/Parámetros que afectan la erosión	2
1.1.1. Características material:	2
1.1.2. Condiciones de impacto:	2
1.1.3. Propiedades partículas:	4
1.1.4. Características mezcla:	5
1.2. Simulación computacional	6
1.3. Modelos de erosión	7
1.4. Pruebas experimentales de erosión	8
1.4.1. Recolección de datos	11
2. Marco Teórico	14
2.1. Método de volúmenes finitos	14
2.1.1. Ecuaciones gobernantes	14
2.2. Modelamiento de flujo multifase	15
2.3. Modelamiento de turbulencia	16
2.4. Modelo partícula discreta (DPM)	18
2.4.1. Lagrangian Particle Tracking	18
3. Método computacional	20
3.1. Introducción a OpenFOAM	20
3.1.1. Estructura de una simulación (case)	20
3.1.1.1. Directorio 0	21
3.1.1.2. Directorio Constant	21
3.1.1.3. Directorio System	21
3.2. Pre-Proceso	22
3.2.1. Geometría y mallado	22
3.3. Modelos numéricos	23
3.3.1. Solver	23
3.3.2. Modelo de turbulencia	23
3.3.3. Discretización temporal	23
3.3.4. Modelo de erosión	24
3.4. Post-procesamiento	24
3.4.1. ParaView	24
3.4.2. Python	24
3.4.3. G3Data Graphs	25
4. Parámetros computacionales	26
4.1. Geometría dominio computacional	26
4.2. Condiciones de borde/iniciales	27
4.3. Configuración simulación	28

4.4.	Configuración de partículas	29
4.4.1.	Ajustes generales	30
4.4.2.	Propiedades físicas	30
4.4.3.	Fuerzas actuando sobre las partículas	31
4.4.4.	Inyección de partículas	31
4.4.5.	Modelos y coeficientes de interacción de partículas	32
4.4.6.	Funciones de nube (CloudFunctions)	33
4.5.	Simulación paralela o Paralelización	34
4.6.	Problemas en el transcurso de las simulaciones	34
4.6.1.	Problemas con las condiciones de borde	34
4.6.2.	Problemas con el mallado	34
4.6.3.	Problema con el post-procesamiento	35
5.	Validación de modelo CFD	37
5.1.	Primer caso de validación código:	37
5.1.1.	Comparación de perfil de velocidades	37
5.1.2.	Visualización de impactos sobre la superficie de la muestra	38
5.1.3.	Velocidad promedio y ángulo de impacto vs distancia del centro	39
5.2.	Segundo caso de validación de código:	41
5.3.	Tercer caso de validación:	43
6.	Proyecto de banco de pruebas para erosión	47
6.1.	Definición del problema y Objetivos	47
6.2.	Estructura funcional	48
6.3.	Generación de conceptos	49
7.	Conclusiones	58
7.1.	Trabajo futuro (Further Work)	59
7.1.1.	Modelos de erosión	59
7.1.2.	Geometría de la muestra (target)	59
7.1.3.	Modelo de interacción partícula-pared	59
7.1.4.	Diseño Banco de pruebas	60
	Bibliografía	61
A.	Anexo	64
A.1.	Códigos en OpenFOAM	64
A.1.1.	U/0	64
A.1.2.	p/0	65
A.1.3.	epsilon/0	66
A.1.4.	k/0	67
A.1.5.	nut/0	67
A.1.6.	decomposeParDict	68
A.2.	blockMeshDict/system	69
A.3.	Tablas Validación	77
A.3.1.	DPM 5 mm	77
A.3.2.	DPM 25 mm	77
A.4.	Imágenes	78

Índice de Tablas

2.1. Constantes empíricas del modelo turbulento	17
4.1. Condiciones de borde para cada patch/límite definido	28
5.1. Resultados de erosión obtenidos por las simulaciones de diferentes modelos realizadas en [8], re-graficadas en Python.	42
6.1. Lista de atributos y categorización	47
6.2. Lista de especificaciones de diseño y categorización	50
A.1. Datos Validación de datos 5.1.3 para JIT 5 mm de distancia	77
A.2. Datos re-validación de datos Fig. 5.13 para JIT 5 mm de distancia	77
A.3. Datos Validación de datos 5.1.3 para JIT 25 mm de distancia	77
A.4. Datos re-validación de datos Fig. 5.13 para JIT 25 mm de distancia	78

Índice de Figuras

1.	Ejemplo de erosión en T. Pelton	IV
2.	Modelo de desgaste	VI
1.1.	Diagrama Causa-Efecto de erosión	1
1.2.	Gráficos erosión vs ángulo de impacto	3
1.3.	Efectos impacto sobre una superficie, respecto si el material es dúctil o frágil, y en relación al ángulo de impacto de la partícula	3
1.4.	Comparación de profundidad erosión según distintas velocidades de entrada	4
1.5.	Relación erosión con tamaño de partícula con distintas velocidades [10]	5
1.6.	Diferentes regímenes de flujos de dos fases	6
1.7.	Clasificación de banco de prueba según su método de mezcla	9
1.8.	Ejemplos de uso de reducción de sección transversal de tobera para producir succión de mezcla con material sólido.	11
1.9.	Layout Tubo de Venturi diseñado para ingresar material sólido al flujo propuesto por A. Lopez	12
1.10.	Layout de propuesta de H.S. Grewal	13
2.1.	Ilustración de número de Stokes	16
2.2.	Algoritmo Euler-Lagrange	18
4.1.	Geometría para distancia 5 mm	26
4.2.	Geometría para distancia 25 mm	27
4.3.	Tipos de mallado	35
5.1.	Perfil velocidad (5 mm)	37
5.2.	Perfil velocidad (25 mm)	38
5.3.	Impactos de partícula a 5mm de distancia	38
5.4.	Impactos de partícula a 25mm de distancia	39
5.5.	Gráfico de velocidades y ángulos de impacto promedios vs distancia desde el centro del target (5 mm)	39
5.6.	Gráfico de velocidades y ángulos de impacto promedios vs distancia desde el centro del target (25 mm)	40
5.7.	Gráfico de velocidades promedios vs distancia desde el centro comparativo	40
5.8.	Gráfico de ángulos de impacto promedios vs distancia desde el centro del target	41
5.9.	Error % entre los resultados esperados y obtenidos (JIT 5mm izquierda y JIT 25mm derecha).	41
5.10.	Comparación de resultados de perfil de erosión obtenidos en OpenFoam®, con los obtenidos por los distintos modelos simulados por A. Lopez.	42
5.11.	Comparación de resultados de perfil de erosión (con rebote en el target) obtenidos en OpenFoam®, con los obtenidos por los distintos modelos simulados por A. Lopez.	42
5.12.	Dominio computacional y mallado para simular banco de pruebas	43
5.13.	Revalidación de resultados con nuevo dominio computacional.	44
5.14.	Revalidación de resultados con nuevo dominio computacional.	45
5.15.	Perfiles de erosión obtenidos al variar la velocidad de entrada del flujo	46

6.1. Árbol de objetivos	48
6.2. Árbol de funciones	48
6.3. Diagrama Caja Negra	49
6.4. Diagrama Lluvia de ideas (Brainstorming)	51
6.5. Diseño conceptual 3D de banco de pruebas propuesto para este informe	52
6.6. Curva Potencia de Bombas Bredel SP32 y SP40	53
6.7. Vista Portador de muestra - Tobera	54
6.8. Vista 3D de diseño conceptual con agitador incluido	55
6.9. Inyección de partículas en diseño de A. López	56
6.10. Layout Tubo de aceleración	56
6.11. Perfil no simétrico debido a la mezcla no uniforme	57
6.12. Vista 3D de diseño conceptual con bomba peristáltica extra	57
A.1. Mallado centro del dominio	78
A.2. Gráficos de ubicación de impactos sobre superficie de muestra obtenidos por A. Lopez en OpenFOAM	79
A.3. Vista ISO de conceptual de portador de pala Pelto	80
A.4. Vista en 3D cabezal portador de pala Pelton	80
A.5. Dimensiones bomba Bredel SP32	81
A.6. Dimensiones bomba Bredel SP40	82

1 | Estado del arte

En esta investigación, se busca un mejor entendimiento del proceso de erosión causado por agua y partículas sólidas al colisionar con una superficie (target) plana y proponer la construcción de un banco de pruebas experimental para investigar el proceso de erosión en geometrías complejas. En esta sección se explica la teoría de erosión y sus diferentes modelos.

Existe una gran variedad de tipos de desgaste, deterioro en materiales como también causas y factores que influyen en esto. En el libro *GUIDE TO FRICTION, WEAR, AND EROSION TESTING* [11], se define primero el desgaste (wear) como la pérdida de un material debido a un proceso, lo cual sub-divide en abrasivo, lubricación, fricción y erosión. La erosión por su parte, siendo un tipo de desgaste que requiere un de un flujo, se subdivide según los elementos actuando en el desgaste que pueden ser flujo gaseoso (temperaturas extremas, alta velocidad), flujo de agua (gotas, cavitación), o la mezcla de partículas sólidas con uno de estos dos flujos, lo que se conoce como *slurry erosion*.

En la literatura, se encuentra el concepto de *slurry*, el cual se traduce como "lodo" en cualquier diccionario. Pero, por definición no se refiere a lo que se conocería como barro (mud), sino que la mezcla de un fluido con partículas de bajo tamaño suspendidas en el, por consiguiente, un *slurry flow*, se refiere a un flujo que transporta/arrastra partículas sólidas [11].

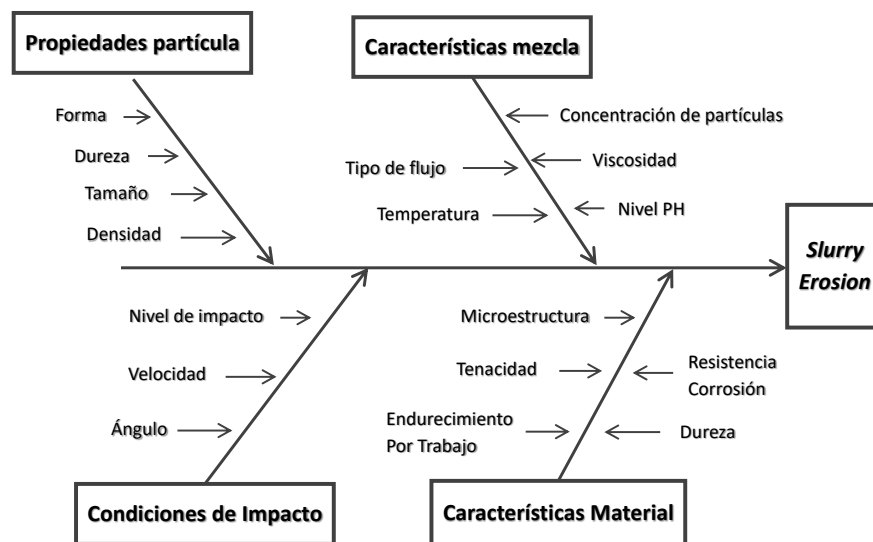


Figura 1.1: Diagrama Causa-Efecto de erosión.

(Fuente: Adaptado de Referencias [2] y [12])

1.1. Factores/Parámetros que afectan la erosión

La erosión es un fenómeno complejo y no ha sido entendido por completo. A partir del siglo XX, el fenómeno de erosión ha sido ampliamente estudiado y a su vez, varias teorías han sido planteadas para describir el mecanismo que ocurre cuando las partículas impactan y remueven material de una superficie. Se destacan como pioneros en esta área a los autores I. Finnie [13] y J. Bitter [14, 15]. En base a sus estudios, hasta hoy en día, se consideran dos mecanismos principales como responsables del fenómeno de erosión por partículas sólidas, los cuales son **corte** y **deformación**.

El autor Vahid Javaheri [2] hace una revisión de varios estudios de modelos y experimentos relacionados a este tipo de erosión específicamente en acero. Esto complementado con el artículo *Slurry erosion: An overview (2020)* [12], y al marco teórico de otros artículos presentes en bibliografía, se puede agrupar en 4 categorías los parámetros principales que afectan este tipo de erosión, los cuales se pueden apreciar en la Figura 1.1, organizados en un Diagrama causa-efecto (conocido como diagrama de Ishikawa o espina de pescado). Estas categorías se relacionan entre si, pero algunas afectan de manera más directa, y suelen ser la que se consideran al crear modelos teóricos para predecir la erosión, como por ejemplo las condiciones de impacto (la velocidad y ángulo de impacto de las partículas), junto con las propiedades mecánicas del material del prueba, mientras que otros como las características del flujo se suelen considerar que afectan de manera más indirecta ya que influyen en la erosión, debido a que afectan las condiciones de impacto.

En orden de sintetizar estas categorías, a continuación se explica lo revisado en la literatura, y lo que se consideró más relevante para el presente trabajo. Cabe mencionar, que el daño que puede producir una partícula cuando golpea la superficie de un material se conoce como erosión potencial y esto se ve reflejado en un perfil sobre una superficie. La mayoría de los estudios de erosión se hacen en superficies planas en orden de entender como afectan en el este perfil mencionado los factores y parámetros. Muchos de los factores se plantean de forma teórica la erosión potencial que pueden generar, pero otra condición puede hacer que esto cambie.

1.1.1. Características material:

La micro-estructura, tenacidad, dureza del material de muestra pueden afectar la forma en que el impacto de partícula les afecta, dando paso a una gran variedad de estudios debido a la gran diversidad de materiales y sus respectivas propiedades. No obstante, el uso de las propiedades mecánicas para estimar el ratio de erosión de materiales siempre ha sido un tema atractivo gracias al menor costo de estudio, la simplicidad y disponibilidad de datos para muchos materiales. Es común que de manera general, el mecanismo de erosión se subdivide según la ductilidad y la fragilidad del material. En un material frágil es común la formación de grietas y quiebres en la superficie (Figura 1.2b), mientras que en un material dúctil (Figura 1.2a) se le atribuyen más los mecanismos de desgaste por corte y deformación mencionados previamente. A su vez, el desgaste por deformación, cuando hay impactos muy repetitivos en ciertos materiales, se produce deformación plástica y endurecimiento local, y en consecuencia se produce una fractura en el material independiente de si este es dúctil.

Existe la posibilidad de que ambos tipos de erosiones ocurran simultáneamente dependiendo de las propiedades de diferentes materiales, así como también habrá uno más dominante que el otro en consecuencia de otros parámetros. Siendo relevante para este trabajo destacar (en base a lo planteado por J. Bitter [14]) de la relación directa con el ángulo de impacto. En la Figura 1.2 se ve que para materiales frágiles la erosión alcanza su punto máximo en ángulos altos mientras que el desgaste por corte alcanza su máximo en ángulos de impacto menores.

1.1.2. Condiciones de impacto:

El ángulo de impacto se relaciona directamente con el tipo de material y el tipo de mecanismo de erosión. Se considera un impacto normal si el ángulo es cercano a 90 grados mientras que ángulos cercanos

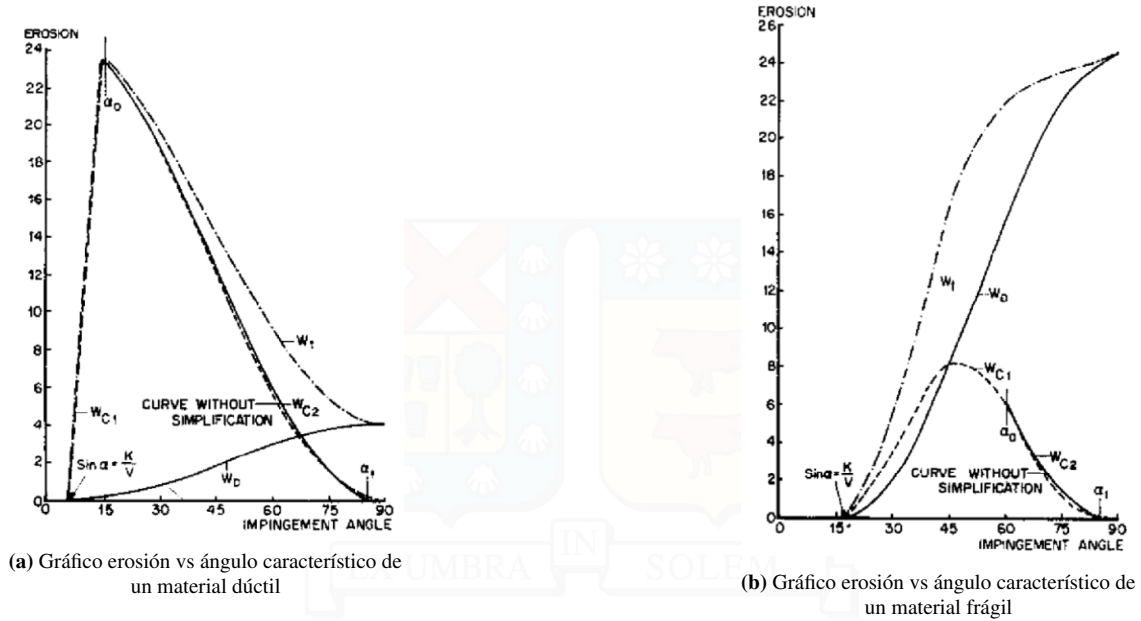


Figura 1.2: Gráficos erosión vs ángulo de impacto [14].

a cero son impacto deslizante u oblicuo como se puede apreciar en la Figura 1.3.

Desde los estudios realizados por Finnie [13], se plantea la velocidad de las partículas (en lo que respecta su magnitud y dirección) como uno de los factores claves en la erosión del material. En base a consideraciones energéticas, se argumenta que el potencial erosivo de una partícula es directamente proporcional con su velocidad de impacto, lo cual posteriormente se modificó como un coeficiente de elevación n , basado en observaciones experimentales. En la referencia [16], por medio de Ansys Fluent se simulaban 5 velocidades distintas manteniendo los demás parámetros constantes, y tal como se aprecia en la Figura 1.4, la profundidad de erosión aumenta a proporcionalmente con el aumento de la velocidad de entrada.

$$\text{Tasa de erosión} \propto (\text{Velocidad partícula})^n \quad (1.1)$$

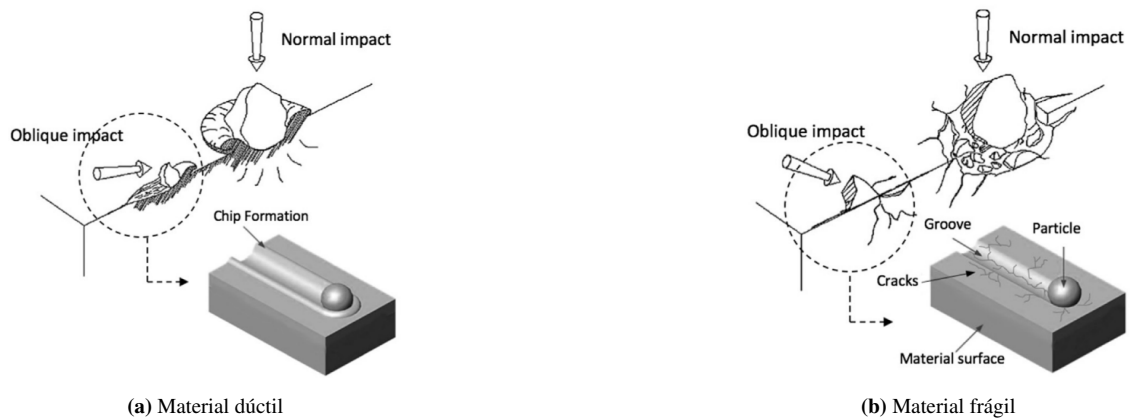


Figura 1.3: Efectos impacto sobre una superficie, respecto si el material es dúctil o frágil, y en relación al ángulo de impacto de la partícula.

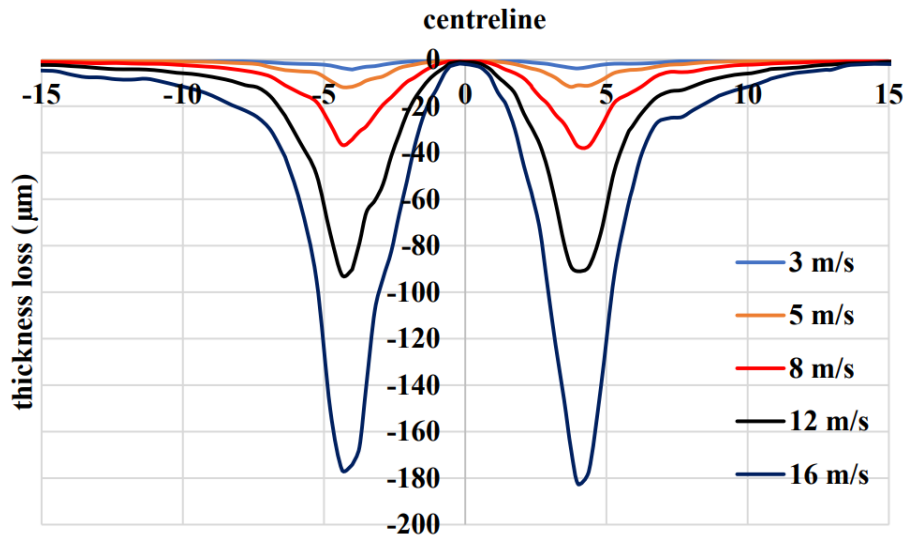


Figura 1.4: Comparación de profundidad erosión según distintas velocidades de entrada [16].

1.1.3. Propiedades partículas:

Las condiciones de la partícula pueden afectar el proceso de erosión directa o indirectamente. Se busca enfocarse en los efectos mas directos observando a la escala de impacto partícula-pared.

De acuerdo con las referencias [10] y [17] el tamaño de la partícula es directamente proporcional al ratio de erosión que produce, hasta alcanzar aproximadamente los 100 μm (Ver Figura 1.5), para luego mantenerse sustancialmente constante para diámetros mayores (lo que se considera un ejemplo de efecto directo). Las partículas de mayor tamaño tendrán una mayor energía cinética que una partícula más pequeña estando en las mismas condiciones de prueba y con la misma velocidad [10].

Por otro lado, un efecto indirecto del tamaño de la partícula se ve en la forma que interactúa con el flujo que la transporta. Se ha visto que el daño hecho por partículas de menor diámetro tienden a seguir las líneas de corriente de flujo, por lo que en una superficie plana con un fluido impactando a 90 grados, esta la posibilidad de que partículas más pequeñas impacten en una mayor área, pero sufrirán efectos de retardo en su velocidad [18], llegando incluso a haber una proporción de partículas que ni siquiera impacten el target debido al flujo [19].

Se habla también de como la forma geométrica de la partícula puede tener un efecto directo. En la referencia [20] se realizaron pruebas en acero AISI 1020 con partículas, las cuales fueron clasificadas en dos categorías principales: esféricas y angulares (más puntiagudas). Los resultados indicaba que las partículas angulares provocaban ratios de erosión mayores, también se discutió que las partículas esféricas crean cráteres mas superficiales y redondeados, mientras que partículas angulares desarrollan cráteres más filosos y profundos.

Finalmente, la dureza puede afectar directamente el potencial erosivo, ya que una con dureza menor que la del material del target, causaría menor desgaste comparado con partículas que son significativamente más duras [21]. Por otra parte, una partícula que tienda a quebrarse se produce lo que se conoce como fragmentación. La fragmentación conlleva a que se produzca lo que se conoce como erosión secundaria. En la referencia [10] se habla con mayor detalle de esta situación y se plantea una forma de poder calcularlo en base al tamaño de las partículas. Sin embargo, el autor Alejandro López en su tesis [22], en su revisión de literatura, plantea que no se espera que haya fragmentación cuando la velocidad de flujo se mantiene bajo 20 m/s, con partículas mas pequeñas que 500 μm , por lo que para esta memoria no se considera que ocurra esto.

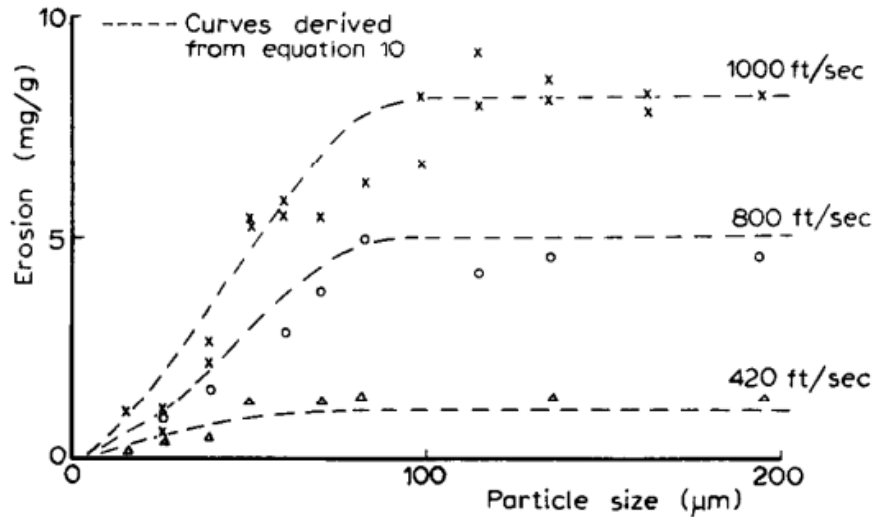


Figura 1.5: Relación erosión con tamaño de partícula con distintas velocidades [10].

1.1.4. Características mezcla:

Para modelar numéricamente y computacionalmente una mezcla, es importante conocer ciertas bases de los flujos multifásicos (o multifásicos). Los flujos multifásicos visto de manera simple, son fluidos que contienen 2 o más fases distintivas (sólido-líquido, líquido-gas, etc) fluyendo simultáneamente en mezcla y que tienen cierto nivel de separación de fase a una escala bien por encima del nivel molecular.

El primer paso para resolver cualquier problema multifásico es determinar cual régimen representa mejor su flujo. En la literatura es más común encontrar modelos de dos flujos que pueden estar completamente separados uno de del otro o ser flujos dispersos, que en otras palabras implica que hay una fase presentada en forma de partículas, gotas o burbujas dispersas en una fase continua transportadora (gas o líquida). En estos distintos flujos también son relevantes los mecanismos de transporte de interfaces.

Como una guía general, hay ciertos parámetros que ayudan a identificar el modelo multifásico apropiado como los siguientes:

- Concentración de partículas
- Flujos diluidos y densos
- Tiempo de respuesta (Número de Stokes)
- Acoplamiento de fase
- Otras consideraciones

En la literatura, siempre se considera una de las fases como CONTINUA y otra como DISPERSA. En mezclas líquido-sólido, la fase líquida sería la fase continua y transportadora, mientras que las partículas sólidas son la fase dispersa dentro de la fase continua. La concentración de partículas es la relación de volumen de partículas con volumen del fluido, que se puede ver reflejada en la fracción volumétrica que es la tasa de volumen de una las fases dividida por el volumen total de la mezcla, debido a esto la suma de las fracciones de ambas fases siempre debe ser igual a una unidad, lo cual, también se puede calcular en base a la masa de ambas fases. Si la concentración de partículas es baja, se conoce como flujo diluido, donde se consideran los efectos del flujo continuo sobre el flujo disperso. Por el contrario si la concentración de partículas se vuelve mayor, entonces se consideran también los efectos de partícula-partícula, y en casos más extremos, en lo que se conocen como flujos densos, se consideran más dominantes los efectos de las partículas sobre el fluido transportador [23].

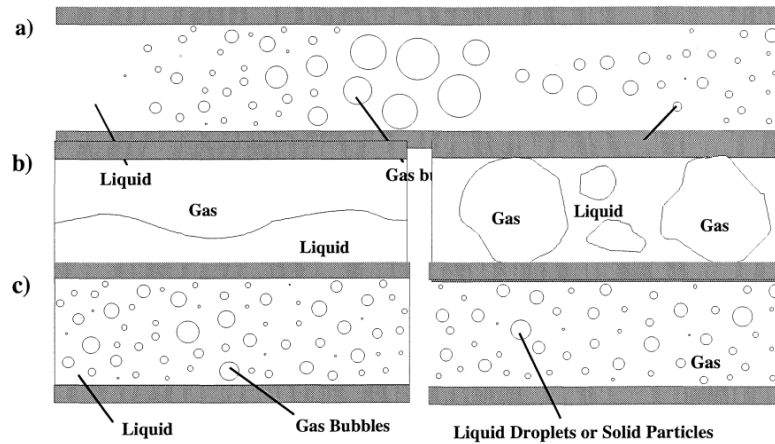


Figura 1.6: a) flujo de dos fases transitorio, b) flujo de dos fases separadas, c) flujo de dos fases dispersa.

(Fuente: [23])

Respecto a como influyen las características del flujo transportador en la erosión, es necesario entender como este influye en el movimiento de las partículas. La interacción entre la fase continua y la fase dispersa se da por las fuerzas de la primera sobre la segunda, estas pueden ser fuerza de arrastre, flotabilidad, fuerzas gravitatorias, diferencias de presión, etc.

En la referencia [19] se habla sobre los efectos del campo del instante, el campo del flujo dependerá de la geometría del material de muestra y la configuración en la que entran en contacto, dando como ejemplo el chorro de agua sobre una placa plana, donde las partículas teóricamente por inercia deberían seguir las líneas de campo, pero en la realidad existen fuerzas que pueden oponerse a su inercia, planteando que la fuerza de arrastre como la más influyente. Es por esto que la viscosidad y la densidad del fluido se vuelven relevantes en el perfil debido a que pueden variar la fuerza de arrastre sobre la partícula. Entonces, en el mismo artículo se concluye que las velocidades y ángulos de impacto sobre un material erosionado pueden diferir ampliamente de la velocidad y ángulo de las líneas de corriente, habiendo incluso casos de que algunas partículas ni siquiera impacten la superficie, por lo que, para entender mejor la pérdida de masa durante el proceso de erosión es necesario complementar modelos analíticos de la trayectoria de partículas con métodos experimentales de erosión.

1.2. Simulación computacional

En orden de entender el comportamiento de la erosión, primero hay que simular un flujo multifase, de manera que se pueda complementar con un modelo teórico de erosión. Como todo código de simulación requiere una validación, para este caso se tiene el artículo *CFD study of Jet Impingement Test erosion using Ansys Fluent and OpenFOAM*. Alejandro López, William Nicholls, Matthew T. Stickland, William M. Dempster [1], el cual será base para la metodología y parámetros computacionales y además el primer caso de validación para el código que se busca realizar.

Para este trabajo al igual que en la referencia [1] se quiere simular un flujo multifásico de tipo disperso. En otras palabras, un flujo continuo (agua) mezclado con un flujo disperso (partículas sólidas) de baja concentración en la mezcla, específicamente 1 % de fracción masa (lo que se traduciría en aproximadamente 0,4 % de fracción volumétrica). Como se mencionó en la sección anterior, la concentración de partículas influirá en como interactúan las partículas, lo que en términos de modelamiento y simulación se traduce como acoplamientos:

Acoplamiento unidireccional. Para baja concentración de partículas. Se puede no considerar la fuerza de las partículas

Acoplamiento bidireccional. Se incluye la fuerza de las partículas en las ecuaciones de flujo.

Acoplamiento cuatro direcciones. Para mayores concentraciones. Se considera interacción partícula-partícula.

Otro aspecto a considerar es que método numérico a utilizar para modelar las fases del flujo. Dos métodos comunes para el cálculo numérico de flujos multifásicos: enfoque Euler-Euler y enfoque Euler-Lagrange, siendo este último el utilizado por el autor y en este trabajo, lo que básicamente consiste en simular la fase continua y hacer un seguimiento lagrangiano de fase dispersa en base al equilibrio de fuerzas entre ambas fases. Al ser un acoplamiento unidireccional, se considera solamente la fuerza de arrastre sobre las partículas, las cuales se consideraron todas con el mismo tamaño y forma esférica, lo cual permite aplicar el modelo de coeficiente de arrastre para esta figura. Esto se explica con mas detalle en 2.4.1.

1.3. Modelos de erosión

Existen diferente modelos matemáticos de erosión en materiales, los cuales se han implementado en los softwares de CFD para poder predecir y estudiar los efectos que pueden haber en distintos materiales. Se considera el trabajo hecho por Alejandro López en *Comparative study of different erosion models in an Eulerian-Lagrangian frame using Open Source software* [8] para verificación y validación, donde se simula y compara cinco modelos de erosión (seleccionados arbitrariamente), de los cuales se destaca *Finnie erosion model*, ya que este se considera uno de los pioneros en cuanto este tema. Además, este modelo viene implementado por defecto en las versiones más actuales de OpenFoam y por consiguiente el que se utilizará en las simulaciones de este trabajo. Este modelo propuesto por I. Finnie en 1960 [13] básicamente plantea el volumen de material Q que es removido por un grano de masa m y velocidad V :

$$Q = \frac{mV^2}{p\psi K} \left(\sin 2\alpha - \frac{6}{K} \sin \alpha^2 \right) \quad \text{si} \quad \tan \alpha \leq \frac{K}{6} \quad (1.2)$$

$$Q = \frac{mV^2}{p\psi K} \left(\frac{K \cos \alpha^2}{6} \right) \quad \text{si} \quad \tan \alpha > \frac{K}{6} \quad (1.3)$$

Donde p es el esfuerzo plástico del material siendo erosionado, α es el ángulo de impacto, ψ es una constante que representa el ratio de profundidad de contacto con la profundidad de corte y K es el ratio entre los componentes de fuerza vertical y horizontal sobre la partícula.

La principal desventaja de esta ecuación se encuentra en la discrepancia entre los resultados experimentales y teóricos cuando las partículas impactan la superficie con incidencia normal. La ecuación de Finnie produce resultados precisos para ángulos de impacto bajos, pero no predice erosión producida por impactos normales a la superficie.

Se menciona también en la referencia [8] el modelo planteado por M. Hashish, quien modificó el modelo de I. Finnie, para erosión hecha por partículas sólidas en materiales dúctiles, tomando en cuenta la forma de la partícula e incluyendo constante no empíricas.

$$W = \frac{7}{\pi} \frac{M}{\rho_p} \left(\frac{V}{C_k} \right)^{2.5} \sin 2\alpha \sqrt{\sin \alpha} \quad (1.4)$$

$$C_k = \sqrt{\frac{3\sigma_f R_f^{\frac{3}{5}}}{\rho_p}} \quad (1.5)$$

Donde M es la masa de las partículas abrasivas, ρ_p es la densidad de las partículas, V es la magnitud de velocidad de las partículas, α es el ángulo de impacto, C_k es un coeficiente que se calcula en base a R_f ,

que es el factor de redondeo de la fase particulada. Este modelo es mas conveniente para ángulos de impacto poco profundos en materiales dúctiles.

Por otro lado, en la referencia [8], se menciona el modelo de erosión de *Grant & Tabakoff* el cual se basa en la suposición de que la erosión depende de dos mecanismos, uno actuando en ángulos de impactos bajos y otros en ángulos mas altos (cercanos a 90 grados), como también una combinación de ambos cuando los impactos ocurre aproximadamente en una zona intermedia a estos ángulos.

$$\epsilon = K_{10}f(\beta_1)(V_{1T}^2 - V_{2T}^2) + f(V_{1N}) \quad (1.6)$$

Donde, ϵ es la erosión por unidad de masa de partículas impactantes, K_1 es el material en contacto, $f(\beta_1)$ es una función empírica del ángulo de impacto de la partícula, V_{1T} es la componente tangencial de la velocidad de partícula entrante, V_{2T} es la componente tangencial de la velocidad de partícula rebotante.

1.4. Pruebas experimentales de erosión

Se han diseñado distintos aparatos que permiten investigar los efectos de uno o un par de los parámetros mencionados anteriormente. Sin embargo no se ha visto que todos los parámetros estén completamente controlados, ya que se requeriría de un modelo que permita controlar y configurar todos los factores de efecto en la erosión, lo cual sería algo demasiado complejo de alcanzar de forma realista.

Revisando en la literatura, no abundan muchas normas estándar de diseño de bancos de prueba. Cabe mencionar que en la referencia [9] se nombran las normas ASTM G26 y ASTM G73, en esta ultima solo se entra en detalle de cuando la mezcla del flujo es gas-sólido. Otro aspecto importante de mencionar es que existen dos configuraciones de pruebas para analizar la erosión, uno es el "pot tester" que básicamente consiste en sumergir un material de muestra en un baño de un fluido con partículas sólidas suspendidas. El material de muestra se hace girar en el líquido para que esta entre en contacto con las partículas erosivas.

La otra configuración es el "Slurry jet impingement" que forzar el flujo particulado por una tobera dirigiendo el flujo hacia un material de muestra. Este método se caracteriza por permitir un control mas preciso sobre la velocidad y ángulo de impacto de las partículas. Este tipo de configuración es el que usará para esta investigación y simulación.

Los autores S. Karthik y H. J. Amarendra [3] hicieron una recopilación y revisión de estudios y experimentos relacionados con "*slurry jet erosion test rig*", planteando una clasificación de 4 tipos de métodos principales, los cuales son nombrados con el apellido de cada autor considerado pionero en su uso. Se explica además, que los bancos de pruebas se pueden clasificar como tipo re-circulación (RC) y sin re-circulación (NRC), dependiendo de si se reutilizan las partículas en el flujo principal después de haber impactado la muestra (esta clasificación también es mencionada en la referencia [9]). La otra forma de clasificar los bancos de pruebas, es de tipo pre-mezclado y tipo post-mezclado, lo cual se refiere a que si se mezclan las partículas en el flujo principal antes de ser presurizado por la bomba o después de ser bombeado por esta.

En la Figura 1.7 se pueden apreciar las 4 categorías según el su método de mezcla y denominadas por su respectivo autor:

a) Levy: Consiste en un estanque que contiene la mezcla fluido-partícula junto con un agitador que mantenga las partículas suspendidas en el fluido (en este caso se uso agua y keroseno), por un lado ingresa al estanque aire comprimido el cual le da la presurización necesaria al fluido para salir por la tobera hacia el material de prueba. En este mismo artículo se menciona que es el método menos popular (3,03 %) de la recopilación hecha por los autores.

b) Turenne: Se utiliza una bomba especializada en flujos con partículas (*slurry pump*) para presurizar un flujo previamente mezclado y conducido luego al material de prueba. Este método es el más

popular, representando un 54,55 % de la recopilación hecha por los autores, dando a entender que para otros investigadores se considera más sencillo adoptar las características de re-circulación (RC) y de pre-mezclado para sus diseños.

- c) Hutchings:** Consiste en utilizar una bomba multi-etapas para conducir el flujo principal de agua hacia una tobera y por consiguiente hacia el material de muestra. Antes de salir el flujo por la tobera, hay una región de baja presión creada por la disminución de la sección transversal, estando esta zona conectada a un eyector permite que la diferencia de presión succione las partículas erosivas mezcladas con agua en un estanque.
- d) Thapa:** Se basa en una investigación hecha por B. Thapa, cayendo la categoría de sin re-circulación y pos-mezcla. Básicamente consiste en que una bomba transporte el flujo principal de agua hacia el material, teniendo una tolva ubicada a lo largo de la línea de flujo principal, la cual contiene las partículas erosivas y que el flujo másico de partículas se regula variando la apertura del orificio de la válvula. Sin embargo, se explica que hay poca información acerca del principio empleado para la introducción de partículas. El autor B. Thapa describe superficialmente que una forma de que las partículas deberían ingresar al flujo por efecto de la gravedad y otra forma es utilizar dos tolvas ubicadas una debajo de la otra, la primera con partículas cayendo a la segunda tolva con agua y luego deberían ser llevadas por el fluido. El autor John Sandstorm utiliza este método en su diseño, pero utilizó tolvas diseñadas por el mismo, las cuales bombean las partículas al flujo principal por medio de cilindros hidráulicos, mientras que otros autores no dan mayores detalles del principio empleado.

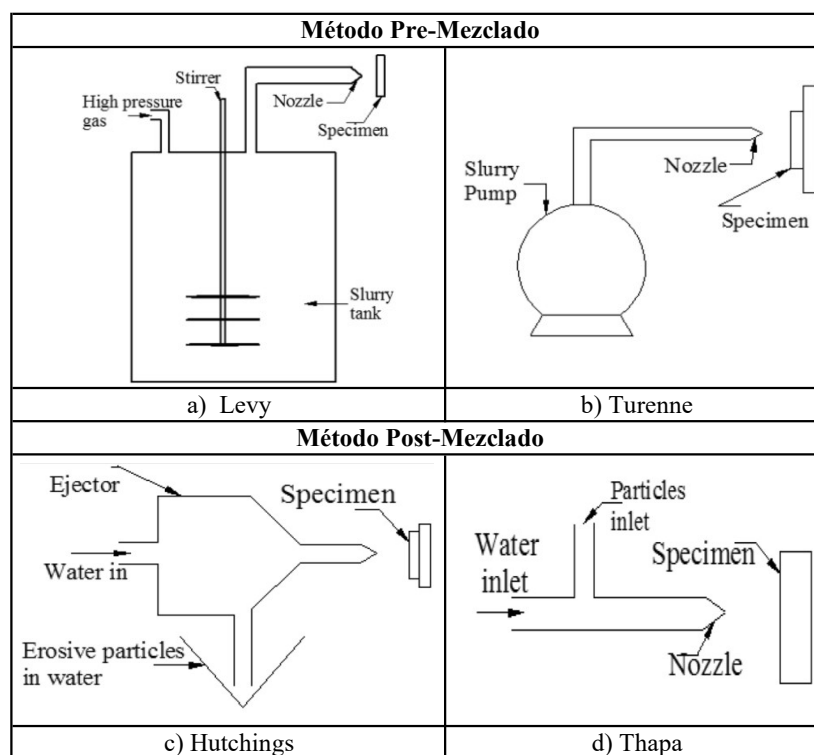


Figura 1.7: Clasificación de banco de prueba según su método de mezcla

(Fuente: Adaptado de [3])

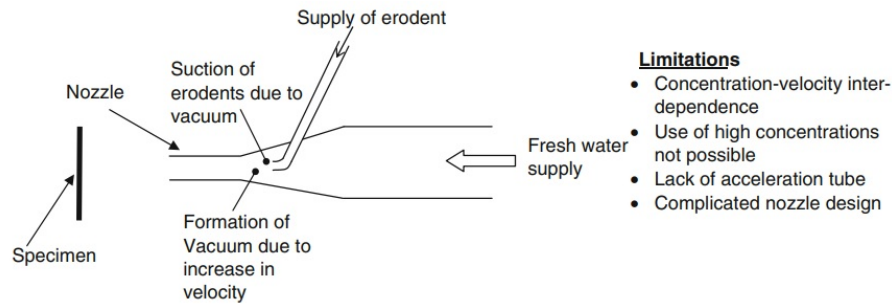
Como ejemplo de la categoría de Turenne se puede mencionar A. Mansouri (2015) [24, 25, 26], Gnanavelu (2009) [27, 28], Nguyen (2014) [29, 30, 31], por lo que se puede afirmar que es un método útil y teóricamente sencillo de replicar. No obstante, existe un problema respecto a la reutilización de las partículas luego de que estas golpean la superficie. Este problema es comentado por el autor H.S. Grewal en su artículo DESIGN AND DEVELOPMENT OF HIGH-VELOCITY SLURRY EROSION TEST RIG USING CFD

(2012) [32] donde afirma que las partículas se fracturan, desintegran y pierden redondeo haciendo que no se mantengan las condiciones impacto y además se pierde control sobre la concentración de sólidos en el flujo. Por lo que recomienda utilizar la configuración de no recirculación (NRC) en los bancos de pruebas.

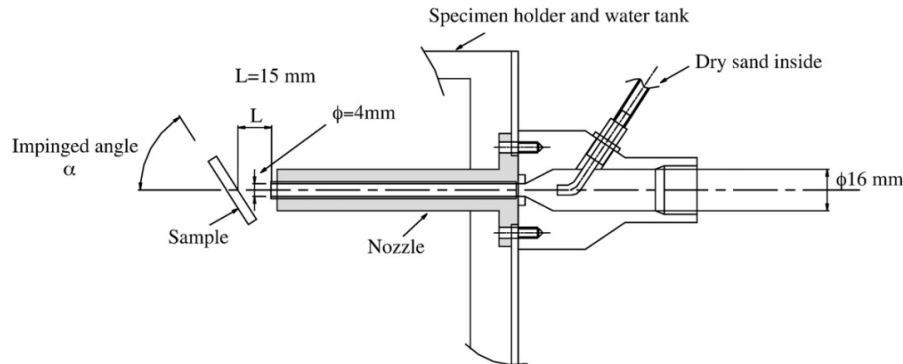
Sin embargo, el diseño de los bancos de prueba de tipo NRC, son más complejos de diseñar y/o de operar, junto con esto, se comenta también en la referencia [32] *“Algunos estudios pueden ser encontrados en la literatura, donde bancos de tipo NRC han sido utilizados para estudiar la respuesta a la erosión de materiales y cubrimientos; sin embargo un diseño detallado no pudo ser encontrado, limitando la posibilidad de reproducirlos efectivamente.”*. En este mismo artículo se describe un método común de inyección de partículas al flujo principal de agua, que consiste en instalar un tubo que se conecte al flujo principal, ubicándose en una reducción de sección transversal, por lo que siguiendo el principio de Bernoulli, se produce un aumento de velocidad y una caída de presión. Si la presión disminuye a un valor menor que la presión atmosférica, se succionarán las partículas al flujo principal. Este método presenta las siguientes dificultades:

- i) Un ejemplo de este método se ve en la Figura 1.8. Donde aprovecha la reducción de sección transversal de la tobera antes de la salida del chorro hacia el target para inyectar las partículas. El problema con esto, como señala H.S Grewal [32] es que se requiere de una cierta distancia antes de la salida para que las partículas puedan acelerar y mezclarse con las líneas de corriente del flujo líquido, por lo que no es recomendable instalarlo en la tobera.
- ii) Otro ejemplo de este método se ve en la Figura 1.9 y fue puesto a prueba por Alejandro López en su tesis [22]. Consistía en instalar un tubo de Venturi hecho en una impresora 3D (Figura), que permitiera inyectar partículas al flujo después de que este saliera de la bomba. No obstante, su intento de utilizarlo en un banco no fue exitoso debido a que se producía contra-presión provocando fuga de agua en vez de ingreso de partículas
- iii) En ambas circunstancias anteriores, como se señala en la referencia [32], existe una co-dependencia entre la concentración y la velocidad del flujo, lo que limita mucho las pruebas que puedan hacerse. Es decir, el diseño solo permite ingresar un pequeño rango de concentración de sólidos según las dimensiones de la reducción de sección transversal y el caudal del flujo principal, e incluso habiendo situaciones, en las que como lo indica A.Mackenzie en su tesis (refiriéndose al diseño de tubo de Venturi de Alejandro Lopez) se requieren de altas velocidades de flujo para producir la diferencia de presión suficiente para que haya ingreso de partículas en vez de fuga de agua.

Entonces, tanto H.S. Grewal (Figura 1.10), como A. Lopez, y A. Mackenzie optan por usar una bomba centrífuga menos costosa que sirva como "bomba sacrificio" para inyectar una mezcla sólido-líquido al flujo principal. En el caso de A. Lopez, realiza la inyección cerca de la salida de la tobera, pero como se explicó anteriormente, conviene que la inyección sea antes como lo hacen H.S. Grewal y A. Mackenzie para que las partículas se mezclen y adapten como flujo disperso dentro del flujo continuo. Para conseguir esto es necesario que el flujo disperso este a la misma presión que la del flujo continuo para que se mezclen en la tubería antes de salir por la tobera. Otro aspecto importante a considerar es el ángulo en el que las partículas ingresan, es por esto que H.S. Grewal propone complementar el diseño del banco con simulaciones computacionales para determinar que ángulo de entrada conviene más para las condiciones de prueba.



(a) Diagrama de jet con succión de material aprovechando el aumento de velocidad en tobera [32]



(b) Diagrama de jet con succión de material aprovechando el aumento de velocidad en tobera [33]

Figura 1.8: Ejemplos de uso de reducción de sección transversal de tobera para producir succión de mezcla con material sólido

1.4.1. Recolección de datos

En esta sección se hablará brevemente de métodos de obtención de datos de pruebas experimentales según lo visto en la literatura.

En primer lugar, enfocándose solamente en el material de prueba, se tiene la práctica de pesar esta pieza antes y después de que el proceso haya sido realizado. Esto debe realizarse con una balanza precisa y con resolución de miligramos, para estimar la cantidad de material que se ha perdido.

Luego, para obtener una imagen del perfil de erosión a lo largo del diámetro de la pieza (asumiendo que se utiliza una placa plana con forma de disco), se puede utilizar un perfilómetro óptico, tal como se menciona en la referencia [24]. Como instrumento opcional podría ser el palpador disponible en el laboratorio de mediciones de la facultad, ya que aplica el mismo principio de esta pieza fina recorra la superficie y registrando las hendiduras sobre el material.

En segundo lugar, existen dos métodos para visualizar los campos de velocidad del flujo y las partículas:

Velocimetría de láser doppler: Otra técnica no invasiva para medir la velocidad de un fluido. La luz del láser ilumina el fluido, y la luz reflejada de las partículas sembradas en el flujo es recogida y procesada. En la práctica, un sólo haz del láser es dividido en dos haces de igual intensidad que se enfocan en un punto común en el campo de flujo. En el punto donde los haces se cortan se forma un modelo de interferencia, definiendo el volumen de medición. Las partículas que se mueven a través del volumen de medición dispersan luz de intensidad variable que es recolectada con un fotodetector. La frecuencia de salida resultante del fotodetector está directamente relacionada con la velocidad de la partícula.

Sistema de velocimetría de partículas: Por sus siglas en inglés PIV (Particle Image Velocimetry), es una técnica de medición que permite obtener el campo de velocidades de flujos de manera instantánea

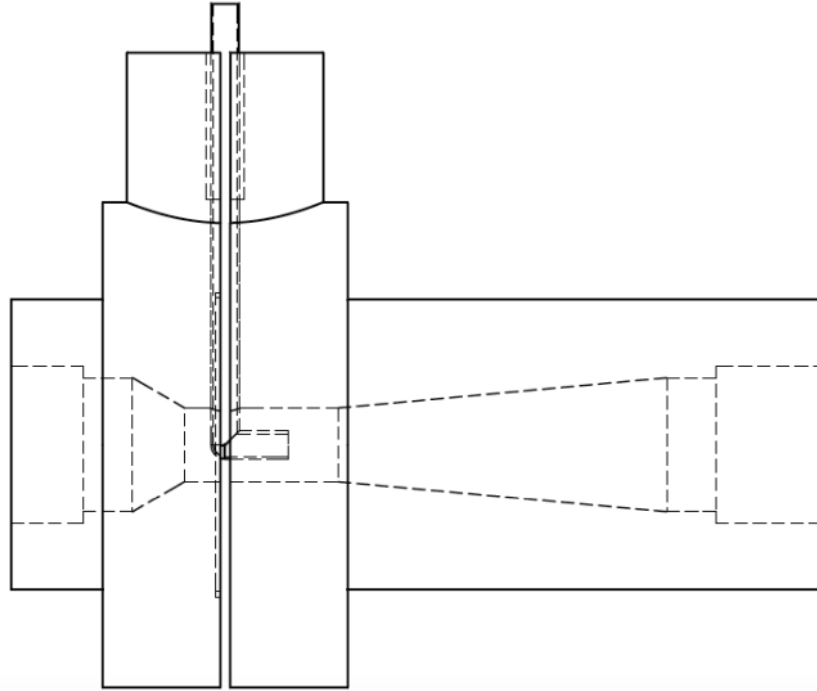


Figura 1.9: Layout Tubo de Venturi diseñado para ingresar material sólido al flujo propuesto por A. Lopez

(Fuente: Obtenido de [22])

y con alta resolución espacial. Esta técnica no es intrusiva, lo cual permite realizar mediciones sin perturbar el flujo. La técnica se basa en determinar sobre la base de dos imágenes capturadas en un intervalo de tiempo conocido, el desplazamiento de las partículas que son sembradas en el flujo. El desarrollo de esta técnica en los últimos años ha permitido su aplicación en la caracterización de flujos complejos logrando un avance notable en el campo de la mecánica de los fluidos.

El método LDV (Laser Doppler Velocimetry) se nombra en la referencia [24], es y fue utilizado de forma efectiva por el autor Y. Zhang [34], al medir la velocidad de las partículas sólidas entre la salida de la tobera y el target y obteniendo resultados muy similares en CFD. Sin embargo, el autor A. Mackenzie en su tesis [35], comenta que el método LDV tiene mayor dificultad de medir la velocidad de las partículas cerca o sobre la superficie del target, debido a la interferencia de la pared con el volumen de medición que puede alcanzar el láser de este método, mientras que el método PIV aunque tiene sus dificultades tiene mayores posibilidades de realizar esta medición.

Por otro lado, el método PIV es comúnmente usado para medir y visualizar las líneas de corriente del fluido en movimiento, por lo que las partículas que vienen mezcladas deben ser lo suficientemente pequeñas y livianas para que sigan las líneas de corriente. Sin embargo, si se busca medir la velocidad de partículas de arena o similares, las cuales es muy probable que no seguirán por completo la dirección que tome el flujo, en especial después de impactar contra la superficie del target.

Por lo que, tal como se menciona en las tesis de A. Mansouri [25] y de A. Mackenzie [35], el método PIV es descrito como un método efectivo no solo para poder comparar los perfiles de velocidad del flujo continuo con el obtenido de las simulaciones hechas por su respectivo autor, sino que también para rastrear las partículas diluidas en un slurry flow (por ejemplo arena), pudiendo estimar la velocidad y ángulo promedio con la que impactan una superficie, pero haciendo la distinción de que se requiere aplicar la técnica de Particle Tracking Velocimetry (PTV) que es una forma de postprocesamiento de las imágenes obtenidas con el equipamiento del método PIV.

En este informe no se entrará en detalle en el uso y funcionamiento de esta técnica ya que como se ha dicho antes el diseño del banco de prueba es solo conceptual, por lo que basta con una explicación

superficial de su uso.

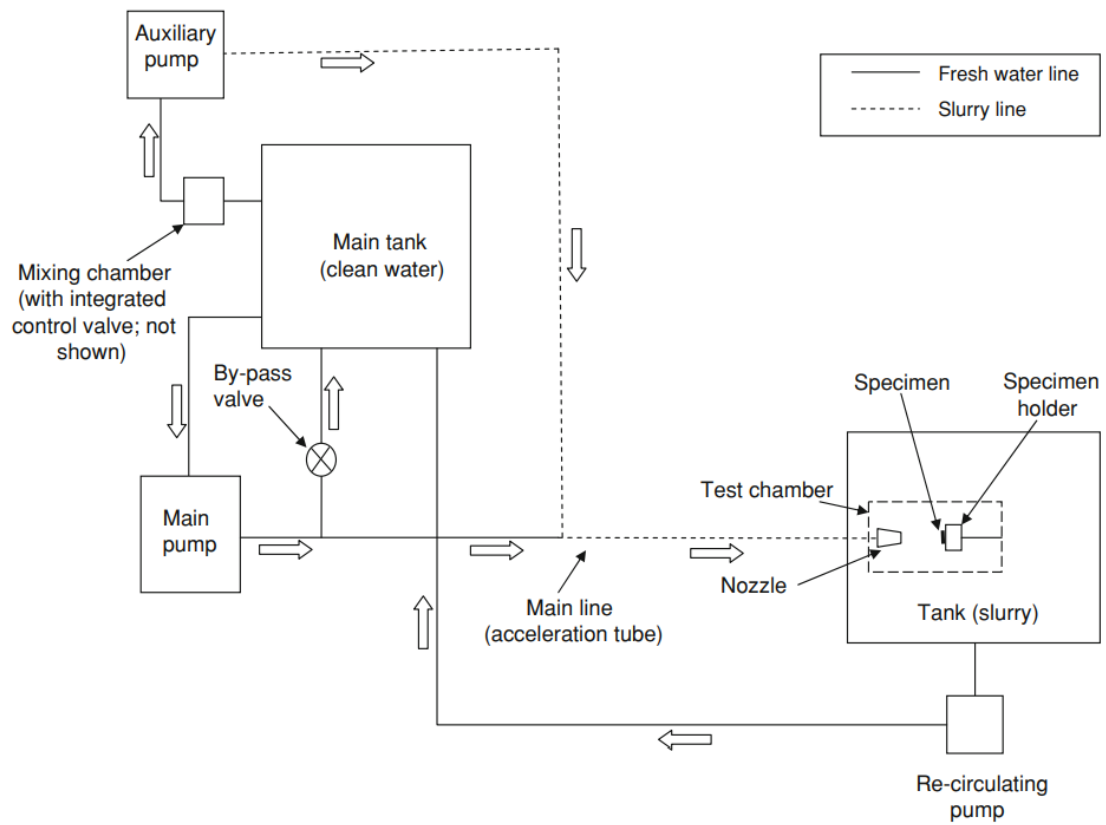


Figura 1.10: Layout de propuesta de H.S. Grewal

(Fuente: Adaptado de [32])

2 | Marco Teórico

2.1. Método de volúmenes finitos

El método de volúmenes finitos, por sus siglas en inglés FVM, se define en la referencia [36] como técnica numérica que toma las ecuaciones diferenciales parciales que representan las leyes de conservación sobre volúmenes diferenciales y las transforma a ecuaciones algebraicas discretas sobre volúmenes finitos (o elementos, o celdas). Estos volúmenes finitos son productos de la discretización de un dominio geométrico, las ecuaciones diferenciales parciales se transforman en ecuaciones algebraicas integrándolas sobre cada elemento discreto. El sistema de ecuaciones algebraicas es entonces resuelto para computar los valores de la variable dependiente para cada uno de los elementos.

En este método, algunos de los términos en la ecuación de conservación son convertidos en flujos faciales y evaluados en los bordes del volumen finito, ya que el flujo entrando a un volumen dado es idéntico al que está saliendo del volumen adyacente. El FVM es estrictamente conservativo y puede ser formulado en el espacio físico en mallas poligonales no estructuradas. Además, en este método es relativamente sencillo implementar condiciones de borde de una forma no invasiva, ya que las variables desconocidas son evaluadas en los centroides de los elementos volumétricos y no en las caras. Estas características hacen del FVM conveniente para la simulación numérica de una variedad de aplicaciones involucrando flujo de fluido, transferencia de calor y/o masa.

2.1.1. Ecuaciones gobernantes

La base para modelar matemáticamente el comportamiento de la gran mayoría de los fluidos son las ecuaciones agrupadas en las ecuaciones de Navier-Stokes: continuidad, conservación de momentum y conservación de energía. A lo que se le agrega la ecuación de estado que es fundamental en la termodinámica.

$$\frac{Dm}{Dt} = 0 \quad (2.1)$$

$$\frac{\delta \rho}{\delta t} + \nabla \cdot (\rho V) = 0$$

$$\frac{Dp}{Dt} = \frac{D(\rho V)}{Dt} = \sum F \quad (2.2)$$

$$\frac{\delta V_i}{\delta t} + V_j \frac{\delta V_i}{\delta x_j} = -\rho g_i + \frac{\delta \tau_{ji}}{\delta x_j}$$

$$\frac{DE}{Dt} = \frac{DQ}{Dt} + \frac{DW}{Dt} \quad (2.3)$$

$$\rho \frac{De}{Dt} = -\nabla \cdot q - p(\nabla \cdot V) + \frac{\delta Q}{\delta t}$$

$$p = p(\rho, T) \quad \text{Caso Euler (no viscoso)} \quad p = \rho RT \quad (2.4)$$

2.2. Modelamiento de flujo multifásico

Como se mencionó en estado del arte, ciertas características necesitan ser revisadas a la hora de modelar y simular un flujo multifásico, como lo es por ejemplo la fracción volúmetrica α_f , que en la referencia [23] se expresa como:

$$\alpha_f = \frac{\delta Vol_f}{\delta Vol_{Total}} \quad (2.5)$$

Donde Vol_f es el volumen de la fase dentro de una celda o dominio, y Vol_{total} es el volumen total de la celda o del dominio. Si se tienen dos fases la suma de sus fracciones volúmetricas siempre debe ser igual a una unidad, $\alpha_{continua} + \alpha_{dispersa} = 1$.

En la referencia [23] se explica también el cargamento de partícula (particulate loading β), el cual se define como el ratio de densidad entre el la fase dispersa (d) y la fase continua (c). Ratio de densidad material, se calcula dividiendo solo la densidad de la fase dispersa por la densidad de la fase continua. Si es mayor que 1000, implica flujo gas-sólido, alrededor de 1 es para flujos líquido-sólido, y menos de 0,001 implica flujo gas-líquido.

$$\beta = \frac{\alpha_d \rho_d}{\alpha_c \rho_c} \quad (2.6)$$

$$\gamma = \frac{\rho_d}{\rho_c} \quad (2.7)$$

Se puede clasificar el flujo como denso o diluido. El flujo diluido es donde el movimiento de la partícula es controlado por la fuerzas de la fase continua, mientras que el flujo denso predominan las colisiones de las partículas en el control de su movimiento. Como se aprecia en la referencia [23], determinar el tipo de flujo es relevante al momento de considerar de seleccionar método de simulación, destacando Euler-Euler, es decir considerando las partículas sólidas como otro fluido más interactuando con el fluido transportador, la cual es más común para flujos densos. Mientras que por el otro lado, el método Euler-Lagrange (rastreo de partículas) es más común para flujos diluidos, es decir baja fracción volúmetrica de sólido, ideal menor a 10 por ciento.

El tiempo de respuesta τ_p , donde d_d es el diámetro de la partícula (asumiendo que tiene forma esférica) y μ_c es la viscosidad dinámica de la fase continua. Por otro lado τ_f es el tiempo característico del fluido, obtenido por el largo característico (L_F) de este dividido por la velocidad de este (V_F).

El número de Stokes es el ratio entre el tiempo de respuesta y el tiempo característico y permite observar si las partículas alcanzan a responder a los cambios en la velocidad del fluido.

$$\tau_p = \frac{\rho_d d_d^2}{18\mu_c} \quad (2.8)$$

$$\tau_f = \frac{L_F}{V_F} \quad (2.9)$$

$$St = \frac{\tau_p}{\tau_f} \quad (2.10)$$

Estas ecuaciones sirven para caracterizar la capacidad de las partículas para seguir cambios repentinos en el flujo continuo. En la literatura y en estudios similares relacionados a la erosión usando JIT, como en la

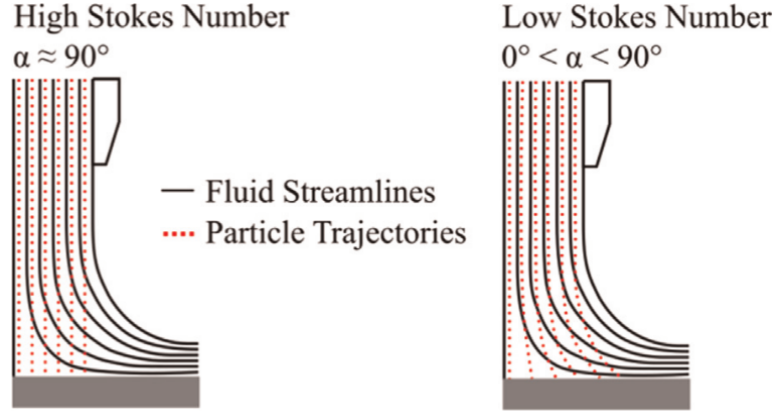


Figura 2.1: Ilustración de número de Stokes [37]

referencia [37], se afirma que para valores altos de número de Stokes, la inercia de las partículas domina sobre los efectos del fluido, por lo que pueden hasta considerarse insignificantes en el movimiento de la fase discreta. Mientras que por el otro lado, valores bajos de este número ($St \ll 1$) las partículas tenderán a seguir la dirección del flujo continuo. Esto aplicado al caso de un chorro de agua-partícula incidiendo en un ángulo de 90° sobre una superficie plana, como se ve en la Figura 2.1, para valores altos de St , el flujo de partículas tenderá a mantenerse en cercano a este ángulo, mientras que en valores bajos pero no menores que 1, las partículas incidirán sobre la superficie con un ángulo que varía entre 0° y 90°

2.3. Modelamiento de turbulencia

Los flujos turbulentos se explican en la referencia [38] como flujos con un movimiento irregular con distintas escalas, que posee variaciones muy aleatorias tanto en tiempo como en espacio, por lo que se pueden obtener distintos valores promedios de un mismo flujo turbulentos.

Debido a que es un fenómeno tan complejo, se han formulado teorías y modelos que puedan explicar y/o predecir su funcionamiento. Se destaca la teoría conocida como "cascada de energía" (presente en la referencia [39], donde de manera simple se explica que en los flujos turbulentos ocurren oscilaciones en los campos de velocidad y de presión que dependen del tiempo. Los vórtices más grandes transmiten energía a los más pequeños, mientras que estos disipan la energía cinética turbulenta como calor.

Para estimar las escalas más pequeñas donde se pueden presentar vórtices son denominadas escalas de Kolmogorov, las cuales en la referencia [39] se expresan como:

$$\eta = \left(\frac{\nu^3}{\epsilon} \right)^{1/4} \quad ; \quad t_\eta = \left(\frac{\eta}{\epsilon} \right)^{1/2} \quad ; \quad u_\eta = (\nu \epsilon)^{1/4} \quad (2.11)$$

donde η es la micro-escala de longitud, t_η es la micro-escala de tiempo, u_η es la micro-escala de velocidad, ν es la viscosidad cinemática y ϵ es la tasa de disipación de calor. A partir de esta teoría se basan los modelos de turbulencia mas utilizados actualmente en CFD.

En la referencia [39], se define como Direct Numerical Simulation (DNS), como el método más preciso y cercano a simular experimentos analíticos de fenómenos de turbulencia, sin embargo, para la resolución de una simulación, la malla y el avance en el tiempo deben ser lo suficientemente finas para poder tener en cuenta hasta los más pequeñas oscilaciones que ocurren dentro del flujo. Por consiguiente, debido a la complejidad de los cálculos, y principalmente al gran costo computacional (y económico) que llevaría simular este tipo de flujo (incluso si no son tan complejos) es que se han creado otros modelos de turbulencia que busquen acercarse lo más posible a una solución realista a un menor costo computacional

y/o económico.

Retomando el concepto de energía cinética turbulenta generada en grandes escalas y disipándose en escalas menores, por medio de análisis dimensional se puede encontrar relación entre la disipación y las variables de las grandes escalas:

$$\epsilon \sim \frac{u}{l} \quad (2.12)$$

donde l es la longitud característica de los grandes vórtices. Entonces juntando esto con la ecuaciones anteriormente planteadas se llega a:

$$\begin{aligned} \eta &= \left(\frac{\nu^3}{\epsilon} \right)^{1/4} \sim \left(\frac{\nu^3 l}{u^3} \right) = l \left(\frac{\nu^3}{u^3 l^3} \right)^{1/4} \\ \Rightarrow \frac{\eta}{l} &\sim \left(\frac{\nu}{ul} \right)^{3/4} = Re^{-3/4} \end{aligned} \quad (2.13)$$

Esto entrega una nueva interpretación del número de Reynolds (el cual normalmente se presenta como el ratio entre fuerzas inerciales y fuerzas viscosas), como el ratio entre las pequeñas y grandes escalas. Este resultado se puede aplicar en la simulación numérica de fluidos, ya que permite estimar el número de nodos que se necesitarían. Por ejemplo, si la escala del problema es 1 [m] en una dimensión, el número de nodos que se necesitarían es $1/\eta$, y si se considera una simulación en 3 dimensiones, entonces el número de nodos se estima con $N = Re^{9/4}$. Entonces, un cuerpo cuya longitud característica se estima como $l \sim 2$ [m], moviéndose en aire ($u \sim 5$ [m/s] y $\nu \sim 10^{-5}$ [m²/s]) da un $Re \sim 10^6$, es decir $N \sim 10^{13}$ nodos. En consecuencia, para guardar la posición de cada nodo en la malla se requieren 3 valores (x, y, z), si cada valor requirieran 4 bytes, esto significaría que $16 \cdot 10^{13}$ bytes son necesarios, en otras palabras, se ocuparían 160 TBytes solo para guardar posiciones, sin contar los datos de las demás variables que se quieran obtener. Por lo tanto, debido a que se busca reducir los costos computacionales (y aprovechando que muchos problemas ingenieriles no requieren de tanta precisión en cuanto a simular turbulencia) se han creado distintos modelos de turbulencia de donde destacan el modelo LES, DES, RANS ($\kappa - \omega$, $\kappa - \epsilon$), entre otros.

De las Ecuaciones RANS (Reynolds Averaged Navier Stokes) se puede destacar el modelo $\kappa - \epsilon$, el cual es mencionado en las referencias [23] y [39], como un modelo útil en la simulación de flujos multifásicos debido a su eficacia y mucho menor costo computacional en comparación a DNS. El modelo $\kappa - \epsilon$ es de orden 2, es decir contiene dos ecuaciones diferenciales para realizar el cierre de la turbulencia. Una para la energía cinética turbulenta κ y otra para la disipación de energía turbulenta ϵ . La ecuación para la disipación de energía turbulenta esta dada por:

$$\frac{\partial \kappa}{\partial t} + U_i \frac{\partial \kappa}{\partial x_i} = \frac{\partial}{\partial x_i} \left(\frac{\nu_t}{\sigma_\kappa} \frac{\partial \kappa}{\partial x_i} \right) + \nu_t \left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) \frac{\partial U_i}{\partial x_j} - \epsilon \quad (2.14)$$

$$\frac{\partial \epsilon}{\partial t} + U_i \frac{\partial \epsilon}{\partial x_i} = \frac{\partial}{\partial x_i} \left(\frac{\nu_t}{\sigma_\epsilon} \frac{\partial \epsilon}{\partial x_i} \right) + c_{1\epsilon} \nu_t \left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) \frac{U_i}{\partial x_j} - c_{2\epsilon} \frac{\epsilon^2}{\kappa} \quad (2.15)$$

$$\nu_t = c_\mu \frac{\kappa^2}{\epsilon} \quad (2.16)$$

Donde ν_t es la viscosidad turbulenta, σ_κ con σ_ϵ son coeficientes de Schmidt turbulento para κ y ϵ respectivamente, mientras que C_μ , $C_{1\epsilon}$ y $C_{2\epsilon}$ son constantes empíricas que no son universales, sino que dependen del flujo o modelo con el que se complementa, no obstante, los valores más comunes a utilizar por defecto por los simuladores computacionales, incluyendo OpenFOAM son:

Tabla 2.1: Constantes empíricas del modelo turbulento (adaptado de la referencia [23]).

C_μ	$C_{1\epsilon}$	$C_{2\epsilon}$	σ_κ	σ_ϵ
0,09	1,44	1,92	1,92	1,3

2.4. Modelo partícula discreta (DPM)

La simulación del estudio base fue hecha en la versión 2.x de OpenFOAM, utilizando el solver *pimpleFoam* acoplado con un diccionario de *lagrangian particle tracking*. Versiones más actuales de OpenFOAM tienen el solver *DPMFoam*, donde las siglas DPM significan Discrete Particle Model, (modelo partícula discreta), en la que su algoritmo ya viene con el enfoque Euler-Lagrange implementado. Como se aprecia en la Figura 2.2 Se resuelve la fase continua representada en el flujo de agua, mediante las ecuaciones de Navier-Stokes (enfoque euleriano), que en el caso de ser una simulación de un flujo transitorio, se realiza un seguimiento de los movimientos de la partícula (fase discreta) al final de cada paso de tiempo acoplando con el enfoque lagrangiano el flujo continuo calculado.

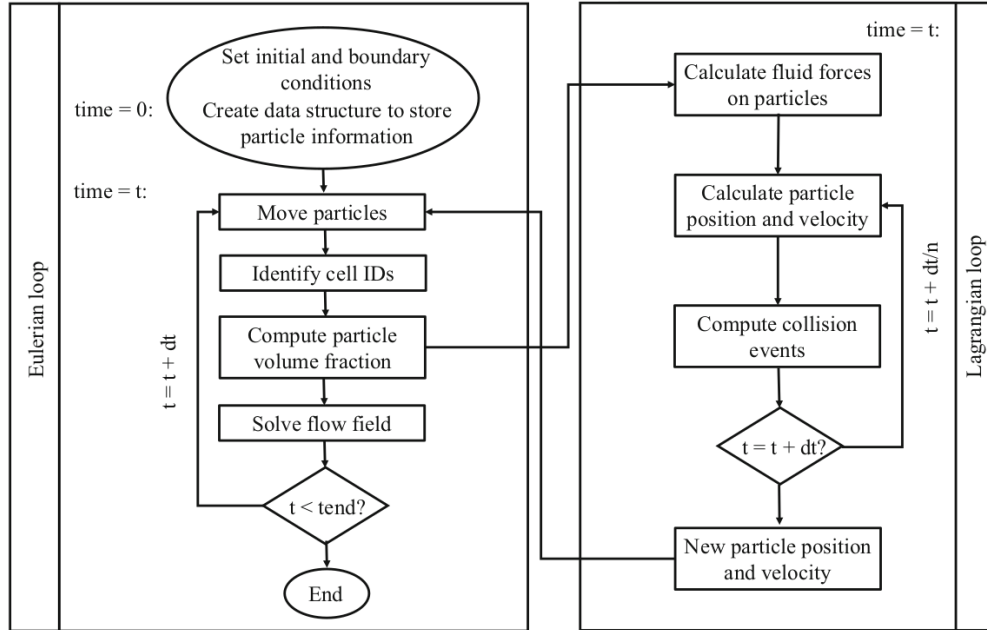


Figura 2.2: Algoritmo Euler-Lagrange [40].

2.4.1. Lagrangian Particle Tracking

Para formar las líneas de corriente, se resuelve la ecuación diferencial para actualizar la posición de la partícula X_p [41]:

$$\frac{dX_p}{dt} = U_p \quad (2.17)$$

Donde U_p es la velocidad de la partícula.

Si las partículas no tuvieran masa, se puede asumir que la velocidad de la partícula es la misma que la velocidad local del flujo continuo $U_p = U$. Entonces, por consiguiente, las partículas seguirían la misma trayectoria que las líneas de corriente del campo del flujo continuo. No obstante, las partículas si tienen masa por lo que $U_p \neq U$ y en consecuencia, es necesario hacer un balance de fuerza para calcular U_p .

Entonces, se resuelve la ecuación explícita de Euler del siguiente paso de tiempo [41]:

$$\frac{dX_p}{dt} = U_p \quad \longrightarrow \quad \frac{X_p^{i+1} - X_p^i}{\Delta t}$$

$$\Rightarrow X_p^{i+1} = X_p^i + U_p^i \cdot \Delta t \quad (2.18)$$

Luego para el balance de fuerza, se aplica la segunda ley de Newton:

$$F_p = m_p \cdot a \quad \longrightarrow \quad F_p = m_p \frac{dU_p}{dt} \quad (2.19)$$

Donde m_p es la masa de la partícula y F_p es la suma de todas las fuerzan que pueden afectar la trayectoria de las partículas como fuerzas gravitatorias, empuje, colisión con paredes o entre partículas y la que más suele destacar es la fuerza de arrastre (drag). El drag actúa en contra de la dirección de la velocidad de la partícula, junto a esto, el flujo continuo que también está en movimiento, provoca que la fuerza de arrastre actue en dirección $U - U_p$

En mecánica de fluidos, la fuerza de arrastre suele estimarse multiplicando la densidad del fluido (ρ_f), por el coeficiente de arrastre del cuerpo afectado (C_D), por el área afectada del cuerpo y por la velocidad del flujo al cuadrado. Pero teniendo en cuenta que la partícula se encuentra en movimiento la fórmula de fuerza de arrastre se reescribe como:

$$F_D = \frac{1}{2} \rho_f C_D |U - U_p| (U - U_p) A_p \quad (2.20)$$

Por otra parte, para estimar el coeficiente de arrastre de la partícula, en muchos estudios y experimentos se considera que todas las partículas tienen forma esférica para poder aplicar los modelos de coeficiente de arrastre disponibles para esta figura geométrica. La ecuación 2.21 muestra el modelo de coeficiente de arrastre para esferas disponible en OpenFOAM, que también fue aplicada por A. Lopez en las referencias [1] y [22].

$$C_D = \begin{cases} \frac{24}{Re_p}, & \text{si } Re_p < 1 \\ \frac{24}{Re_p} (1 + 0,15 Re_p^{0,687}), & \text{si } 1 \leq Re_p \leq 1000 \\ 0,44, & \text{si } Re_p > 1000 \end{cases} \quad (2.21)$$

3 | Método computacional

3.1. Introducción a OpenFOAM

Se explica en la referencia [42] que OpenFOAM (Open Field Operation and Manipulation), es un software de simulación numérica enfocado principalmente a la dinámica de fluidos computacional (CFD), basados en técnicas de volumen finito. es capaz de resolver tanto flujos sencillos como complejos, incluyendo flujos turbulentos, fenómenos de transferencia de calor, reacciones químicas e incluso problemas de otros campos como electromagnetismo.

Este software es de tipo open-source, es decir, su código fuente tiene libre acceso y puede ser modificado, utilizando lenguaje C++. En otras palabras, OpenFOAM se forma de una librería de aplicaciones que pueden ser solvers y utilidades. Los primeros resuelven un problema físico según las ecuaciones correspondientes, mientras que las utilidades se ocupan entre otras cosas para manipular datos, preparar el solver, analizar resultados. Entonces, es posible resolver tanto flujos sencillos como complejos, incluyendo flujos turbulentos, fenómenos de transferencia de calor, reacciones químicas e incluso problemas de otros campos como electromagnetismo o incluso economía.

Ventajas:

- Menor costo económico en comparación a otros software con licencias costosas
- Flexibilidad y capacidad de ser usado en paralelo a otros programas
- Permite entender mejor el proceso de cálculo y hacer modificaciones

Desventajas:

- Requiere mayor conocimiento de programación (en especial lenguaje C++ si se quiere modificar el código fuente)
- Mayor tiempo de aprendizaje
- Hay poca documentación y esta se va modificando frecuentemente por actualizaciones

3.1.1. Estructura de una simulación (case)

Un problema ingenieril desarrollado en este software, se compone de un conjunto de carpetas en donde se especifican solvers, utilidades, librerías o diccionarios entre otras cosas correspondientes a lo que necesita el usuario. Este conjunto de carpetas que conforman la simulación computacional se le conoce como *case*, en donde se destacan las 3 carpetas principales que son las que se encuentran prácticamente en cualquier simulación de OpenFOAM [7] y estas son:

3.1.1.1. Directorio 0

Esta carpeta corresponde a los directorios de tiempo. Específicamente, posee el nombre 0, ya que es el tiempo en donde comienza a correr cualquier simulación y en esta encuentran definidas las condiciones de borde e iniciales necesarias para la simulación (como velocidad, presión y/o temperatura) lo que se verá con mayor detalle en el Capítulo 4. También a medida que se lleva a cabo una simulación se van creando nuevas carpetas de directorios de tiempo donde se contienen las propiedades calculadas para el tiempo correspondiente. Esto depende del tiempo total que se busca simular y el paso de tiempo (δt) que se determine. Por ejemplo, un paso de tiempo (o time-step en inglés) de 0.1 segundos para 1 segundo de simulación, se crearán diez carpetas aparte de la carpeta inicial 0, cuyos nombres serán la sumatoria de los time-steps que hayan pasado (0.1, 0.2, ... 1).

3.1.1.2. Directorio Constant

En esta carpeta se ubican principalmente los archivos relacionados con las propiedades del fluido como por ejemplo el diccionario `transportProperties` (donde se especifican propiedades tales como densidad o viscosidad del fluido). Por otro lado se pueden especificar fuerzas externas, como las fuerzas gravitatorias es decir la magnitud que se considera y cual dirección se encuentran sus efectos respecto al dominio computacional. Para especificar las dimensiones, en las plantillas de texto de los diccionarios (donde es necesario especificar) se encuentra la palabra clave `dimensions` y luego entre dos corchetes hay 7 espacios [1 2 3 4 5 6 7] que representan masa [kg], longitud [m], tiempo [s], temperatura [K], cantidad [mol], corriente [A] y luminosidad [cd] respectivamente. Entonces, dando por ejemplo para la aceleración de gravedad se tendría:

```
dimensions      [0 1 -2 0 0 0 0];
value           (-9.81 0 0);
```

Donde se indica que la dimensión es en m/s^2 y que el valor de aceleración es 9,81 en dirección negativa del eje X.

Otra carpeta importante que contiene este directorio se denomina `polymesh`, la cual contiene información acerca de la malla. Esta se genera automáticamente luego de generar el mallado corriendo el comando `blockMesh`

3.1.1.3. Directorio System

Se encuentran los archivos que especifican condiciones y configuraciones relacionadas con el cálculo de la solución del problema, donde se destacan:

fvSchemes: Donde se escogen los métodos de interpolación y las opciones de discretización en general.

fvSolution: Donde se establecen parámetros del solucionador (solver) relacionado a su algoritmo, como tolerancia, método de cálculo de residuos, entre otras cosas.

controlDict: Se especifica todo el mallado temporal, número de time-steps, y cuales pasos de tiempo aparecerán en los directorios de tiempo, es decir pueden aparecer todos, los tres últimos, o solo el resultado final. También es posible incorporar funciones de post-procesamiento pero no son obligatorias.

blockMeshDict: Es donde se definen los parámetros para la geometría y mallado que generará la utilidad `blockMesh` (lo que se puede ver en mayor detalle en las secciones 3.2.1, 4.1 y en Anexo).

3.2. Pre-Proceso

3.2.1. Geometría y mallado

La geometría utilizada en primera instancia es la misma usada por A. Lopez [1]. Al no haber información específica de que se usó para hacer el mallado de esta, entonces se utilizó el software BlockMesh de OpenFoam el cual es el generador de mallado más básico que se tiene, utilizado para formación de bloques y figuras relativamente simples. En este generador también está la opción de generar figuras cilíndricas y semi-cilíndricas, lo cual conlleva a una mayor sencillez de mallado, no obstante, al mismo tiempo se vuelve más complicado el refinamiento de esta, debido a se producen errores cuando el mallado no es uniforme.

El dominio que se busca replicar, se tiene claro que el diámetro interno de la tobera es de 7 [mm] mientras que el diámetro del target es de 250 [mm]. Por otro lado, la altura del dominio está en función del eje X, por lo que el inlet se ubica en $X = 0$, mientras que el target en $X = 0, 105$ [m] y el largo de la tobera es de 80 ó 100 [mm] dependiendo de si el dominio tiene el target a distancia de 5 ó 25 [mm] desde la salida de la tobera respectivamente. En cuanto el resto de las medidas dominio, al no haber mayor información en el material escrito, se asume arbitrariamente sus valores.

Para realizar un mallado en BlockMesh, lo primero es definir una cierta cantidad de puntos por medio de coordenadas cartesianas, con los cuales se definen los límites del dominio y de los bloques, con respecto a la ubicación y la distancia entre estos. Cada bloque se compone de 8 puntos por lo que mientras mas compleja sea la geometría mayor cantidad de bloques van a ser necesarios y la mayoría de las veces, mayor refinamiento.

Entonces, para iniciar el proceso de mallado, se abre el la planilla de texto BlockMeshDict en la carpeta system del directorio, donde primero, se da la opción de especificar convertToMeter, que básicamente permite definir una escala. ya que la unidad de medida es en metro, al definir 0,001 por ejemplo, entonces todos los números que se utilicen posteriormente estarán en milímetro.

Luego, se encuentra la sección vertices donde se definen los puntos a utilizar, escribiendo su ubicación entre paréntesis, aplicando coordenadas cartesianas en la escala que fue definida previamente. A cada punto el software le asigna un número entero desde cero en adelante dependiendo del orden que se escriben y que posteriormente se utiliza para denominar los vértices en las secciones posteriores, por lo que es recomendable escribir la enumeración al costado usando doble barra diagonal (las barras diagonales evitan que un carácter influya en el código). Además, también se recomienda tener bien definidos la cantidad de puntos necesarios y su orden, ya al que solo agregar un vértice extra cambia la enumeración de todos los vértices que estén debajo de este y por consiguiente hay que cambiar manualmente los números que ya se hayan podido escribir en las secciones posteriores.

En la siguiente sección blocks, es donde se definen los bloques, usando la enumeración de los vértices para indicar la cara superior y la cara inferior. Se recomienda que la formación de bloques siga siempre la misma dirección, ya que puede volverse confuso para el usuario posteriormente o incluso producir errores al momento de correr el programa de mallado. Por medio de la función hex entre paréntesis escriben los 4 puntos que forman la cara superior del bloque siguiendo un orden en base a la regla de la mano derecha, y luego se escriben los cuatro puntos que formen la cara inferior, los vértices deben seguir el mismo orden que la cara superior o el programa arrojará error. Junto a esto, en otro paréntesis se escriben 3 números que deben ser enteros, los cuales definen la cantidad de divisiones que tendrá el bloque en dirección (X,Y,Z) de coordenadas cartesianas. A continuación, viene la definición de grading, donde lo más usual es la función simpleGrading, con la que se define si las divisiones definidas en su respectiva dirección previamente, deben ser iguales (si se escribe 1) o si en caso de escribir por ejemplo 0,5, esto significa que la última celda medirá la mitad de lo que mide la primera celda en esa dirección, mientras que las celdas que se encuentren entremedio, van reduciendo esta medida de manera uniforme.

A medida que se forman distintos bloques, si estos comparten una misma cara, el software automáticamente las une y considera como un mallado interno. Pero hay que tener en consideración que la cantidad

de divisiones (o cantidad de celdas) de las caras que una cada bloque deben ser consistentes o de lo contrario el software arrojará error al intentar hacer el mallado.

Definidos los bloques, es posible en la sección `edges` definir la formación de curvas en las orillas los bloques que lo requieran, por medio de arcos o splines. Usualmente no se ve la utilización de esta sección en tutoriales, ya que la mayoría de los dominios que se ven en estos no tienen curvas o se realizan sus mallados en otros softwares debido a la mayor complejidad de estos. Entonces, para casos como el presente, se define por medio `arc` que consiste en tomar dos puntos que estén ubicados en la orilla del bloque (nuevamente se recalca la importancia de tener bien definida la enumeración de los vértices) y entre paréntesis se escriben las coordenadas de un punto que se encuentre entre estos y que tenga el mismo radio de la curva que se busca definir, de esta forma el software al realizar el mallado hace una interpolación de este punto con los otros dos vértices creando celdas con orillas curvadas.

Finalmente viene la sección `boundary` en donde se definen los bordes o límites del dominio con un título escogido por el usuario, tipo de boundary (`patch`, `wall`, `wedge`, etc) y se especifican las caras pertenecientes a los bloques, por medio de escribir entre paréntesis 4 vértices (siguiendo regla de la mano derecha). Si hay más de una cara se escribe debajo de manera similar y el software las une en un solo boundary cuando se corre el software. Adicionalmente hay una sección llamada `mergePatchPairs` donde es posible unir caras internas que obstaculizan la simulación que se busca hacer.

3.3. Modelos numéricos

3.3.1. Solver

Debido a que se utiliza una versión más moderna del software utilizado en el estudio base, para realizar las simulaciones se elige el solver `DPMFoam` el cual es para flujos transientes incompresibles y su algoritmo de manera similar a `pimpleFoam` utilizado por A. Lopez, es una combinación de los algoritmos PISO y SIMPLE. Las siglas DPM en el nombre del solver indican que es específico para Discrete Particle Model, su principal ventaja que trae es que no necesita la modificación al código con lenguaje C++ que hizo A. Lopez en su trabajo para poder acoplar las fases continua y discreta de la simulación.

3.3.2. Modelo de turbulencia

En el directorio `Constant` en el diccionario `turbulenceProperties` se puede habilitar el modelo de turbulencia mientras que en `momentumTransport` se implementa el modelo de turbulencia RAS $\kappa - \epsilon$ (explicado en 2.3) junto con los coeficientes mostrados en la tabla 2.1.

3.3.3. Discretización temporal

En OpenFOAM para mantener la estabilidad de los modelos numéricos, se calcula para cada paso de tiempo el número de Courant C_o para cada una de las celdas de la malla numérica.

$$C_o = 0,5 \cdot \left(\frac{\phi}{V_{celda}} \right) \cdot \Delta t \quad (3.1)$$

Donde ϕ es el flujo que pasa por las celdas, V_{celda} es el volumen de la celda y Δt es el paso temporal. A diferencia de lo que se hizo en el caso de validación, en esta tesis se optó por el habilitar el comando `adjustTimeStep` en el diccionario `controlDict`, ya que de esta forma el software ajusta el paso de tiempo de acuerdo al valor máximo del número de Courant, el cual se especifica en el mismo diccionario, en `maxCo` como 1, ya que para un modelo de esquema explícito C_o no debe superar este valor para mantener la estabilidad, en caso de que lo sobrepase la simulación se detendrá automáticamente.

3.3.4. Modelo de erosión

El único modelo de erosión disponible en OpenFOAM (por ahora) es el de I.Finnie, y esta implementado en el archivo `ParticleErosion.C`. Como se explica en la referencia [43], en el diccionario `kinematicCloudProperties` se habilita este modelo (como se puede ver en 4.4.6), si las partículas golpean el borde especificado por el usuario, el software calcula la magnitud de velocidad de impacto de la partícula y la almacena como $mag(U)$. Seguido de esto se determina y almacena la dirección de la velocidad en el vector $Udir$, para que con la componente normal de la superficie del patch, se obtenga el ángulo de impacto y es almacenado en $alpha$. Entonces, junto al resto de los coeficientes especificados por el usuario, se calcula el volumen de pérdida de material Q por medio de las ecuaciones 1.2 y 1.3 en la ubicación de la malla, donde impacto la partícula, los resultados se escriben como un campo escalar que se puede visualizar posteriormente.

3.4. Post-procesamiento

Terminada la simulación, todo lo relacionado a extraer, ordenar, visualizar resultados, realizar cálculos extras, entre otras cosas, se conoce como post-procesamiento, siendo posible aplicar las herramientas que el mismo software otorga como también herramientas externas que permitan realizar el proceso al usuario.

3.4.1. ParaView

El software principal para el post-procesamiento que hay para OpenFOAM, viene incluido con la instalación de este mismo y se conoce como *ParaView*, sirviendo de visualizador y analizador de resultados.

Lo primero que se hace es visualizar los perfiles de campos de velocidad que se forman por el flujo como se puede apreciar en las Figuras 5.1 y 5.2.

Es posible visualizar sub-dominios específicos y campos escalares y/o lagrangiano marcando y desmarcando casillas en el costado derecho del visualizador. Entonces, dejando marcada solo la casilla de *kinematicParticleTrack* se visualiza el rastreo de partículas sólidas desde el *inlet* hasta la superficie del *target*. En la barra superior se busca y selecciona el comando **Clip** la cual puede extraer las partículas específicamente ubicadas justo sobre la superficie del *target*. Luego es posible abrir una ventana al costado derecho que muestra una tabla con los datos de la ubicación (en coordenadas cartesianas) y del vector de velocidad de las partículas extraídas. Esta información se puede exportar en un archivo .CVS.

De manera similar, si se deja marcada las casillas del sub-dominio del *target*, la casilla de *kinematic:Q* y utilizar el comando **Plot over line**, especificando que se grafique el eje **Y** estas se puede extraer también en formato .CVS el perfil de erosión a lo largo del diámetro de la superficie de la muestra. Estos archivos exportados se pueden visualizar luego en otro software.

3.4.2. Python

Por medio del software *Jupyter Notebook*, se extrae la información de los archivos .CVS obtenidos previamente, con lo que se puede hacer post-procesamiento adicional de los resultados. Comenzando por separar las partículas que estén lo más cercanas posible a la superficie del *target* (un margen de $2 \cdot 10^{-4}$ [m]). De esta forma, se extrae de las partículas que cumplen con esa condición, sus otras dos coordenadas cartesianas (en eje **Z** e **Y**) con las cuales se crea un gráfico de dispersión (como se ve en la sección 5.1.2 de los impactos sobre la superficie).

Luego, con los datos del vector de velocidad de cada partícula extraída, los cuales vienen escritos con sus respectivas componentes $\mathbf{X}, \mathbf{Y}$ y $\mathbf{Z}$, es posible obtener el ángulo de impacto de la partícula. Considerando que para obtener el seno del ángulo entre un vector y un plano, se utiliza la fórmula.

$$\sin \alpha = \frac{|\vec{V} \cdot \vec{n}|}{|\vec{V}| \cdot |\vec{n}|} \quad (3.2)$$

Donde $\vec{n}$ es la componente normal al plano, que en el caso de la superficie de impacto vendría siendo $(-1, 0, 0)$. Entonces el ángulo:

$$\alpha = \arcsin \frac{V_x \cdot (-1) + V_y \cdot (0) + V_z \cdot (0)}{\sqrt{V_x^2 + V_y^2 + V_z^2}} \cdot \sqrt{(-1)^2}$$

$$\Rightarrow \alpha = \arcsin \frac{V_x}{\sqrt{V_x^2 + V_y^2 + V_z^2}}$$

Obtenidos los valores de velocidad y ángulo de impacto. De forma similar a como se explica en el trabajo de A.Lopez, se agrupan en tramos de 1 [mm] de distancia desde el centro del target, se calculan los promedios de cada tramo y finalmente se crea un gráfico lineal respectivamente de la velocidad y del ángulo respecto a la distancia del centro del target.

3.4.3. G3Data Graphs

Por medio del software *G3Data Graph* disponible en Ubuntu, se pueden extraer los datos de velocidad, ángulo de impacto y de ratio de erosión de los papers de A. López, o de otros documentos con los que se quiera comparar los datos obtenidos en las simulaciones. Cabe destacar que estos datos son estimaciones de lo entregado en los gráficos publicados en los papers y otros documentos, ya que no se tienen los datos numéricos exactos que obtuvo el autor de sus respectivas simulaciones. No obstante, por medio de estos datos, se pueden crear en *Jupyter Notebook* gráficos idénticos a los expuestos en las publicaciones mencionadas, de manera que en la sección **Verificación y validación** es posible apreciar la comparación de los resultados esperados con los obtenidos junto con estimar los errores porcentuales.

4 | Parámetros computacionales

Las simulaciones fueron realizadas en la versión 8 del software OpenFoam, por lo que se debe tener en cuenta que pueden haber diferencias en la estructura de algunos códigos y algoritmos, lo cual podría influir en los resultados que se busca replicar.

En el presente capítulo se busca también especificar distintos aspectos, parámetros, valores, entre otras cosas que hayan tenido que ser definidas de forma arbitraria, debido a la ausencia de especificación e información en la referencia [1].

4.1. Geometría dominio computacional

Reiterando lo planteado en el capítulo anterior, al faltar información de algunas dimensiones, se determinaron de forma arbitraria al momento de hacer el mallado, tal como se puede apreciar en las figuras:

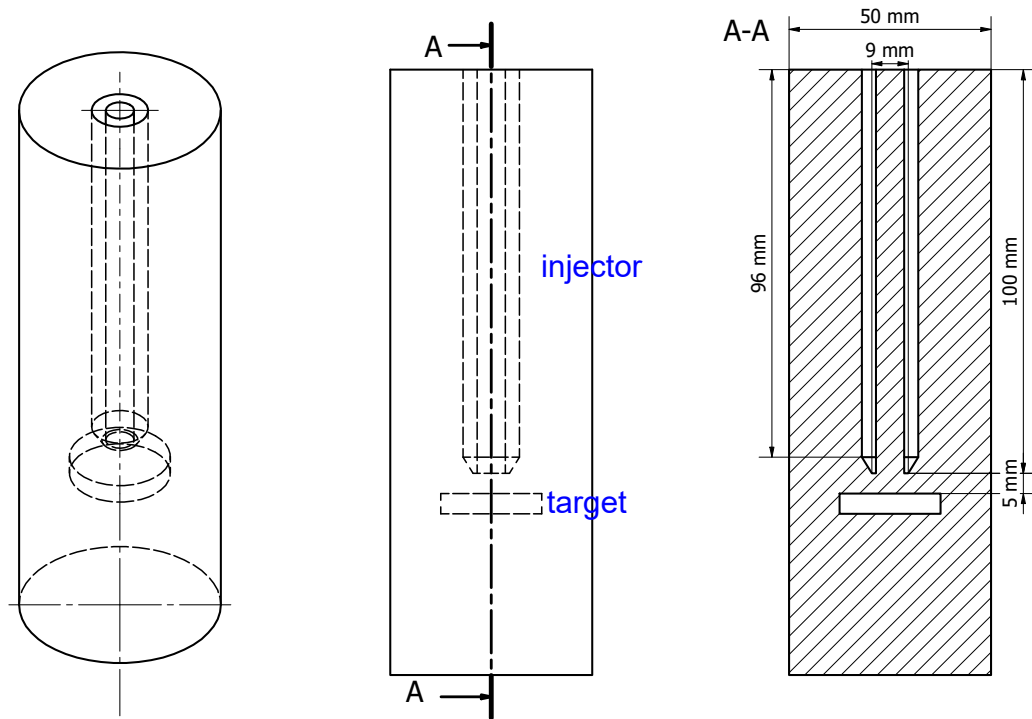


Figura 4.1: Geometría para distancia 5 mm.

(Fuente: Elaboración propia en Autodesk Inventor)

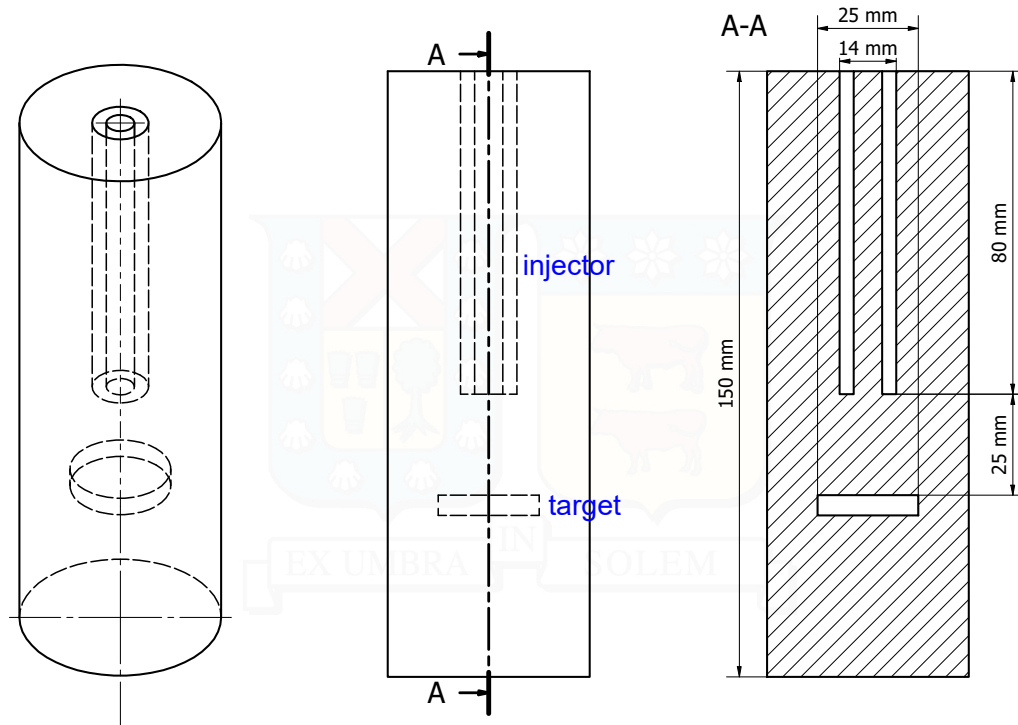


Figura 4.2: Geometría para distancia 25 mmn.// (Fuente: Elaboración propia en Autodesk Inventor)

4.2. Condiciones de borde/iniciales

En el directorio 0, se encuentran las plantillas de texto que configuran las condiciones borde/iniciales de las variables de velocidad U , presión estática p , y de turbulencia k .water, ϵ .water y ν .water. Para cada variable se definen las condiciones iniciales respecto a los bordes o límites definidos previamente en el mallado.

fixedValue: Impone una condición de Dirichlet, en otras palabras, un valor fijo para todas las celdas del patch. Para inlet es $10 [m/s]$, mientras que para target fue cero.

noSlip: Impone condición de no deslizamiento, ideal para paredes sólidas o algún patch que asimile velocidad cero.

zeroGradient: Impone una condición de Neumann, es decir, el gradiente normal a la variable es igual a cero.

inletOutlet: Intercambia entre las condiciones fixedValue y zeroGradient dependiendo de la dirección de U en patch. Entrega mayor estabilidad en el topOutlet que solo zeroGradient, ya que permite que flujo salga del dominio pero se puede especificar que no entre flujo debido a gradientes de presión.

fixedFluxPressure: Ajusta el gradiente de presión, de forma de que el flujo en el patch sea el impuesto por la condición de velocidad.

calculated: La variable se deriva del resultado de otros campos.

kqRWallFunction: Establece una condición inicial para la energía cinética turbulenta.

epsilonWallFunction: Establece una condición inicial para la disipación de energía cinética turbulenta.

nutWallFunction: Establece una condición inicial para la viscosidad turbulenta basada en la energía cinética turbulenta.

cyclic: Utilizada principalmente condiciones de borde periódicas (por ejemplo dominios que tengan rotación) y para tratar 2 regiones como si estuvieran físicamente conectadas.

Tabla 4.1: Condiciones de borde para cada patch/límite definido.

Patch	U	p	k.water	epsilon.water	nut.water
inlet	fixedValue	fixedFluxPressure	fixedValue	fixedValue	calculated
internalWalls	noSlip	zeroGradient	kqRWallFunc	epsilonWallFunc	nutWallFunc
externalWalls	noSlip	zeroGradient	kqRWallFunc	epsilonWallFunc	nutWallFunc
targer	fixedValue	fixedFluxPressure	kqRWallFunc	epsilonWallFunc	nutWallFunc
center1	cyclic	cyclic	cyclic	cyclic	cyclic
center2	cyclic	cyclic	cyclic	cyclic	cyclic
topOutlet	inletOutlet	fixedValue	zeroGradient	zeroGradient	calculated
bottomOutlet	noslip	fixedFluxPressure	zeroGradient	zeroGradient	calculated

Adicionalmente, cabe recordar lo que se explicó en la sección 3.1.1.2, donde se especifica la dirección y magnitud de la aceleración de gravedad en el diccionario **g** disponible en el directorio **constant**. Además, en este directorio, se especifica la densidad y viscosidad cinemática del flujo continuo como $998 [kg/m^3]$ y $10^{-6} [m^2/s]$ respectivamente en el diccionario **transportProperties**. mientras que en **momentumTransport** se implementa el modelo de turbulencia RAS $\kappa - \epsilon$ junto con los coeficientes mostrados en la tabla 2.1.

4.3. Configuración simulación

En el directorio **System**, de dejo la configuración por default de los diccionarios **fvSchemes** y **fvSolution**.

En el diccionario **controlDict**, en la primera línea donde dice **application**, se especifica que se use el solver **DPMFoam**.

```

application      DPMFoam;

startFrom        latestTime;

startTime        0;

stopAt           endTime;

endTime          1; // 5

deltaT           1; // 1.8 e-5

```

Luego se configura para que el inicio sea **startTime**, que significa que iniciara del tiempo indicado en la siguiente línea, en este caso 0 (el tiempo esta en segundos). De manera similar se configura para **endTime**, en otras palabras, que se la simulación se detenga en el tiempo indicado en la línea siguiente, en este caso, después de un segundo. En **deltaT** se especifica el paso de tiempo de la simulación, la cual en el artículo de A. Lopez [1] es de $1,875 \cdot 10^{-5} [s]$, pero para este trabajo se deja escrito valor 1, lo cual se justifica al final de esta sección

```

writeControl      runTime;

writeInterval     1;

```

```

purgeWrite      0;

writeFormat     binary;

writePrecision  6;

writeCompression off;

timeFormat      general;

timePrecision   6;

```

Prosiguiendo con la configuración, se especifica en `writeControl` la opción `runTime` que implica que el software escribirá resultados en el tiempo indicado en `writeInterval`, en este caso escribirá los resultados obtenidos en el segundo 1, (osea, terminando la simulación), entonces el software crea un directorio con el nombre del time-step, en formato binario (especificado en `writeFormat`) de manera que se visualizan en el post-procesamiento. Se encuentra la posibilidad de, por ejemplo de configurar para cada 0,2 segundos, que en otras palabras, el software creará cinco carpetas en el directorio que contiene los datos de velocidad, presión, erosión, dependiendo de lo que se especifica el usuario, no obstante esto implica mayor uso de memoria del PC en el que se esta simulando, por lo que se recomienda buscar que se registren dentro de lo posible, solo los resultados necesarios. En la línea `purgeWrite` esta la opción de escribir la cantidad de directorios que pueden quedar, por ejemplo, si se configura 2 para una simulación de 5 [s], con `writeInterval` igual a 1, entonces se crearán los dos primeros directorios, para cuando se forme el tercero, se borrará el primero dejando siempre 2 directorios, hasta que termine la simulación.

Por otro lado, `timeFormat` se elige el formato en que se nombran los directorios, dejando `general` por default, mientras `writePrecision` y `timePrecision` se especifica que hayan máximo de 6 decimales en los resultados y en los time-steps. Adicionalmente, `writeCompression` da la posibilidad de que los resultados se compriman en *gzip*, lo cual no fue usado en esta ocasión.

```

runTimeModifiable yes;

adjustTimeStep yes;

maxCo 1;

maxDeltaT 0.1;

```

La línea de `runTimeModifiable`, si se encuentra activado con `yes`, permite modificar algunas configuraciones como la duración o intervalo de registro de resultados, independiente de que si la simulación ya este corriendo.

Finalmente, `adjustTimeStep` permite que el Δt (mencionado previamente como igual a 1, se vaya modificando automáticamente a medida que avanza la simulación, en base al valor del número de Courant, como se explicó en la sección 3.3.3. Además, en `maxDeltaT` se configura para que time-step no sea mayor a 0,1 segundos (lo cual no debería ocurrir si el mallado esta bien hecho para la simulación).

4.4. Configuración de partículas

En el diccionario `kinematicCloudProperties` ubicado en el directorio `constant` se definen las propiedades relacionadas con las partículas. Las más importantes definidas para las simulaciones realizadas para este informe son:

4.4.1. Ajustes generales

```

active          true;
coupled         false;
transient       true;
cellValueSourceCorrection on;
maxCo           1;
calcFrequency   0.001;
maxTrackTime    100;

```

- Activar (true) o desactivar (false) la nube de partículas (fase dispersa) en la simulación quedando la fase continua.
- Habilitar (true) o deshabilitar (false) acoplamiento bidireccional, es decir la influencia de la fase discreta sobre la fase continua, lo cual en este caso no es utilizado.
- Especificar si la simulación del flujo es transitorio (true) o estacionario (false).
- Opción de habilitar corrección de momento transferido a la fase Euleriana.
- Número de Courant máximo estimado por el usuario.
- Frecuencia de calculo y tiempo máximo de rastreo en segundos.

Elegir esquemas de interpolación/integración para el rastreo de partículas y tratamiento de términos fuentes (source terms). Se dejaron definidos como en la plantilla del tutorial de DPMFoam.

```

interpolationSchemes
{
    rho.water          cell;
    U.water            cellPoint;
    mu.water           cell;
    alpha.water         cell;
}
averagingMethod basic;
integrationSchemes
{
    U                  Euler;
}
sourceTerms //
{
    schemes
    {
        U semiImplicit 1;
    }
}

```

Por otro, se puede especificar un modelo de dispersión de partículas que básicamente le agregan aleatoriedad a la forma que se distribuye el trayecto de las partículas y al lugar de impacto en la pared que lo hace ser más parecido a lo que ocurre en la realidad. El modelo elegido fue **stochasticDispersionRAS** que le agrega dispersión a las partículas en base a los campos calculados de energía cinética κ y de disipación ϵ de energía turbulenta.

4.4.2. Propiedades físicas

Definir propiedades físicas de la partícula como densidad que es $2206 [kg/m^3]$ para la simulación con 5 mm de distancia tobera-superficie, y $2300 [kg/m^3]$ para 25 mm de distancia. Se define también el

modulo de Young [N/m^2], el coeficiente de Poisson [–] y la máxima fracción volumétrica (**alphaMax**) [–].

```
constantProperties
{
    rho0          2206; //2300;
    alphaMax      0.9;
    poissonsRatio 0.35;
    youngsModulus 1e8;
}
```

4.4.3. Fuerzas actuando sobre las partículas

Se definen las fuerzas que actúan sobre las partículas como fuerza de arrastre (drag force), gravedad, fuerzas de presión, flotabilidad entre otras.

```
subModels
{
    particleForces
    {
        sphereDrag;
    }
}
```

Para todas las simulaciones hechas, solo se considera la fuerza de arrastre y se especifica la implementación del modelo de coeficiente de arrastre C_D para una esfera, (ver ecuación 2.21).

4.4.4. Inyección de partículas

Escoger un modelo de inyección de partículas disponibles en el software. Se encuentra la opción de colocar un nombre elegido por el usuario. El tipo de modelo más utilizado y que se aplica aquí se denomina **patchInjection** y consiste en seleccionar uno o varios de los bordes del dominio computacional, que fueron definidos durante el mallado, en este caso fue **inlet**.

En **parcelBasisType** se puede elegir como enumerar/representar las partículas ya sea por número o masa, en el caso de **fixed**, implica que hay un número fijo de partícula en cada parcela que se especifica en la línea **nParticle**. Por otro lado, en **massTotal** se especifica la masa total de partículas que se inyectarán, pero en este caso se puede dejar como 0 y no influirá en la simulación.

```
model1
{
    type          patchInjection;
    parcelBasisType fixed;
    patchName      inlet;
    U0             (10 0 0);
    parcelsPerSecond 53308;
    meanParticleDiameter 0.00025;
    nParticle       1;
    sizeDistribution
    {
        type          fixedValue;
        fixedValueDistribution
        {
            value      0.00025;
        }
    }
}
```

```

    }
    flowRateProfile constant 1;
    massTotal          0;
    SOI 0;
    duration 2000;

```

- **U0**: es la velocidad inicial de las partículas cuando ingresan al dominio.
- **parcelsPerSecond**: Cantidad de parcelas que se simulan por segundo, por lo que al ser **nParticle** igual a 1, significa que en el target deberían chocar alrededor 53308 partículas en la simulación de 1 segundo.
- **meanParticleDiameter**: Diámetro de la partícula en metros.
- **sizeDistribution**: Se especifica el tipo de distribución de tamaño de partículas en base a estadística como por ejemplo distribución tipo **normal** respecto a la campana de Gauss. Por otro lado, esta la opción de que todas partículas tengan el mismo tamaño con **fixedValue** como fue en el presente caso.
- **flowRateProfile**: Distribución de partículas inyectadas con respecto a la duración de inyección.
- **SOI**: Start-Of-Injection, segundo de la simulación en el que inicia la inyección de partículas. En este caso se especifica que empiece apenas inicie la simulación. Cabe mencionar que como se explicó en el algoritmo de DPMFoam, primero se debe empezar a simular el flujo de la fase continua antes de iniciar el rastreo de la fase dispersa por lo que puede haber una pequeña diferencia de tiempo de iniciar la inyección.
- **duration** Duración de la inyección de partículas. No influye que sea mayor al tiempo total que se está simulando.

4.4.5. Modelos y coeficientes de interacción de partículas

Se define modelo de interacción partículas-paredes, en este caso se utiliza **localInteraction** donde se especifica la interacción de las partículas todos los patches disponibles pudiendo elegir entre rebote (rebound), pegado (stick), escape (salida de las partículas del dominio) o ninguno (este no se recomienda porque puede provocar errores, se recomienda usarse solo si hay plena seguridad de que no hay interacción con las partículas).

```

patchInteractionModel localInteraction;
localInteractionCoeffs
{
    patches
    (
        internalWalls
        {
            type escape;
        }
        "center1 | center2 "
        {
            type none;
        }
        externalWalls
        {
            type escape;
        }
        target
        {

```

```

        type escape ; // rebound
        // e    0;
        // mu   0;
    }
    "inlet | bottomOutlet | topOutlet "
    {
        type escape ;
    }
    );
}

```

Se elige este modelo porque da la posibilidad de que no todas las paredes tengan la misma condición. Por lo que en caso de querer simular que las partículas reboten específicamente en el *target*, no afecta los demás patches. El término *e* representa el coeficiente de restitución para la componente vertical de la velocidad incidente, mientras que *mu* representa el coeficiente de atenuación para el componente tangencial de la velocidad incidente, ambos coeficientes se definen por el usuario en base a información de modelos de interacción partícula-pared que se pueden encontrar en la literatura y a las condiciones que se busca simular. Para este caso específico, A. López [1] definió condición de escape para todas las superficies en el primer caso de validación.

Por otro lado, se encuentra la opción de definir modelo y coeficientes relacionados a la interacción partícula-partícula pero como no se utiliza ese tipo de acoplamiento, no influyen en la simulación los valores puestos por descarte en la plantilla, por lo que se pueden ignorar simplemente. Además se deja especificado como *none* a la modelación de transferencia de calor, capa superficial, radiación, entre otras, ya que no se consideran influyentes para estas simulaciones.

4.4.6. Funciones de nube (CloudFunctions)

Se conocen como *cloudFunctions* las utilidades que permiten calcular y/o registrar variables relacionadas a la nube de partículas inyectadas al dominio. Las utilizadas son:

particleTracks : Guarda todas las variables de la partícula (velocidad, presión, etc). En postprocesamiento en ParaView es posible exportar a formato *CVS* las variables que el usuario necesita.

particleErosion : Esta función crea un campo de erosión producido por las partículas en los patches especificados por el usuario. Los coeficientes *psi* y *K* vienen definidos por defecto, mientras que *p* es el yield strength (límite elástico) del material de la muestra medida en Pascales.

```

cloudFunctions
{
    particleTracks1
    {
        type            particleTracks ;
        trackInterval    1;
        maxSamples       100000000;
        resetOnWrite     yes ;
        log              yes ;
    }
    particleErosion1
    {
        type particleErosion ;
        patches
        (
            target
        );
    }
}

```

```

        p 11000000;
        psi 2;
        K 2;
    }

```

4.5. Simulación paralela o Paralelización

Uno de los métodos más comunes para disminuir el tiempo de cálculo y de simulación es la computación paralela de datos, en otras palabras, la subdivisión del dominio en un número específico de partes, lo que se conoce como descomposición del dominio. A cada subdivisión se le asigna un procesador o núcleo del computador donde se realiza la simulación. Para implementar esto se utiliza el diccionario `decomposeParDict` (Ver Anexo A.1.6), ubicado en la carpeta `system`, donde se especifica el número de núcleos virtuales (en este caso 6 era el máximo del PC utilizado) y el modelo de paralelización a utilizar. Además, en las simulaciones se utiliza el modelo `scotch` que es el mas simple de usar ya que solo se especifica el número de núcleos en `numberOfSubdomains` y los demás parámetros son automáticos, por lo que se deja el resto de la plantilla tal cual fue obtenida del tutorial ya que no influye en la simulación.

Por otra parte, una vez que la simulación esta completada, en orden de visualizar los resultados pasar al post-procesamiento, es necesario reunir las subdivisiones por medio del comando `reconstructPar`. Mientras más time-steps se hallan escrito y guardado, mayor tiempo tomará la reconstrucción del dominio.

4.6. Problemas en el transcurso de las simulaciones

4.6.1. Problemas con las condiciones de borde

Las condiciones de borde expuestas en la Tabla 4.1 fueron las últimas en ser configuradas. La mayoría fueron puestas en la primera instancia en base a la información que había en el material base. En los patches con los que más hubo complicaciones fue con `topOutlet` y con `bottomOutlet`. Esto debido a que en primera instancia, se daba a entender que ambos eran salida de presión (pressure outlet) del dominio, sin embargo al momento de visualizar los resultados de la simulación el campo de velocidad tenía un perfil distinto al esperado. Se procedió a realizar otras simulaciones cambiando las condiciones de borde de ambos patches, de modo que a ensayo y error se llego al campo de velocidad que se esperaba dejando solo `topOutlet` como pressure-outlet con la condición `inletOutlet` mientras que `bottomOutlet` se deja con condición de no deslizamiento en velocidad y en la presión se dejó `fixedFluxValue` de modo que la presión tomará el valor necesario para mantener el flujo establecido.

Por otro lado, los dos patches `center1` y `center2`, previamente formaban un solo plano que al cual en primera instancia se asignó la condición de borde `symmetryPlane` (la cual revisando en literatura se encontró que esta pensado para dominios en 2D), así que se cambió la condición de `symmetry`, pero en ambos casos ocurrió un efecto en el campo de velocidad que se puede apreciar en la figura. Entonces, para solucionar esto se optó por dividir el plano de simetría en dos partes iguales y usar la condición de borde `cyclic` pero escrito de forma que de que no haya rotación y al ser el dominio axial-mente simétrico la condición de borde termina asimilando la de simetría y solucionando el problema.

4.6.2. Problemas con el mallado

El primer problema con el mallado fue la falta de información respecto a dominio y mallado con forma semi-cilíndrica, de modo que requirió practica, ensayo y error poder ir aprendiendo. Segundo fue la cantidad de bloques requeridos para formar la geometría del dominio computacional, muchas veces se lanzo

error debido a que el número y/o tamaño de las celdas no coincidían, o incluso ocasiones en que dos bloques no quedaban bien unidos formando un par de bordes (patches) internos que en teoría era donde debía pasar libremente el flujo simulado, por lo que asignarle una condición de borde, fuera cual sea, iba a cambiar la dinámica de toda la simulación, por lo que fue necesario rehacer varias veces el mallado, investigar en la literatura y encontrar soluciones alternas como por ejemplo la utilidad `mergePatchPairs` en el diccionario de `blockMeshDict` que permite unir dos patches internos para que no haya bordes interiores no deseados en el dominio.

Por otra parte, hubo un problema con el tipo de mallado, al ser la figura semi-cilíndrica, en primera instancia fue una malla polar tipo O, que funcionaba solo hasta cierto grado de refinamiento, sin embargo, era necesario el refinamiento de la malla en especial en la zona de tobera y en la pared del target. El problema se daba que en este tipo de malla las celdas cercanas al centro de simetría axial, siempre tendrán un volumen más reducido por lo que al aumentar el número de celdas, el tamaño de las que se encontraban en esa zona se reducía tanto que el número de Courant superaba el valor unitario, provocando inestabilidad, que el paso de tiempo se redujera a valores de 10^{-24} y finalmente el software arrojara error e interrumpiera la simulación. Para solucionar esto se rediseñó el mallado para que fuera *butterfly type* [44], tal como se puede apreciar en la figura 4.3. Debido a que era la mitad de un cilindro el dominio utilizado, se utilizó la mitad un trapecio en el centro (como se aprecia en la Figura A.1 en Anexo) en vez de la mitad de un cuadrado (lo que terminaría siendo un rectángulo o un triángulo en el centro). Gracias a esto se pudo refinar mejor el mallado, completar las simulaciones y obtener mejores resultados.

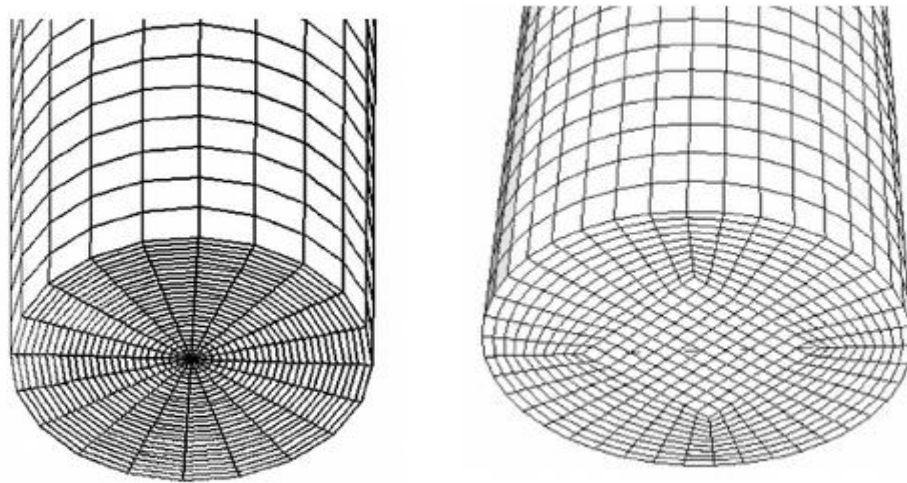


Figura 4.3: Malla tipo O a la derecha y tipo butterfly a la izquierda [44].

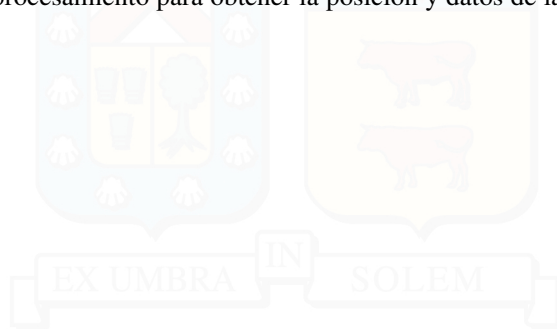
4.6.3. Problema con el post-procesamiento

En un principio se aplicó la función `patchPostProcessing`, la cual registra todas las partículas (junto con sus variables velocidad, ubicación, N° de celda, entre otras) que entren en contacto con el patch que especifique el usuario en el diccionario `kinematicCloudProperties` como se puede apreciar en el siguiente ejemplo:

```
patchPostProcessing1
{
    type                patchPostProcessing;
    maxStoredParcels    100000;
    patches              ( target );
}
```

Sin embargo, a partir de la versión del software OpenFOAM 5.0, se utilizan método baricéntrico de rastreo de partículas que hacen que los solucionadores sean más robustos pero cuando se obtiene el output de `patchPostProcessing` entrega la posición de las partículas en coordenadas baricéntricas, lo cual dificulta considerablemente el post-procesamiento de los resultados, en especial para graficar los puntos de impactos.

Debido a esto, se optó por utilizar la función `particleTracks`, que tiene la desventaja de que guarda el rastreo de la partícula por todo su recorrido por el dominio (no solo cuando llega al target) haciendo que los resultados de la simulación ocupen mayor cantidad de memoria para ser guardados y haciendo un poco mas tedioso el post-procesamiento para obtener la posición y datos de las partículas que entran en contacto con el target.



5 | Validación de modelo CFD

5.1. Primer caso de validación código:

Se simularon ambas geometrías expuestas en la referencia [1]. Tal como se explicó en los capítulos anteriores, muchos parámetros y ajustes tuvieron que ser asumidos de forma arbitraria buscando primero que el perfil de velocidad se pareciera lo más posible al mostrado en el artículo. Como se puede apreciar en la sección 5.1.1, los perfiles obtenidos de manera cualitativa se parecen lo suficiente para considerar que las simulaciones van por buen camino.

Tal como se explicó en la sección de post-procesamiento, se extraen los datos de los resultados de velocidad, se calculan sus respectivos ángulos, se agrupan y promedian cada 0,5 mm de distancia desde el centro y se muestran en los gráficos de las Figuras 5.5 y 5.6. Además, se muestra la representación gráfica de los impactos de las partículas sobre la superficie de la muestra (target).

5.1.1. Comparación de perfil de velocidades

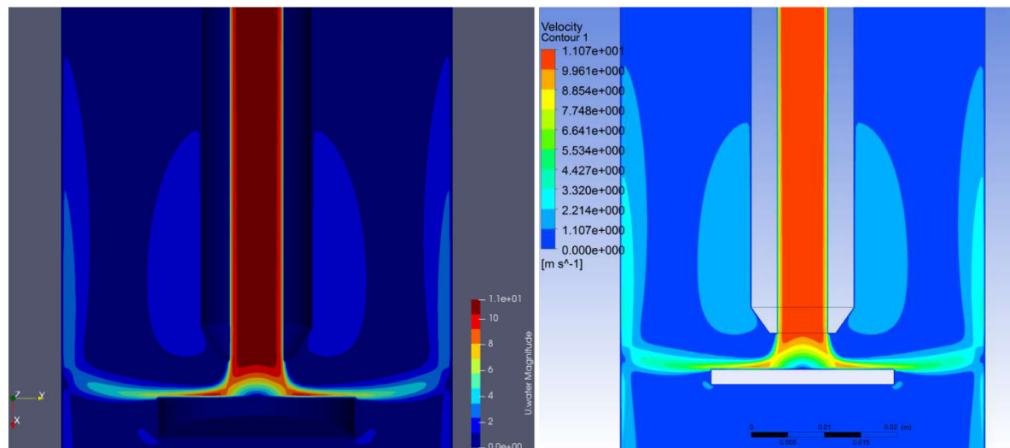


Figura 5.1: Perfil velocidad para 5 mm de distancia desde la salida del flujo de la tubería. En la derecha es el obtenido y el de la izquierda es el resultado obtenido por A. Lopez [1].

De acuerdo con las Figuras 5.1 y 5.2, donde se muestra la distribución de velocidades para el target a 5 y 25 mm de distancia respectivamente, se ve que la velocidad es alta al salir del inyector, pero su punto máximo es un poco antes de alcanzar el target, habiendo luego una reducción significativa al colidir con este. Al impactar el fluido y las partículas se dispersan hacia los costados, alcanzando alrededor de la mitad de la velocidad inicial.

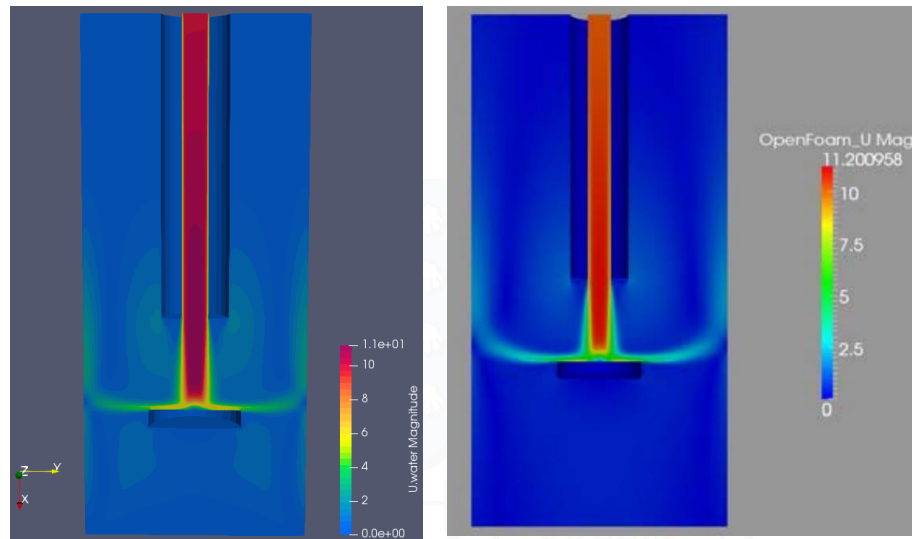


Figura 5.2: Perfil velocidad obtenido para 25 mm de distancia desde la salida del flujo de la tubería. En la derecha es el obtenido y el de la izquierda es el resultado obtenido por A. Lopez [1].

5.1.2. Visualización de impactos sobre la superficie de la muestra

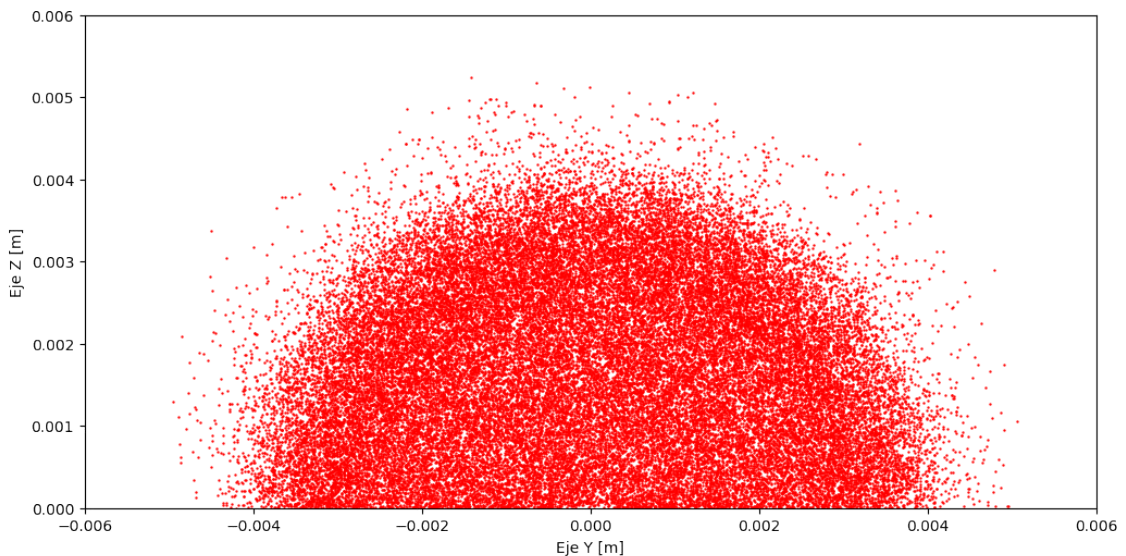


Figura 5.3: Graficación de concentración de impactos sobre el target cuando esta a 5mm de distancia.

Respecto a la Figura 5.3, se que la dispersión radial de las partículas se aproxima a 4 [mm] de forma similar a lo que se ve en Figura A.2a (Ver Anexo). Otro aspecto a recalcar es que la distribución de las partículas es notoriamente uniforme en la Figura 5.3 hasta que pasa los 4 [mm] de distancia desde el centro, mientras que en la Fig. en anexo hay mayor concentración en el centro de la superficie (alrededor de 1 [mm] de radio) y en el radio de 4 [mm].

En cuanto a la Figura 5.4, se pudo observar que hubo menor dispersión radial (aproximadamente hasta 5 [mm] de radio) en comparación a lo obtenido por A. López en Figura A.2b (Ver Anexo), que alcanzó

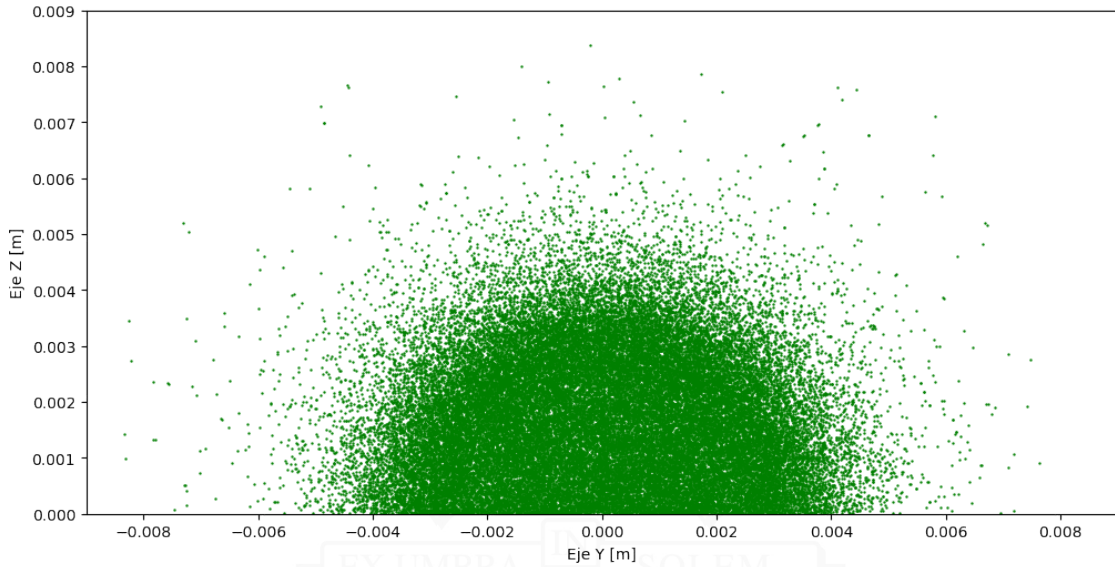


Figura 5.4: Graficación de concentración de impactos sobre el target cuando esta a 25mm de distancia.

alrededor de 12 [mm] de dispersión radial. No obstante, considerando que se utilizó una versión del software más actual, con un solver (DPMFoam) más especializado para este tipo de simulación, por lo que pueden haber diferencias sutiles en la codificación de los algoritmos que resuelven el flujo y el rastreo de partículas. Además, este software para el rastreo de partículas actualmente ocupa las coordenadas baricéntricas lo que en teoría da mayor robustez a la simulación y rastreo, según la página oficial de OpenFOAM [45]. Junto a esto, recordando que según lo visto en la literatura, de acuerdo al número de Stokes, al ser este mayor que 1 es normal que la mayoría de las partículas se concentren más en la zona cercana al diámetro de la boca de salida del chorro, mientras que una menor cantidad va a tender a seguir las líneas del flujo continuo

5.1.3. Velocidad promedio y ángulo de impacto vs distancia del centro

En la Figura 5.5 se muestran las velocidades y ángulos de impacto promedios en radios con intervalos de 0,5 milímetros. Se puede apreciar que la velocidad varia muy poco entre los 6-7 [m/s], teniendo su punto máximo en el radio 3 milímetros. El ángulo de impacto tuvo su punto máximo en el centro de la muestra y su valor fue disminuyendo casi linealmente.

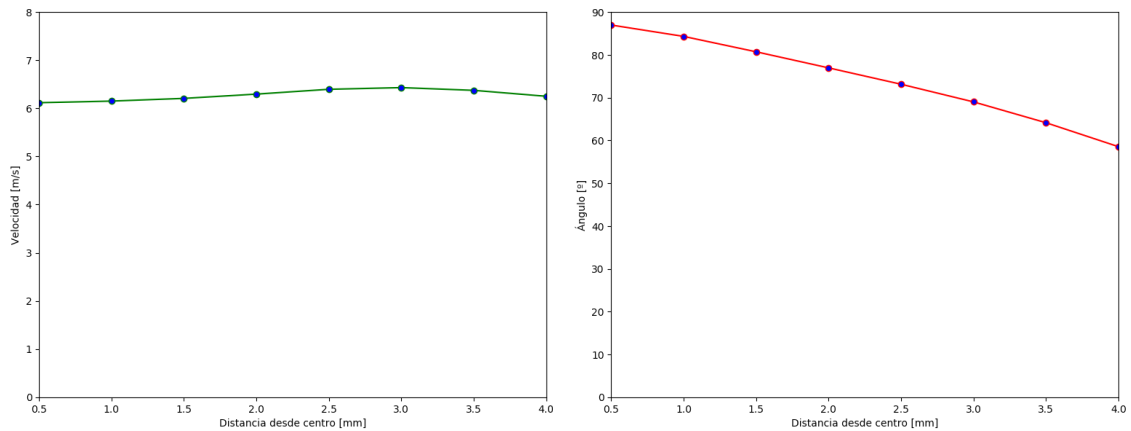


Figura 5.5: Gráfico de velocidades y ángulos de impacto promedios vs distancia desde el centro del target (5 mm).

El ángulo de impacto promedio se comporta de forma parecida en Figura 5.6, mientras que la velocidad promedio aumenta levemente a medida que se aleja del centro hasta el radio 2,5 [mm] y disminuye de forma muy parecida a los ángulos hasta alcanzar aproximadamente 5,34 [m/s]

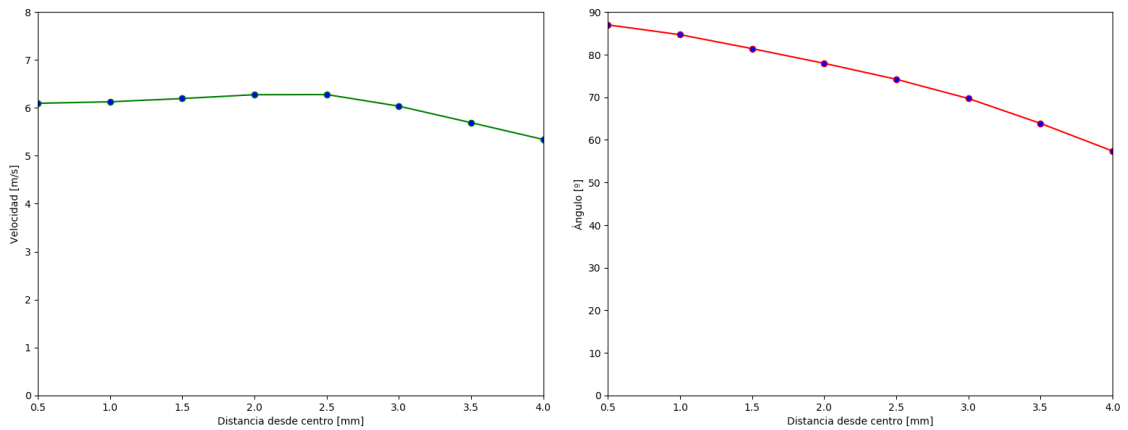


Figura 5.6: Gráfico de velocidades y ángulos de impacto promedios vs distancia desde el centro del target (25 mm).

Como se explicó en la sección 3.4.3, se extrajeron los resultados de A. Lopez [1] y se visualizan junto los resultados obtenidos por las simulaciones hechas como se ve en las Figuras 5.7 y 5.8. De esta forma, se busca comparar la velocidad y ángulo de impacto promedios sobre la superficie que se obtuvieron

Junto a esto, de manera similar como hizo el autor con los resultados de promedio de velocidad y ángulo de impacto, se calcula el error porcentual de entre cada valor respecto a su distancia del centro del target, y se luego visualizan en Figura 5.9.

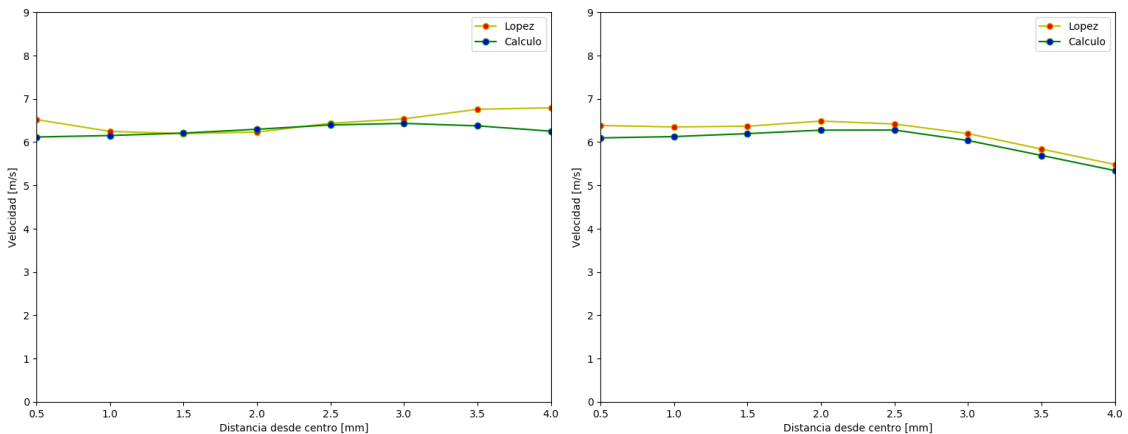


Figura 5.7: Gráfico de velocidad promedio vs distancia desde el centro, comparando resultados obtenidos con los esperados.

Se puede observar, los resultados no fueron exactamente iguales a los obtenidos por A. López en la referencia [1], pero a través del error relativo porcentual, se destaca que tanto en velocidad como en ángulo de impacto, no se superó el 10 % de error. Por consiguiente, tal como hizo el autor al comparar sus resultados, se puede considerar que estos son aceptables.

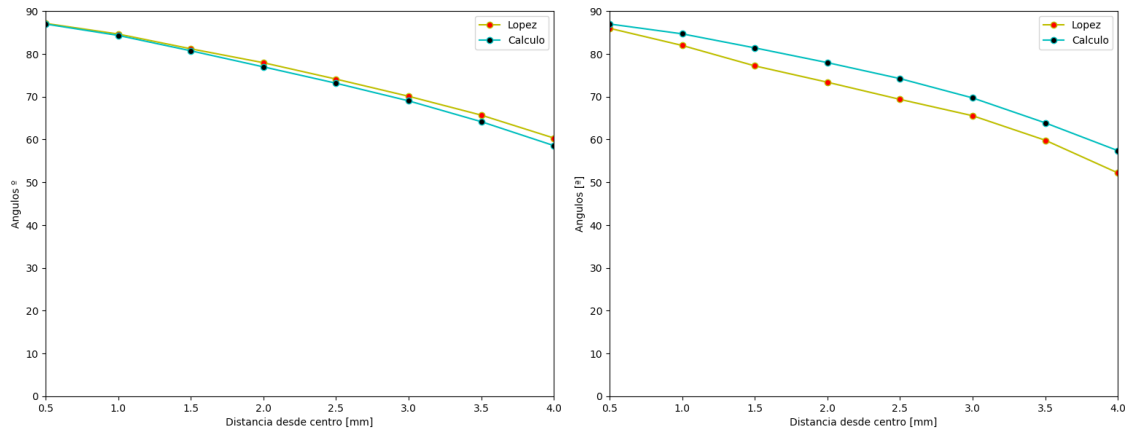


Figura 5.8: Gráfico de ángulo de impacto promediado vs distancia desde el centro, comparando resultados obtenidos con los esperados.

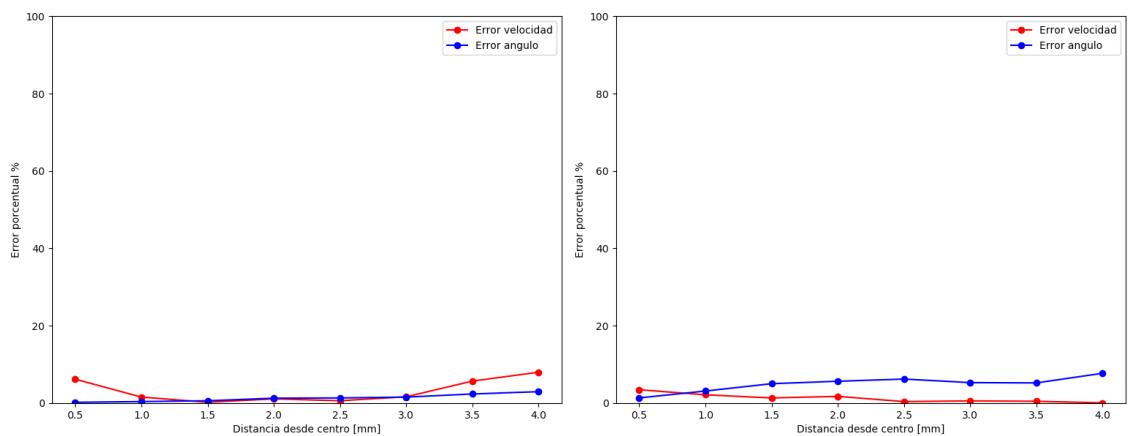


Figura 5.9: Error % entre los resultados esperados y obtenidos (JIT 5mm izquierda y JIT 25mm derecha).

5.2. Segundo caso de validación de código:

Se tomaron los gráficos presentados por A. Lopez [8] para poder compararlos con el perfil de erosión obtenido en la simulación de 5 mm. Cabe destacar, que de manera similar a lo que ocurre con el primer caso de validación, debido a la falta de información, también se hicieron ajustes de manera arbitraria en la simulación hecha para este informe.

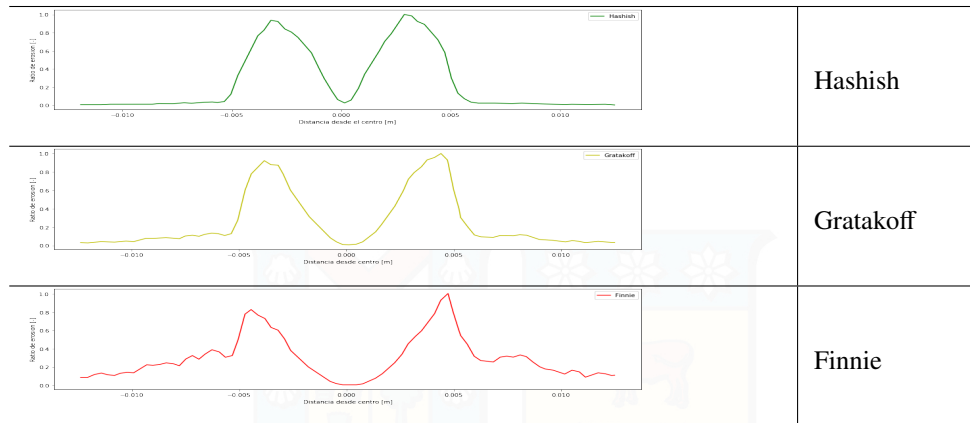


Tabla 5.1: Resultados de erosión obtenidos por las simulaciones de diferentes modelos realizadas en [8], re-graficadas en Python.

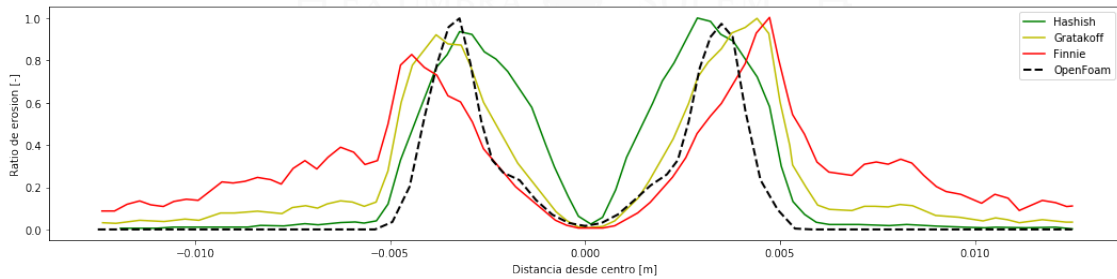


Figura 5.10: Comparación de resultados de perfil de erosión obtenidos en OpenFoam®, con los obtenidos por los distintos modelos simulados por A. Lopez.

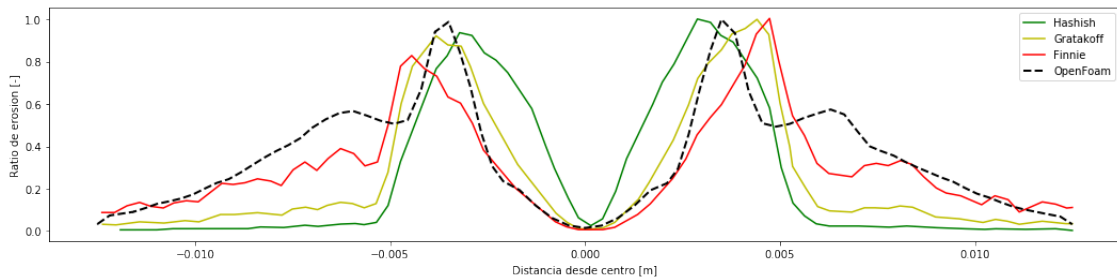


Figura 5.11: Comparación de resultados de perfil de erosión (con rebote en el target) obtenidos en OpenFoam®, con los obtenidos por los distintos modelos simulados por A. Lopez.

En la Figura 5.10 se puede apreciar que el perfil de erosión es cualitativamente muy similar al de los otros modelos, considerando que este no toma en cuenta el rebote de las partículas en el target. Se puede observar que predomina la pérdida de material por corte que es la que se ve en ángulos de impacto menores a 90° (común en materiales dúctiles) y por eso se distingue más notoriamente en un radio de $1 - 5 [mm]$. Por otro lado, contrario a lo que se ve en el centro del material no se ven efectos notorios de erosión, específicamente en el modelo de Finnie es porque no está diseñado para predecir erosión por impactos en ángulos cercanos a 90° respecto a la superficie, que es donde predomina la erosión por desgaste (predominante en materiales frágiles). Se puede argumentar que el material tiene mejor resistencia a la impacto por lo que tampoco hay efectos notorios de erosión en el centro en los otros modelos en relación a la erosión por corte. Por otra parte, en radios mayores a $5 [mm]$ no se ve desgaste en el perfil obtenido ya que las partículas no entran en contacto

con el material ya que no se dispersan más con el flujo y debido a que no tienen rebote en esa simulación, sino que desaparecen del dominio una vez que entran en contacto con la superficie del target.

En la Figura 5.11 se puede apreciar que el perfil de erosión se expandió con respecto a lo visto en Figura 5.10 dando a entender que por rebote las partículas tienen la posibilidad de impactar más de una vez en el target, removiendo más material en radios superiores. Por otro lado, ambas figuras, al ser simulaciones en base del modelo de Finnie, tienen como similitud, la zona central de la superficie, donde no se visualiza erosión por impactos en ángulos cercanos a 90° respecto la superficie.

5.3. Tercer caso de validación:

Para simular la propuesta de banco de prueba, se debe especificar primero el dominio necesario. En base al dominio utilizado en simulaciones anteriores, y al dominio utilizado por los autores V.B. Nguyen (Referencia [29]) y A. Mansouri (Referencias [24] y [25]), para la propuesta de un nuevo dominio computacional, lo primero que se hace es reducir la cantidad de celdas que rodean la pared interna por donde pasa el flujo particulado, quedando solo a 20 [mm] por encima de la salida del flujo como se ve en la referencia [24], mientras que se reducen las celdas debajo del target, dejando sin borde inferior a este, y el outlet inferior se reduce a las celdas que rodean la muestra. Lo segundo, es quitar la condición de borde de las paredes externas y dejarlas solo como salida de presión. Entonces los patches de externalWall, topOutlet y bottomOutlet se transforman a una sola condición de borde pressureOutlet (salida de presión).

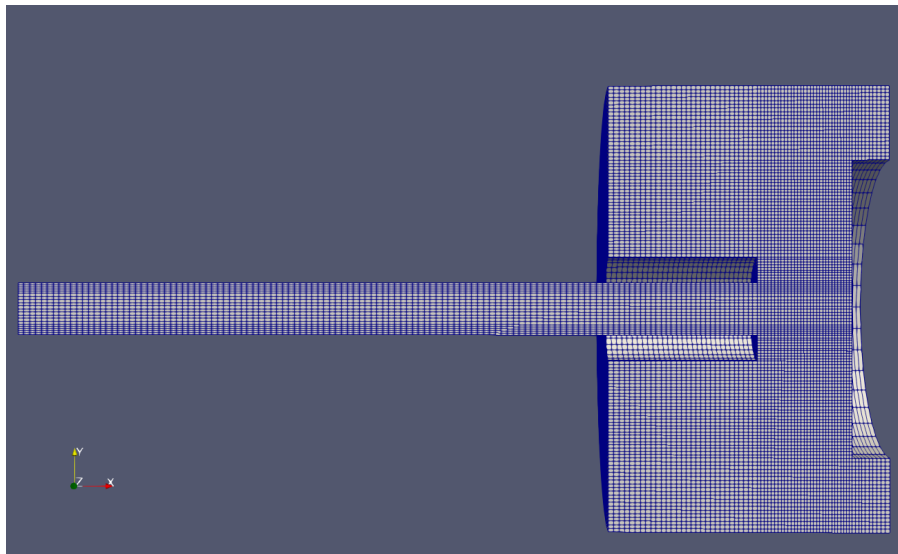


Figura 5.12: Dominio computacional y mallado para simular banco de pruebas.

El resto de las condiciones de borde e iniciales se mantienen igual. Con esto se procede a simular nuevamente el primer caso de validación, con ambas distancias.

Al volver a simular con el nuevo dominio con las condiciones de distancia de la tobera de 5 y 25 milímetros, los campos de velocidad con el dominio y mallado propuesto no varían notoriamente de sus contra-partes salvo por los costados. En la Figura 5.13 se observa que se obtienen resultados muy parecidos a lo obtenido con el dominio computacional anterior, habiendo solo diferencias decimales y centesimales. Para apreciar con mayor detalle numérico se pueden ver la tablas con los datos de los gráficos en el Anexo A.3

Cabe destacar, que las simulaciones tardaron en promedio 3 horas menos de lo que se tardaba con los mallados anteriores. Esto se debe a la menor cantidad de celdas necesarias para formar el dominio

computacional, pero manteniendo el mismo refinamiento de las celdas cercanas a las paredes con las que interactúan con el flujo.

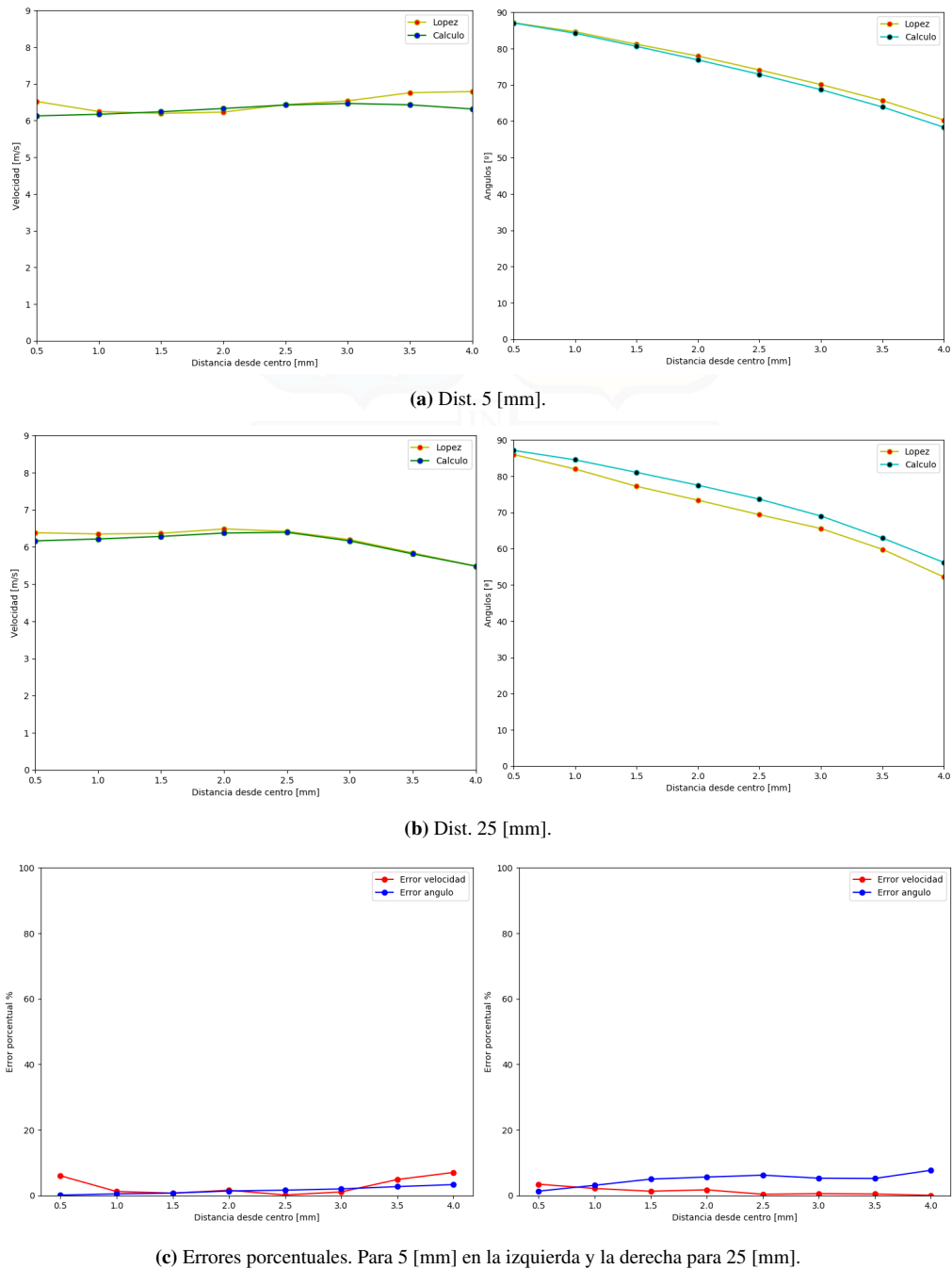


Figura 5.13: Revalidación de resultados con nuevo dominio computacional.

Se modifica nuevamente el dominio dejando la distancia de salida de flujo y la muestra con un valor de 12,7 [mm] como se ve en las referencias [24, 16], que tienen geometrías similares a la propuesta (Figura 5.12). En base a estos artículos se cambian otros pocos parámetros computacionales para realizar otras simulaciones con este dominio, y poder visualizar que efecto pueden tener el perfil de erosión (el cual se predice en base al modelo de Finnie). En el directorio 0 se modifica la velocidad del flujo a 14 [m/s] y en base a esto se recalculan los valores iniciales de κ y ϵ . En `kinematicCloudProperties`, se cambia el

diámetro de partícula a $300\ [\mu\text{m}]$, su densidad a $2650\ [\text{kg}/\text{m}^3]$ y se estima el número de partículas en base a estos nuevos parámetros pero manteniendo la fracción másica de partícula alrededor de 1 % (lo que implica 0,5 % al simular la mitad de un flujo). Se procede a simular un segundo de flujo para visualizar el campo de velocidades.

En Figura 5.14 se ve la simulación hecha para distancia salida del flujo-muestra de 12,7 [mm], junto con el campo de velocidad obtenido en la referencia [24] y se puede observar que es más importante que se simule bien el punto de estancamiento y la zonas de mayor y menor velocidad. Esto se puede corroborar mejor, al rehacer los gráficos de velocidad y ángulo de impacto promedios como lo visto en la sección 5.1.3, obteniendo lo que se ve en las Figura 5.13, mostrando casi nula variación resultados y por ende casi nula diferencias de los errores porcentuales obtenidos, respecto a los calculado en el capítulo de Verificación y Validación.

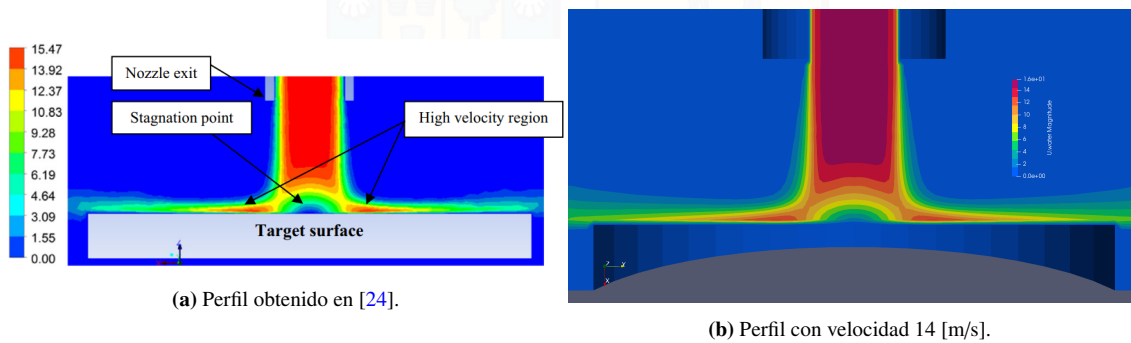


Figura 5.14: Revalidación de resultados con nuevo dominio computacional.

Luego se realiza una simulación para los siguientes valores velocidad del flujo: 3, 5, 8, 12 y 16 [m/s]. En base a esto, nuevamente se calculan los valores iniciales de κ y ϵ y se estima la cantidad de partículas para cada caso, manteniendo la fracción de masa sólida. Además se desactiva la función `particleTrack` ya que no es necesario y se ahorra memoria del PC en el que se simula, solo se deja la función `particleErosion`, cambiando el valor de p a 209 [MPa] para que la muestra simule acero inoxidable como material.

En la Figura 5.15 se pueden observar los perfiles de erosión obtenidos de las simulaciones considerando con las condiciones vistas en la referencia [16]. Estos perfiles se graficaron en Jupiter Notebook (Python), recordando que son en unidades volumétricas de material perdido, de forma similar a lo visto con el segundo caso de validación de código se adaptan respecto al valor máximo alcanzado, ya que lo que más interesa es el perfil de erosión producido a lo largo de la línea central del material de muestra. Entonces, haciendo solo un análisis cualitativo, se ve que efectivamente que la profundidad de erosión aumenta sobre el material, si se incrementa solo la velocidad del flujo, si mantienen constantes los demás parámetros. Recordando la Figura 1.4, en vista y consideración de que en este caso se utilizó OpenFOAM en vez de Ansys, y que en este último su modelo de erosión entregaba resultados en profundidad de material perdido en vez de volumen, se ve que hay una gran similitud en la forma de los perfiles. Por otro lado, se puede observar que el valor máximo de erosión ocurrió en un radio menor a 5 [mm] como en la Figura 1.4. Sin embargo, los puntos más altos obtenidos en OpenFOAM se encuentran cercano al centro, lo cual se puede deber a que otra diferencia fue que se consideró condición de escape, mientras que en la referencia [16] se consideró condición `reflect` (que es el equivalente a `rebound` en OpenFOAM), entonces, recordando lo visto en la sección 5.2, el que las partículas reboten sobre la superficie da pie a que impacten más de una vez y expandan el perfil de erosión, lo cual también explica porque en la Figura 5.15 hay cero erosión en radios mayores a 5 [mm] a diferencia de la Figura 1.4.

En síntesis, los resultados obtenidos en los dos primeros casos de validación son lo suficientemente parecidos a los obtenidos por A. Lopez en las referencias [1] y [8]. Cabe recordar que el autor en la referencia [1] se enfocaba más en validar los resultados con otro software llamado Ansys Fluent, por lo que obtener resultados con un margen de error porcentual menor al 10 % se puede considerar aceptable. Mientras que por otra parte, en la referencia [8] se enfocaba en la comparación cualitativa ya que los modelos de erosión no entregaban resultados con las mismas unidades, por lo que era necesario estandarizarlas para poder

compararlas, en este caso visualmente se obtienen un perfil parecido a lo que se espera al aplicar el modelo de erosión de Finnie con flujo mezcla líquido-sólido, en una placa plana, incluso considerando que no se podía replicar el modelo de rebote y solo se utilizaba el que estaba por default en OpenFOAM 8.

En el tercer caso de validación se hicieron algunos cambios para asemejar las simulaciones hechas por Mansouri [24] (que utilizó Ansys Fluent) obteniéndose un perfil de velocidades muy parecido, tanto en valores de velocidad como en la ubicación del punto de estancamiento y las regiones de altas velocidades en la imagen. Junto a esto los resultados de perfil de erosión, considerando que no se aplicó condición de rebote, son lo suficientemente similares en los puntos más altos y más bajos de erosión para considerar que el código de OpenFOAM 8 es confiable para aplicar en otras simulaciones de erosión de pruebas experimentales con características relativamente diferentes a las iniciales (presentadas por A. Lopez). Siempre teniendo en cuenta que se puede ir mejorando y agregando características que ayuden a mejorar el realismo, como por ejemplo un modelo de rebote.

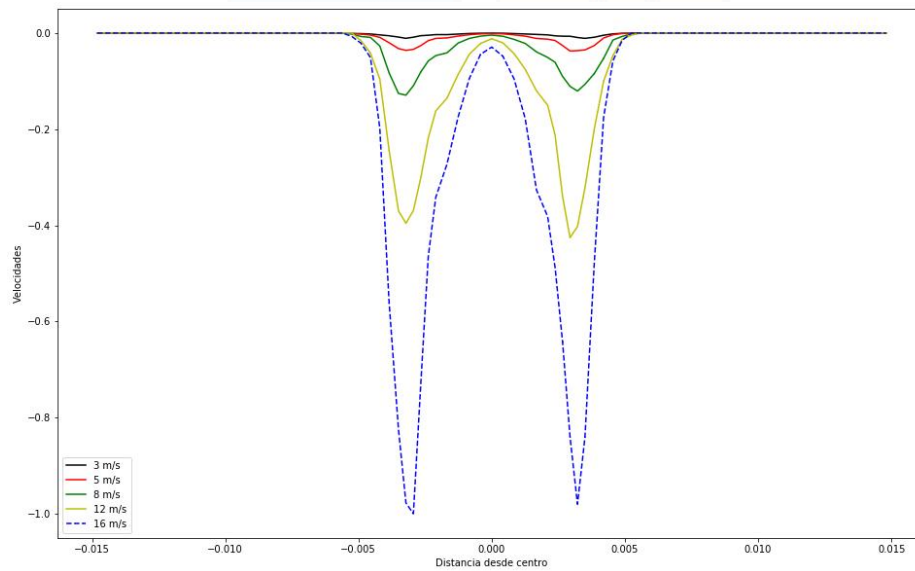


Figura 5.15: Perfiles de erosión obtenidos al variar la velocidad de entrada del flujo.

(Fuente: Elaboración propia en Jupiter Notebook)

6 | Proyecto de banco de pruebas para erosión

6.1. Definición del problema y Objetivos

Se requiere diseñar un banco de pruebas de erosión, lo cual hay una gran variedad dependiendo de que parámetros se busca estudiar y que tipos de resultados se busca obtener, ya sea masa erosionada, el perfil de erosión, la velocidad de partículas entre otras. En orden de acotar, se define una lista de atributos con la que se busca esclarecer lo que se espera que cumpla este diseño. En primera instancia que sea posible realizar pruebas experimentales de las simulaciones ya realizadas, mientras que en segundo lugar, se permita cierta libertad de modificar algunos parámetros para simulaciones/pruebas futuras.

En la Tabla 6.1 se escriben los atributos más generales del banco que se busca diseñar junto con la clasificación de cada uno dentro de las tres siguientes categorías:

Objetivos : Atributos que debe tener el diseño y posteriormente facilitan su desarrollo.

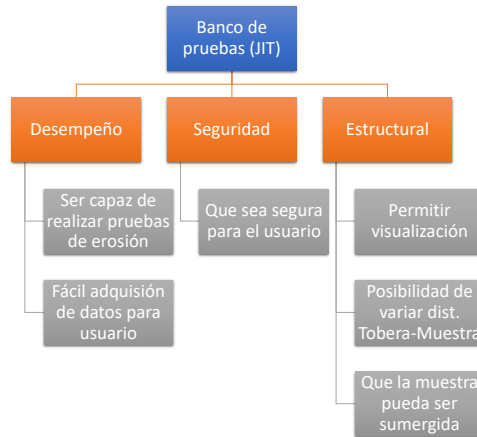
Restricciones : Atributos que deben ser considerados obligatoriamente.

Funciones : Son las acciones que debe realizar el banco.

Tabla 6.1: Lista de atributos y categorización

Atributo	Categoría
Capacidad de realizar pruebas de erosión exitosas	Objetivo
Banco con fines académicos	Restricción
Que la muestra tenga la posibilidad de estar sumergida o al aire libre	Objetivo
Visualización de la prueba	Objetivo
Inyección de partículas sólidas	Función
Montaje y desmontaje de muestras sencillo para el usuario	Objetivo
Regular el caudal del flujo	Función
Variación la distancia entre la salida de tobera y la muestra	Objetivo
Desplazamiento del portador de muestra	Función
Adquisición sencilla de datos para el usuario	Objetivo
Que sea segura para el usuario	Objetivo
Realizar pruebas de erosión	Función

Por otra parte, la Figura 6.1 se presenta el árbol de objetivos que se consideran esenciales y separados respecto a las características de Desempeño, Seguridad y Estructura.

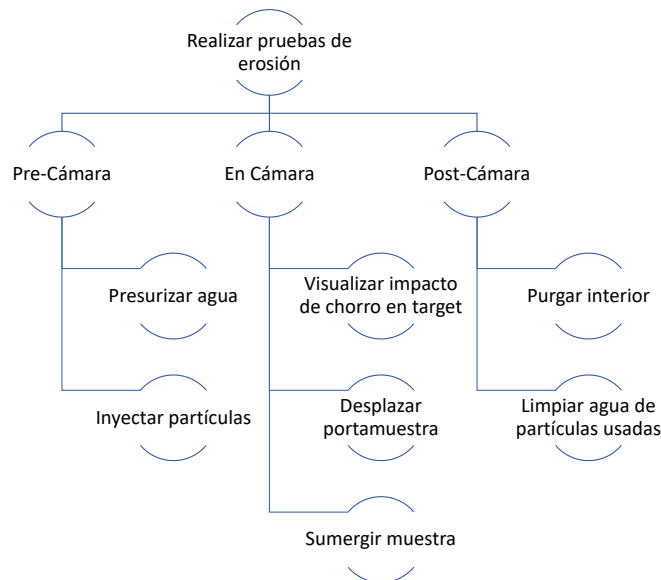
**Figura 6.1:** Árbol de objetivos.

(Fuente: Elaboración propia)

6.2. Estructura funcional

La Figura 6.2 muestra la herramienta árbol de funciones, dividiéndolas en tres categorías según lo que ocurre antes, durante y después de la cámara de pruebas.

Por otro lado, en la Figura 6.3 se puede apreciar el diagrama Caja Negra, donde los principales flujos de entrada son de energía eléctrica, flujo de agua y material particulado junto con el material de muestra. De esto debería salir disipada vibraciones y ruido, debería salir la muestra erosionada y la mezcla de agua con partículas.

**Figura 6.2:** Árbol de funciones.

(Fuente: Elaboración propia)

**Figura 6.3:** Diagrama Caja negra.

(Fuente: Elaboración propia)

6.3. Generación de conceptos

Prosiguiendo con la siguiente fase de diseño, se busca ser más específico con lo que se busca hacer con este diseño. Dejando más claro que es lo más esencial y necesario que se debe cumplir, junto con lo que se desea cumplir, y finalmente lo es solo opcional respecto a los objetivos primarios del diseño. Se puede apreciar de forma resumida en la Tabla 6.2.

Se utilizó la herramienta Brainstorming (Lluvia de ideas) como se puede apreciar en la Figura 6.4. El centro de la lluvia de ideas es el banco de pruebas de erosión y se exploran las posibilidades de componentes que cumplen con las especificaciones.

La disyuntiva que más se destaca, tomando en cuenta lo visto en la Figura 1.7 es si esta debe ser pre-mezclado o post-mezclado. En la referencia [3] se da a entender que el método de Turenne es el más popular entre investigadores representando un 54,55 %, dando a entender que es más sencillo de adaptar a nuevos diseño las características de re-circulación (RC) y de pre-mezclado. No obstante, al ser este diseño netamente conceptual, que busca hacer pruebas experimentales directamente relacionadas a las simulaciones hechas en OpenFOAM®, se considera necesario el banco de pruebas sea de tipo NRC, es decir que no haya re-circulación de partículas, (independiente si se reutiliza el agua o no), De esta forma hay mayor posibilidad de que las partículas mantengan su forma/redondeo y por consiguiente se mantengan las condiciones de prueba que busca hacer tanto experimental como computacionalmente.

Teniendo definido que el diseño necesita ser de tipo NRC, siguiendo aun con la disyuntiva pre o post mezclado, ya que la mayoría que se han visto de premezclado una bomba (slurry pump) para bombear una mezcla almacenada en un estanque, haciendo necesario el uso de un agitador para que esta se mantenga lo más uniforme posible, pero gastando de todas formas el rodete y voluta de la bomba y al mismo tiempo cambiando la forma inicial de las partículas cuando son bombeadas, mientras que en postmezclado se puede evitar este problema, pero se vuelve complejo la forma que se inyectan las partículas ya que esto se vuelve más difícil si el flujo transportador ya se encuentra presurizado. Sin embargo, de la lluvia de ideas, se puede concluir que hay varios aspectos que se pueden solucionar independiente de si el diseño es pre o post-mezclado, pudiéndose visualizar en la Figura 6.5.

Comenzando con la presurización del agua, se opta por una bomba peristáltica. Para esta bomba se decidió por un modelo SPX (bomba peristáltica), tomando inspiración del artículo del autor V. Nguyen [29], quien utiliza un banco de tipo premezclado y RC, pudiéndose ver esta misma configuración en las

referencias [30] y [31] de este mismo autor.

Tabla 6.2: Lista de especificaciones de diseño y categorización

Especificaciones de diseño	Categoría
Presurización del agua	<i>Necesario</i>
Caja de pruebas que pueda llenarse de agua o estar vacía	<i>Necesario</i>
Tener un portador de muestras	<i>Necesario</i>
Portador de uso sencillo	<i>Necesario</i>
Cambiar la distancia entre la tobera y la muestra	<i>Necesario</i>
Cambiar el ángulo de impacto del chorro	<i>Deseado</i>
Tener re-circulación de agua	<i>Deseado</i>
No reutilizar las partículas una vez chocan contra la muestra	<i>Necesario</i>
Limpieza sencilla	<i>Deseado</i>
Visualización de prueba	<i>Necesario</i>
Tobera intercambiable de diámetro	<i>Deseado</i>
Medidor/Regulador de caudal	<i>Necesario</i>
Inyección de partículas para que se mezclen con el agua	<i>Necesario</i>
Posibilidad de variar la concentración de la mezcla	<i>Deseado</i>
Adquisición de datos de muestra erosionada	<i>Deseado</i>
Posibilidad de cambiar la geometría de la muestra en prueba	<i>Deseado</i>
Banco de prueba agradable a la vista	<i>Opcional</i>
Tamaño compacto	<i>Opcional</i>
Diseño modular	<i>Opcional</i>

La bomba peristáltica es un tipo de bomba que consiste en una manguera siendo presionada por un brazo excéntrico con rodillos. La principal ventaja de esta bomba es que el flujo particulado no entra en contacto con los componentes de esta, salvo la manguera hidráulica principal por lo que el flujo no produce desgaste en su interior. La marca de esta bomba, Bredel SPX posee una variedad de tamaños dependiendo de la necesidad de uso. Para este banco, en base a las simulaciones hechas, se considera en primera instancia en que el flujo saldrá por una tobera con 7 [mm] de diámetro, con lo que arbitrariamente, para una velocidad en el rango de 10 – 20 [m/s], con lo que se estima que el caudal variaría entre 1500 y 3000 litros por hora, por lo que se recomendarían los modelos SP32 o el modelo SP40. Se puede ver en sus respectivas curvas potencia (Figura 6.6) que cumplen con el rango de caudal. Junto a esto, el fondo verde (según el manual que se encuentra en la web) significa que pueden entregar ese caudal de manera continua (mientras que el fondo amarillo significa que solo podría bombear ese caudal por 2 horas y luego debe quedar al menos 1 hora sin usarse). Se estima que la presión de trabajo de la bomba no requiere ser mayor a 500 [kPa] por lo que se estima que se en ambos casos se requeriría una potencia cercana a 1 [kW]. Esta bomba al igual que el lugar de testeo se encuentra apoyado sobre un armazón de metal (por ejemplo fierro fundido), el cual se puede diseñar según la necesidad/disponibilidad de espacio. Las dimensiones de ambas bombas se pueden ver en milímetros en las Figuras A.5 y A.6 en Anexo.

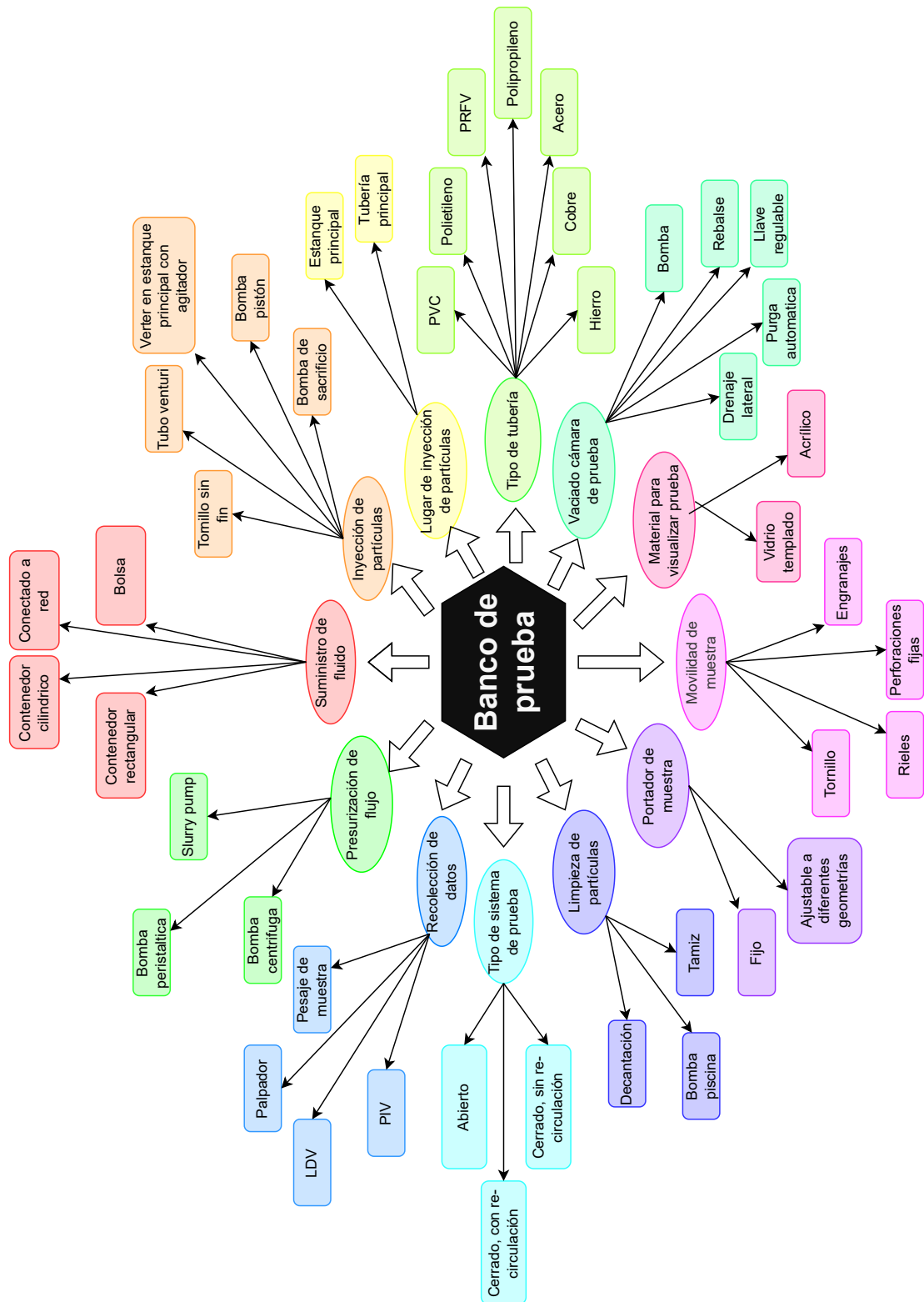


Figura 6.4: Diagrama Lluvia de ideas (Brainstorming).
(Fuente: Elaboración propia)

En búsqueda de evitar un gasto excesivo de agua, se opta por que esta sea suministrado por un contenedor principal, el cual para aprovechar mejor el espacio se decide por una forma rectangular, de igual manera que se opta por un segundo contenedor con la misma forma que reciba el agua. Como medida de volumen para el diseño conceptual se deja un metro cúbico al igual que el volumen que tiene la cámara de prueba.

Prosiguiendo con la cámara de prueba, además de diseñarse con dimensiones 1x1x1 [m], para que haya suficiente espacio para que el chorro solo choque contra la muestra, se optan por acrílicos ya que otorgan suficiente transparencia para visualizar la prueba, son menos frágiles que el vidrio y se puede sellar usando silicona en los bordes. Otro aspecto importante de la cámara de prueba, es que esta hecha con una base de aluminio (u otro metal relativamente liviano pero resistente) y 3 paredes hechas de acrílico ya que estas permiten suficiente transparencia para visualizar la prueba. Mientras que una de las paredes es completamente cerrada y tiene un orificio para dejar pasar la tubería que conecta la salida de la bomba con la tobera. Para esta tubería se opta por utilizar PVC, ya que es un material relativamente barato, reemplazable (ya que cualquier tubería se gastará por el paso de flujo con partículas, otro motivo por que mejor que se deforme este material antes que las partículas) y es común para tuberías con agua ya que tiene un factor de rugosidad relativamente bajo que da menos dificultad de desplazarse al líquido.

Por otra parte, dentro de la cámara de muestra, se instalan unos rieles por donde se puede desplazar el portador de objetos. Como se aprecia en la Figura 6.7, por medio de dos tornillos se asegura en una posición (por lo que es ideal que haya una graduación milimétrica junto a los rieles), en base a las simulaciones, con una distancia mínima de 5 milímetros de la salida de la tobera y dando un rango de 170 milímetros de esta misma. Cabe destacar que la tobera en este diseño se modeló en base a las simulaciones hechas (diámetro de salida de 7 [mm]), en base al diámetro de tubería del modelo de bomba SPX Bredel (44 [mm]) y de material tungsteno tomando como referencia el artículo [32].

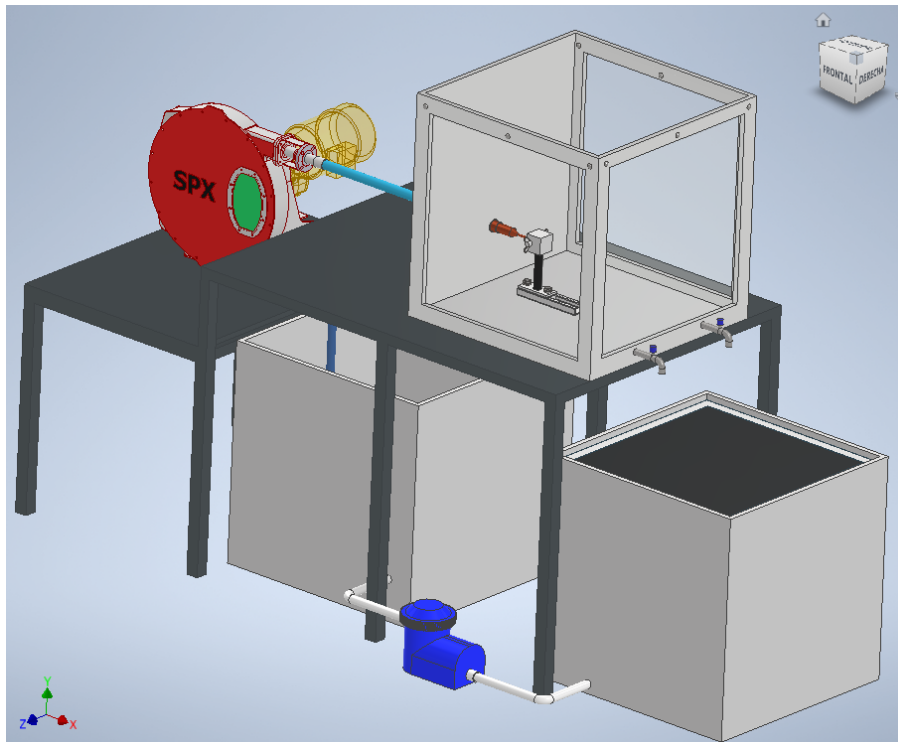
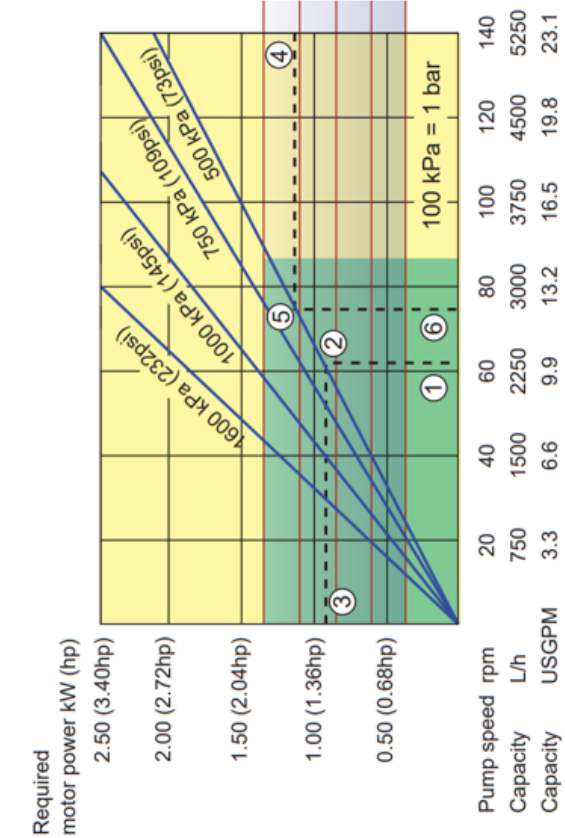


Figura 6.5: Diseño conceptual 3D de banco de pruebas.

(Fuente: Elaboración propia en Autodesk Inventor)

BREDEL 32



BREDEL 40

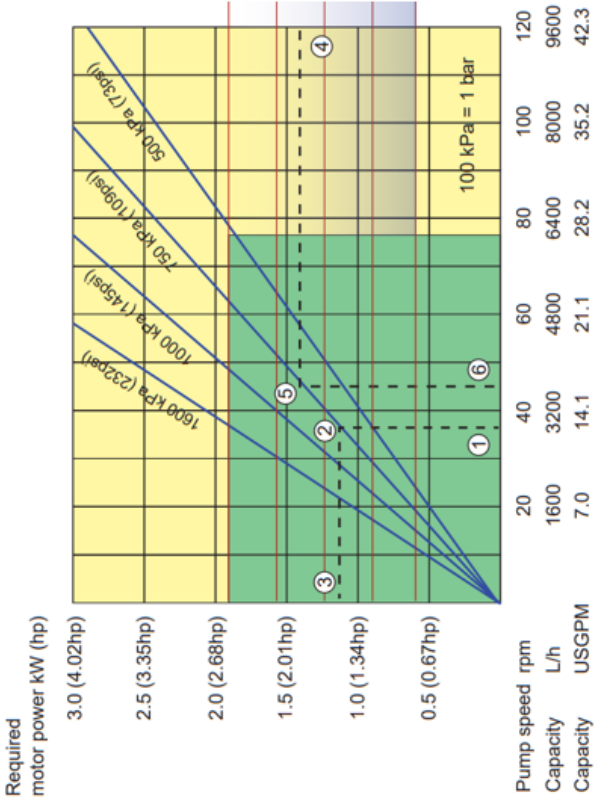


Figura 6.6: Curva Potencia de Bombas Bredel SP32 y SP40.
(Fuente: Catálogo Bredel)

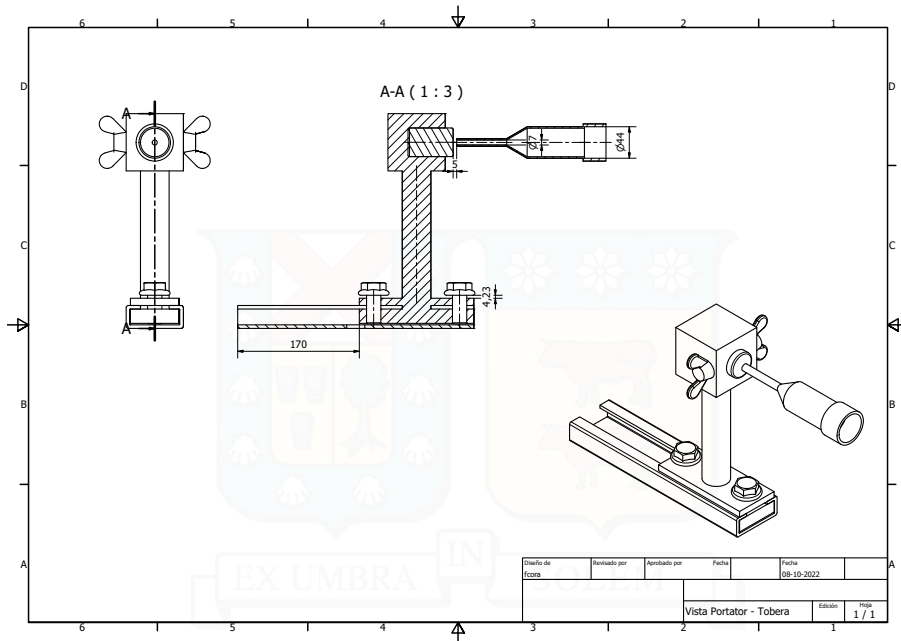


Figura 6.7: Vista Portador de muestra - Tobera.

(Fuente: Elaboración propia en Autodesk Inventor)

Como se mencionó previamente hay un estanque receptor del agua que salga de la cámara de prueba. Se optó por instalar llaves de purga con caudal regulable. Para limpiar el agua, se tomaron dos medidas, primero el agua que cae desde la cámara de prueba pasa por un tamizado sobre el estanque receptor, mientras que este en la parte inferior está conectado por medio de una manguera al estanque principal y el agua es bombeada hacia este por medio de una bomba de piscina que termina por filtrar el agua y dejarla lista para ser reutilizada.

Respecto a la recolección de datos, se pueden utilizar todas las opciones según desee el usuario. Pero la más relevante sería el Método PIV que sirve para obtener el perfil de velocidad del flujo transportador (Para esto se necesitaría mezclar el flujo con partículas más pequeñas y livianas que puedan seguir las líneas de corriente) y para estimar las velocidades y ángulos de impacto de las partículas que son dos de los parámetros más relevantes en más de un modelo matemático de predicción de erosión (como por el ejemplo el modelo de Finnie que se simuló en el segundo caso de validación). Tal como se mencionó en estado del arte, el modelo de LDV también puede ser útil para visualizar partículas, pero este presenta problemas cuando estas impactan en una superficie debido a como llega el láser a la muestra.

Finalmente, se proponen dos variaciones en el diseño conceptual dependiendo de si son pre o post-mezclado:

Banco NRC tipo pre-mezclado:

Para que este banco de pruebas sea de tipo pre-mezclado, pero siga siendo también NRC, entonces la mezcla debe hacerse en el estanque principal, agregando el volumen total de partículas necesario para la concentración que se busca. Por ejemplo, para un 1 metro cúbico de agua, si se busca concentración volumétrica de 0,00904 [-] de partículas cuya densidad y diámetro son 2206 $[kg/m^3]$ y 250 $[\mu m]$ respectivamente, se requerirían aproximadamente 20,15 [kg] de estas partículas.

Luego, para evitar la sedimentación de las partículas, es necesario instalar un agitador en el estanque (Figura 6.8), el cual mantenga la mezcla partícula-agua lo más uniforme posible. De esta forma, se bombea la esta mezcla directamente hacia el material de muestra.

La primera desventaja que se presenta es el costo adicional de instalar un agitador. La siguiente desventaja es que para no se reutilicen las partículas, el agua se debe filtrar y devolver al estanque principal,

como se mencionó previamente, pero no al mismo tiempo que ocurre la prueba ya que eso cambiaría la concentración de la mezcla sólido-líquido que se busca estudiar, por lo que el tiempo de prueba se limita al tiempo que el estanque con la mezcla tarde en vaciarse, o estar muy cerca de esto (para evitar problemas con la bomba). Luego de esto se rellena el estanque principal con agua filtrada, y se realiza otra mezcla en caso de que se quiera alargar el tiempo de prueba del material en cuestión.

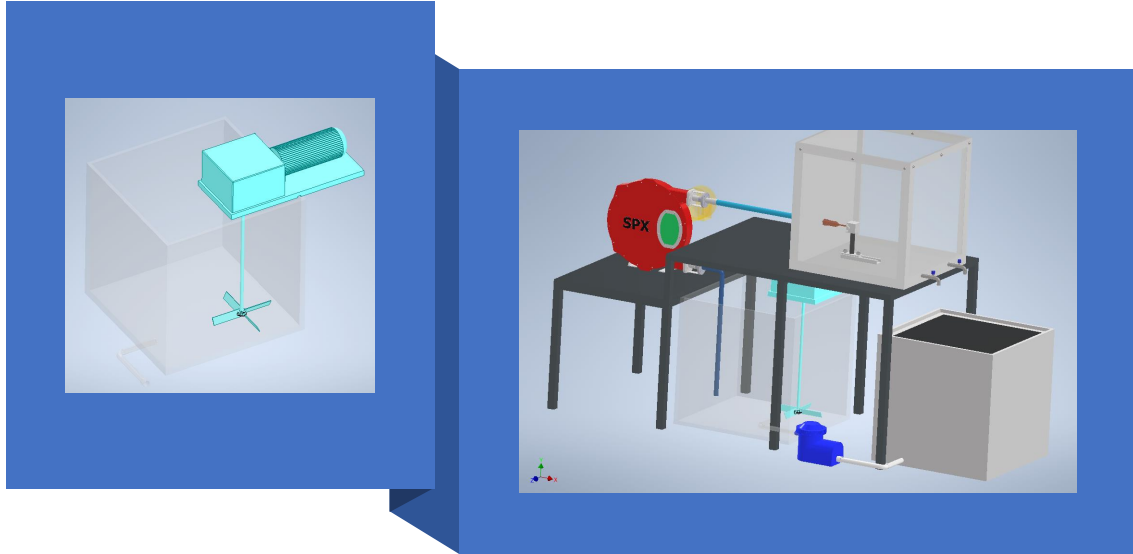


Figura 6.8: Vista ISO de diseño conceptual con agitador incluido.

(Fuente: Elaboración propia en Autodesk Inventor)

Banco NRC tipo post-mezclado:

Recordando lo explicado en el capítulo 2, la configuración NRC es más compleja de diseñar y los artículos que la utilizan suelen dar muy poca información de su diseño haciendo aun más complicado poder replicarlas.

Por otro lado, no se recomienda utilizar un cambio de sección (o tubo de Venturi) ya que además de ser complicado su fabricación e instalación, su diseño limita mucho la co-dependencia entre la concentración y la velocidad del fluido, dando pie a en muchos casos no haya succión de partículas y en cambio hayan fugas de líquido debido a la contra-presión que se forma. Respecto a esto se puede ver el ejemplo de la Figura 1.9, el cual se terminó descartando ya que presentó fugas y contrapresiones que terminaban expulsando el flujo en vez de succionar las partículas, se concluyó que este método es más adecuado para flujos con mayores velocidades.

Entonces, siguiendo el ejemplo de los autores A. Lopez [22], H.S. Grewal [32] y A. Mackenzie [35], se opta por bombear una mezcla partícula-líquido al flujo luego de salir de la bomba principal, con la diferencia de que de en vez de usar una bomba de "sacrificio", se recomienda usar otra bomba peristáltica, que tal como se explicó previamente, tiene la ventaja de que solo la manguera interior sufriría desgaste, mientras el resto de sus componentes no entra en contacto con el flujo de erosión.

Por otro lado, Hay una diferencia respecto a la distancia de la tobera donde se inyectan las partículas. Como se puede apreciar en la Figura A. Lopez [22] inyecta las partículas muy cerca de la salida de la tobera (Figura 6.9), lo que no es conveniente según explica H.S. Grewal [32], en su artículo describe que se requiere lo que se conoce como Tubo de aceleración (Figura 6.10), debido a que las partículas inyectadas requieren recorrer una cierta distancia para mezclarse de manera uniforme en el flujo transportado, ya que de lo contrario se forma un perfil no simétrico, tal como lo asevera también A. Mackenzie en la referencia [35] (Figura 6.11). Por consiguiente es necesario lo que se conoce un tubo de aceleración como se ilustra en

la Figura , para este diseño se estima que un metro de distancia de la tobera como mínimo (como se ve en la Figura 6.12) para que estas alcancen a mezclarse bien.

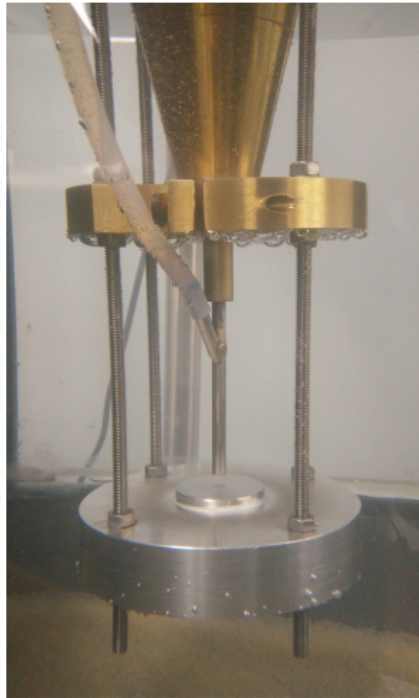


Figura 6.9: Inyección de partículas en diseño de A. López.

(Fuente: [22])

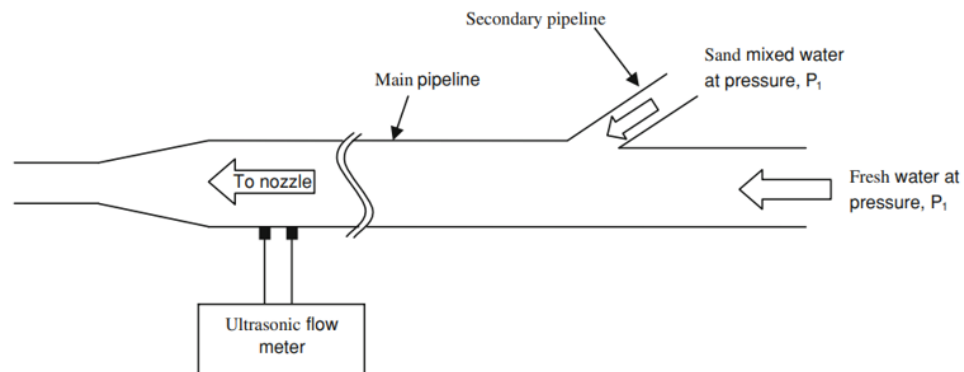


Figura 6.10: Layout Tubo de aceleración.

(Fuente: [32])

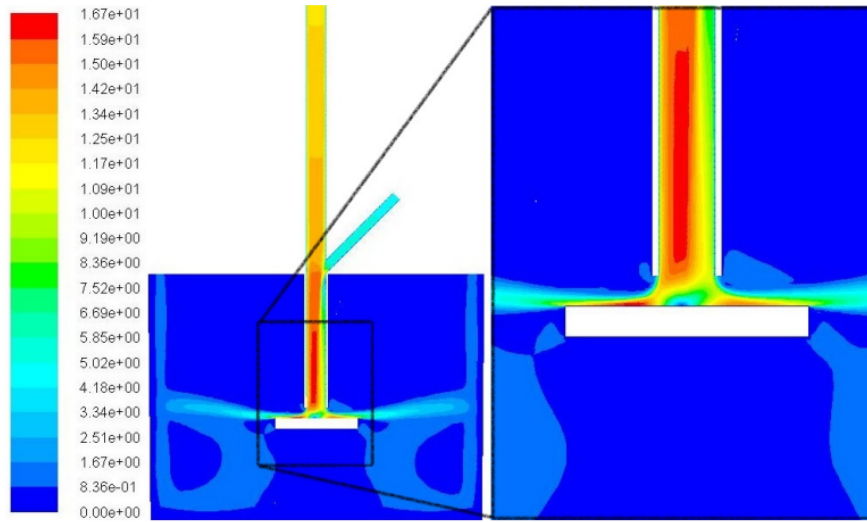


Figura 6.11: Perfil no simétrico debido a la mezcla no uniforme.

(Fuente: [35])

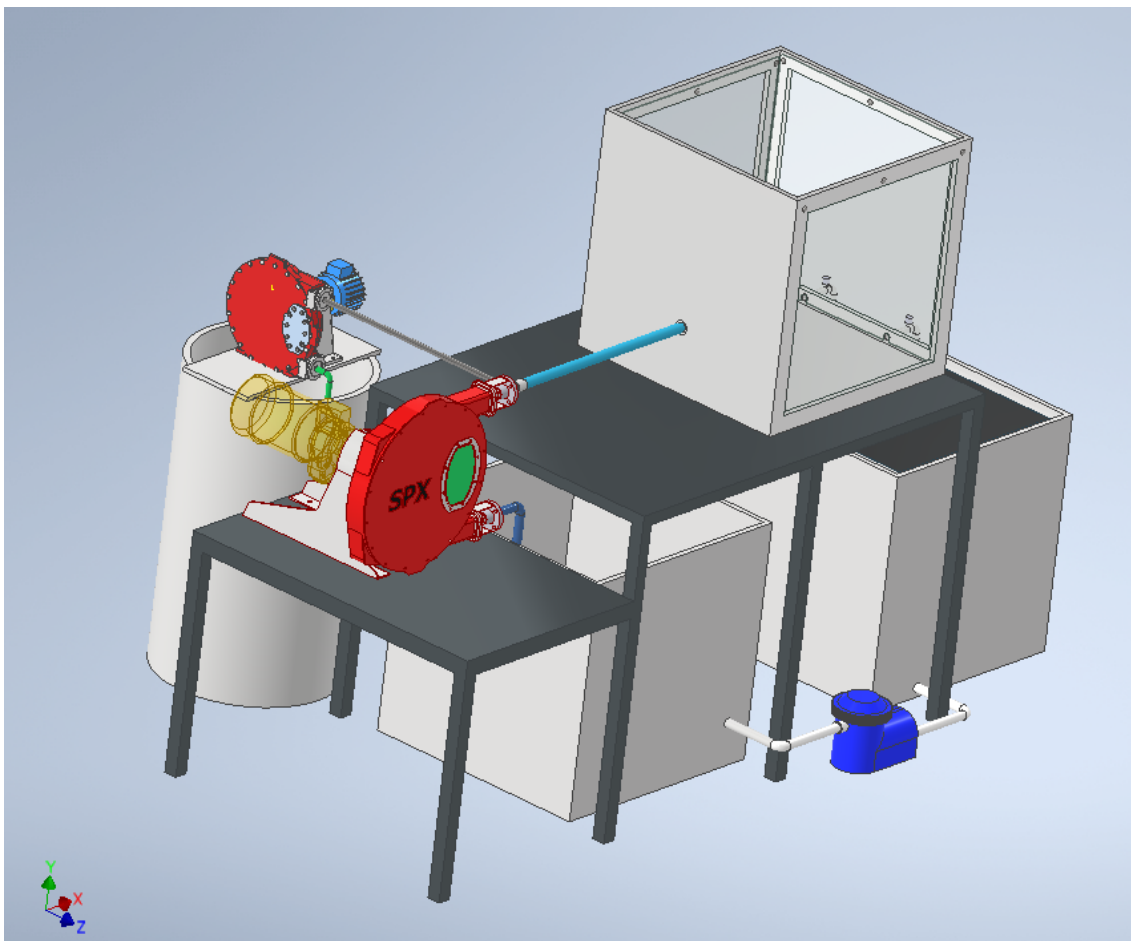


Figura 6.12: Vista ISO de diseño conceptual con bomba peristáltica extra.

(Fuente: Elaboración propia en Autodesk Inventor)

7 | Conclusiones

En el presente trabajo se consigue estudiar y aprender lo básico del software OpenFOAM 8, con la finalidad de realizar la verificación del estudio base en el que se orienta este trabajo.

Después de haber hecho una revisión de literatura, relacionada a bases teóricas de la erosión y a otros casos de simulación, específicamente de choque de chorro de agua (siendo esta un flujo diluido, o en otras palabras de baja concentración de partículas). Por consiguiente, se puede concluir que todos los parámetros mencionados se relacionan unos con los otros en mayor o menor medida dependiendo de cada caso, pero siempre se consideran relevantes en cualquier caso, la velocidad y ángulo de impacto sobre la superficie del material sometido a la erosión, ya que cualquiera de los demás parámetros o factores pueden afectar directa o indirectamente en estos y por consiguiente pueden influir en la tasa y en el perfil de erosión. Seguido de esto, en vista y consideración del tipo de flujo, mezcla, sumado a la importancia de recolectar data de velocidad y ángulo de impacto de las partículas.

Entre el enfoque numérico Euler-Euler y el enfoque Euler-Lagrange, el segundo se puede considerar como el más adecuado, siendo más específicos el modelo de partícula discreta (DPM) para flujos diluidos, es decir de baja concentración de sólidos en un flujo líquido ya que esto se acerca más a la realidad de los flujos que se usan para las turbinas Pelton. Además, el modelo de partícula discreta a pesar de llegar a ser más costoso computacionalmente que un modelo con enfoque Euler-Euler, el hecho de que el rastreo de las partículas sea con base lagrangiana permite estimar y estudiar a en mayor detalle la velocidad y ángulo de impacto sobre la superficie del target, en comparación a simular la fase sólida (diluida) como si fuera otro fluido mezclado con el flujo continuo.

Se consigue validar los resultados obtenidos de las simulaciones con los obtenidos por A. Lopez en las referencias [8] y [1]. Consecuentemente, se puede afirmar que el código funciona para una predicción aproximada de las ubicaciones, velocidades, ángulo de impacto, de las partículas y además, usando este mismo código se pudo obtener resultados coherentes respecto a la velocidad y tasa de erosión sobre la muestra. Adicionalmente, se pudo modificar parte del dominio computacional para que fuera menos costoso computacionalmente, obteniendo casi los mismos resultados de validación y en menor tiempo de simulación, para luego poder poner a prueba el código con otras velocidades para poder ver como afecta el perfil de erosión sobre la superficie del target.

Finalmente, se pudo diseñar conceptualmente un banco de pruebas de choque de chorro (Jet Impingement Test) de tipo NRC, es decir, sin recirculación de las partículas, ya que estas se fracturan, pulverizan o pierden su redondeo luego de chocar, afectando las condiciones de impacto y de concentración de la mezcla y por consiguiente el experimento se vería afectado.

No obstante, hay dos opciones respecto al mezclado de partículas con el flujo continuo. La primera es hacer la mezcla en el estanque principal y aprovechar que la bomba es de tipo peristáltica por lo que resiste mejor el desgaste producido por el flujo particulado. La principal desventaja que tiene es que limita el tiempo de uso y la necesidad de un agitador. La segunda opción es inyectar las partículas por medio de otra bomba, lo cual entregaría mayor libertad y variedad de concentración de partículas en la mezcla y permitiría que el banco funcione constantemente si se sincronizan ambas bombas con el relleno del estanque principal por medio del filtro de piscina. Sin embargo esto es potencialmente más costoso y más complejo de diseñar y de implementar que la primera opción, por lo que en primera instancia se recomendaría optar por esa

opción.

Finalmente el banco de pruebas tiene flexibilidad respecto al uso del método PIV y respecto a si la prueba se quiere hacer con la muestra al aire libre o que esta se encuentre sumergida por medio de la regulación de las válvulas de purga en el costado de la cámara de testeo.

7.1. Trabajo futuro (Further Work)

7.1.1. Modelos de erosión

Para este trabajo se estaba limitado al modelo propuesto por Finnie, el cual entrega resultados en unidades volumétricas, complicando la comparación de resultados en la literatura (ya que estos se pueden mostrar con unidades de masa, unidades longitudinales como profundidad de desgaste en el material), por lo que, por un lado, se requeriría realizar un post-procesamiento a los resultados para adaptarlos a las unidades que se buscan comparar (lo que a su vez ponen en riesgo de que aumenten los errores de calculo, aproximación, entre otros), mientras que por otro lado, en la etapa de pre-procesamiento se implemente otro modelo teórico de erosión que se adecue más a las datos y unidades experimentales con los que se busca comparar. Cabe recalcar que si es posible implementar otros modelos teóricos de erosión en OpenFOAM, como lo hizo A. Lopez [8], pero esto requiere un conocimiento del lenguaje de programación C++, y un procedimiento más complejo de lo que se busca en el alcance de este trabajo.

7.1.2. Geometría de la muestra (target)

Por un lado que el mallado se limite solo a utilizar blockMesh, a su vez limita la complejidad de la geometría que se busca dar al target. Incluso dejando la muestra con su forma estándar de disco, pero cambiando el ángulo de esta respecto a la dirección del flujo, puede provocar que algunas celdas queden con forma de tetraedros en vez de hexaedro que sería lo ideal, no obstante, esto no necesariamente impedirá que se realice la simulación, pero si limitará el refinamiento de la malla (de forma similar a lo mencionado en la sección 4.6.2). Por lo tanto, ciertos cambios en la geometría y/o su posición, pueden requerir aplicar otras técnicas de mallado y refinamiento.

Por otro lado, figuras más complejas (entre las que se puede encontrar la cuchara de una turbina Pelton) pueden requerir el uso de otros softwares como por ejemplo Autodesk Inventor para figuras en 3D, así como el uso de otras utilidades como snappyHexMesh para implementar estas figuras en el mallado lo que conlleva a una metodología computacional más compleja y más difícil de aprender.

7.1.3. Modelo de interacción partícula-pared

Para un trabajo que busque estudiar con más detalle los efectos del rebote de partículas específicamente con las paredes sólidas, ya que la interacción partícula-partícula no se consideró en todo este trabajo así como el acoplamiento bidireccional. Entonces, con un modelo de interacción partícula-pared, como por ejemplo, implementar una función de coeficiente de restitución en base a su ángulo de impacto como lo visto en la referencia [16], en vez que se especifique un solo coeficiente respecto a la componente normal y otro para la componente tangencial para todas las partículas con su respectiva velocidad que es lo que se puede hacer en OpenFOAM actualmente, sin necesidad de implementarlo personalmente (lo que requeriría conocimiento previo del lenguaje C++ y por consiguiente, un procedimiento computacional más complejo)

7.1.4. Diseño Banco de pruebas

Hacer un estudio más profundo, agregando costos económicos y revisión de un posible lugar para saber mejor cual diseño es más conveniente.



Bibliografía

- [1] Alejandro López, William Nicholls, Matthew T Stickland, and William M Dempster. Cfd study of jet impingement test erosion using ansys fluent® and openfoam®. *Computer Physics Communications*, 197:88–95, 2015. (document), 1.2, 2.4.1, 3.2.1, 4, 4.3, 4.4.5, 5.1, 5.1, 5.2, 5.1.3, 5.1.3, 5.3, 7
- [2] Vahid Javaheri, David Porter, and Veli-Tapani Kuokkala. Slurry erosion of steel—review of tests, mechanisms and materials. *Wear*, 408:248–273, 2018. (document), 1.1, 1.1
- [3] S Karthik and HJ Amarendra. Slurry jet erosion test rig: A review of erosive particles induction methods and its test parameters. *Journal of Bio-and Tribo-Corrosion*, 6(3):1–15, 2020. (document), 1.4, 1.7, 6.3
- [4] Saurabh Sangal, Mukesh Kumar Singhal, and RP Saini. Hydro-abrasive erosion in hydro turbines: a review. *International Journal of Green Energy*, 15(4):232–253, 2018. (document)
- [5] Sailesh Chitrakar, Hari Prasad Neopane, and Ole Gunnar Dahlhaug. A review on sediment erosion challenges in hydraulic turbines. *Sedimentation engineering*, page 9, 2018. (document), 1
- [6] A Kapali, S Chitrakar, O Shrestha, HP Neopane, and BS Thapa. A review on experimental study of sediment erosion in hydraulic turbines at laboratory conditions. In *Journal of Physics: Conference Series*, volume 1266, page 012016. IOP Publishing, 2019. (document)
- [7] Jone Rivrud Rygg. Cfd analysis of a pelton turbine in openfoam. Master’s thesis, Institutt for energi-og prosessteknikk, 2013. (document), 3.1.1
- [8] Alejandro Lopez, Matthew Stickland, and William Dempster. Comparative study different erosion models in an eulerian-lagrangian frame using open source software. In *12th European Fluid Machinery Congress*, 2014. (document), 1.3, 1.3, 1.3, 5.2, 5.1, 5.3, 7, 7.1.1
- [9] Marta Buszko and Alicja Krella. Slurry erosion-design of test devices. *Advances in materials science*, 17(2):5–17, 2017. 2, 1.4
- [10] GP Tilly. A two stage mechanism of ductile erosion. *Wear*, 23(1):87–96, 1973. (document), 1.1.3, 1.5
- [11] Kenneth G Budinski. *Guide to friction, wear and erosion testing*. ASTM international West Conshohocken, USA, 2007. 1
- [12] Nitish Kumar, Shivam Mishra, et al. Slurry erosion: An overview. *Materials Today: Proceedings*, 25:659–663, 2020. 1.1, 1.1
- [13] Iain Finnie. Erosion of surfaces by solid particles. *wear*, 3(2):87–103, 1960. 1.1, 1.1.2, 1.3
- [14] JGA Bitter. A study of erosion phenomena part i. *wear*, 6(1):5–21, 1963. 1.1, 1.1.1, 1.2
- [15] JGA Bitter. A study of erosion phenomena: Part ii. *Wear*, 6(3):169–190, 1963. 1.1
- [16] Rached Ben-Mansour, Hassan M Badr, Abdulrazaq A Araoye, and Ihsan Ul Haq Toor. Computational analysis of water-submerged jet erosion. *Energies*, 14(11):3074, 2021. 1.1.2, 1.4, 5.3, 5.3, 7.1.3

- [17] Eirik Volent and Ole Gunnar Dahlhaug. Literature review relevant to particle erosion in complex geometries. In *Journal of Physics: Conference Series*, volume 1042, page 012002. IOP Publishing, 2018. 1.1.3
- [18] Randall S Lynn, Kien K Wong, and Hector McI Clark. On the particle size effect in slurry erosion. *Wear*, 149(1-2):55–71, 1991. 1.1.3
- [19] Hector McI Clark. The influence of the flow field in slurry erosion. *Wear*, 152(2):223–240, 1992. 1.1.3, 1.1.4
- [20] Alan V Levy and Pauline Chik. The effects of erodent composition and shape on the erosion of steel. *Wear*, 89(2):151–162, 1983. 1.1.3
- [21] Shiva Suthan Rajahram. *Erosion-corrosion mechanisms of stainless steel UNS S31603*. PhD thesis, University of Southampton, 2010. 1.1.3
- [22] Alejandro Lopez Garcia. Computational fluid dynamics study of erosion processes. 2017. 1.1.3, II, 1.9, 2.4.1, 6.3, 6.9
- [23] Martin Sommerfeld. *Best practice guidelines for computational fluid dynamics of dispersed multiphase flows*. Ercoftac, 2008. 1.1.4, 1.6, 2.2, 2.2, 2.2, 2.3, 2.1
- [24] A Mansouri, H Arabnejad, SA Shirazi, and BS McLaury. A combined cfd/experimental methodology for erosion prediction. *Wear*, 332:1090–1097, 2015. 1.4, 1.4.1, 5.3, 5.3, 5.14a, 5.3
- [25] Amir Mansouri. *A combined CFD-experimental method for developing an erosion equation for both gas-sand and liquid-sand flows*. The University of Tulsa, 2016. 1.4, 1.4.1, 5.3
- [26] Alasdair MacKenzie, A Lopez, Konstantinos Ritos, MT Stickland, and WM Dempster. A comparison of cfd software packages' ability to model a submerged jet. In *The 11th International Conference on CFD in the Minerals and Process Industries. Dickson, Australia: Commonwealth Scientific and Industrial Research Organisation (CSIRO)*, pages 1–4, 2015. 1.4
- [27] ANKANJF Gnanavelu, N Kapur, A Neville, and JF Flores. An integrated methodology for predicting material wear rates due to erosion. *Wear*, 267(11):1935–1944, 2009. 1.4
- [28] A Gnanavelu, N Kapur, A Neville, JF Flores, and N Ghorbani. A numerical investigation of a geometry independent integrated method to predict erosion rates in slurry erosion. *Wear*, 271(5-6):712–719, 2011. 1.4
- [29] VB Nguyen, QB Nguyen, ZG Liu, S Wan, CYH Lim, and YW Zhang. A combined numerical–experimental study on the effect of surface evolution on the water–sand multiphase flow characteristics and the material erosion behavior. *Wear*, 319(1-2):96–109, 2014. 1.4, 5.3, 6.3
- [30] QB Nguyen, CYH Lim, VB Nguyen, YM Wan, B Nai, YW Zhang, and M Gupta. Slurry erosion characteristics and erosion mechanisms of stainless steel. *Tribology International*, 79:1–7, 2014. 1.4, 6.3
- [31] VB Nguyen, QB Nguyen, YW Zhang, CYH Lim, and BC Khoo. Effect of particle size on erosion characteristics. *Wear*, 348:126–137, 2016. 1.4, 6.3
- [32] HS Grewal, Anupam Agrawal, and H Singh. Design and development of high-velocity slurry erosion test rig using cfd. *Journal of materials engineering and performance*, 22(1):152–161, 2013. 1.4, I, III, 1.8a, 1.10, 6.3, 6.3, 6.10
- [33] MC Lin, LS Chang, HC Lin, CH Yang, and KM Lin. A study of high-speed slurry erosion of nicrbsi thermal-sprayed coating. *Surface and Coatings Technology*, 201(6):3193–3198, 2006. 1.8b
- [34] Y Zhang, EP Reuterfors, B Sf McLaury, SA Shirazi, and EF Rybicki. Comparison of computed and measured particle velocities and erosion in water and air flows. *Wear*, 263(1-6):330–338, 2007. 1.4.1

- [35] Alasdair Mackenzie. Development and validation of a novel hybrid cfd slurry model. 2019. 1.4.1, 6.3, 6.11
- [36] Marwan Darwish and Fadl Moukalled. *The finite volume method in computational fluid dynamics: an advanced introduction with OpenFOAM® and Matlab®*. Springer, 2021. 2.1
- [37] T Frosell, M Fripp, and E Gutmark. Investigation of slurry concentration effects on solid particle erosion rate for an impinging jet. *Wear*, 342:33–43, 2015. 2.1, 2.2
- [38] Gustavo Richmond Navarro. Modelos de turbulencia introductorio. *sf [En línea]*. Available: https://www.tec.ac.cr/sites/default/files/media/doc/modelos_de_turbulencia_intr oductorio.pdf. 2.3
- [39] Clayton T Crowe, John D Schwarzkopf, Martin Sommerfeld, and Yutaka Tsuji. *Multiphase flows with droplets and particles*. CRC press, 2011. 2.3, 2.3, 2.3
- [40] C Fernandes, D Semyonov, LL Ferrás, and J Miguel Nóbrega. Validation of the cfd-dpm solver dpmfoam in openfoam® through analytical, numerical and experimental comparisons. *Granular Matter*, 20(4):1–18, 2018. 2.2
- [41] Centro de ayuda de creo parametric. [https://support.ptc.com/help/creo/creo_pma/r6.0/spanish/index.html\\$\\$page/simulate/cfd/Particle/DiscreteParticleModel.html](https://support.ptc.com/help/creo/creo_pma/r6.0/spanish/index.html$$page/simulate/cfd/Particle/DiscreteParticleModel.html). (Accessed on 07/27/2022). 2.4.1, 2.4.1
- [42] E. Martin. Introducción a openfoam. URL: <https://1library.co/document/y4k0xx5q-introduccion-a-openfoam.html>, 2015. (Accessed on 03/24/2022). 3.1
- [43] Alejandro Lopez. Lpt for erosion modeling in openfoam, 2014. 3.3.4
- [44] Valente Hernandez-Perez, Mukhtar Abdulkadir, and BJ Azzopardi. Grid generation issues in the cfd modelling of two-phase flow in a pipe. *The Journal of Computational Multiphase Flows*, 3(1):13–26, 2011. 4.6.2, 4.3
- [45] Openfoam barycentric tracking | cfd direct. <https://cfd.direct/openfoam/free-software/barycentric-tracking/>, 2017. (Accessed on 06/15/2022). 5.1.2

A | Anexo

A.1. Códigos en OpenFOAM

A.1.1. U/0

```

dimensions      [0 1 -1 0 0 0 0];

internalField    uniform (0 0 0);

boundaryField
{
    inlet
    {
        type      fixedValue;
        value      uniform (10 0 0);
    }
    internalWalls
    {
        type      noSlip;
    }
    externalWalls
    {
        type      noSlip;
    }
    target
    {
        type      fixedValue;
        value      uniform (0 0 0);
    }
    center1
    {
        type      cyclic;
    }
    center2
    {
        type      cyclic;
    }
    topOutlet
    {
        type      inletOutlet;
    }
}

```

```

        phi
        inletValue
        value
    }
    bottomOutlet
    {
        type
    }
}

```

A.1.2. p/0

```

dimensions      [0 2 -2 0 0 0 0];

internalField    uniform 0;

boundaryField
{
    inlet
    {
        type
        value
    }
    internalWalls
    {
        type
    }
    externalWalls
    {
        type
    }
    target
    {
        type
        value
    }
    center1
    {
        type
    }
    center2
    {
        type
    }
    topOutlet
    {
        type
        value
    }
    bottomOutlet
    {

```

```

        type      fixedFluxPressure;
        value      $internalField;
    }
}

```

A.1.3. epsilon/0

```

dimensions      [0 2 -3 0 0 0 0];
internalField    uniform 39.45;
boundaryField
{
    inlet
    {
        type      fixedValue;
        value      uniform 39.45;
    }
    internalWalls
    {
        type      epsilonWallFunction;
        value      uniform 39.45;
    }
    externalWalls
    {
        type      epsilonWallFunction;
        value      uniform 39.45;
    }
    target
    {
        type      epsilonWallFunction;
        value      uniform 39.45;
    }
    center1
    {
        type      cyclic;
    }
    center2
    {
        type      cyclic;
    }
    bottomOutlet
    {
        type      zeroGradient;
    }
    topOutlet
    {
        type      zeroGradient;
    }
}

```

A.1.4. k/0

```

dimensions      [0 2 -2 0 0 0 0];

internalField    uniform 0.24;

boundaryField
{
    inlet
    {
        type      fixedValue;
        value      uniform 0.24;
    }
    internalWalls
    {
        type      kqRWallFunction;
        value      uniform 0.24;
    }
    externalWalls
    {
        type      kqRWallFunction;
        value      uniform 0.24;
    }
    target
    {
        type      kqRWallFunction;
        value      uniform 0.24;
    }
    center1
    {
        type      cyclic;
    }
    center2
    {
        type      cyclic;
    }
    bottomOutlet
    {
        type      zeroGradient;
    }
    topOutlet
    {
        type      zeroGradient;
    }
}

```

A.1.5. nut/0

```

dimensions      [0 2 -1 0 0 0 0];

internalField    uniform 0;

```

```

boundaryField
{
    inlet
    {
        type            calculated;
        value            uniform 0;
    }
    internalWalls
    {
        type            nutkWallFunction;
        value            uniform 0;
    }
    externalWalls
    {
        type            nutkWallFunction;
        value            uniform 0;
    }
    target
    {
        type            nutkWallFunction;
        value            uniform 0;
    }
    center1
    {
        type            cyclic;
    }
    center2
    {
        type            cyclic;
    }
    bottomOutlet
    {
        type            calculated;
        value            uniform 0;
    }
    topOutlet
    {
        type            calculated;
        value            uniform 0;
    }
}

```

A.1.6. decomposeParDict

```

numberOfSubdomains 6;

method            scotch;

simpleCoeffs
{
    n              (1 4 1);
    delta          0.001;
}

```

```
}
```

A.2. blockMeshDict/system

```
convertToMeters 0.0001;
```

```
vertices
```

```
(
  (0 0 0) //0
  (0 -21 0) //1
  (0 -14.85 -14.85) //2
  (0 0 -21) //3
  (0 14.85 -14.85) //4
  (0 21 0) //5
  (0 -35 0) //6
  (0 -24.7487 -24.7487) //7
  (0 0 -35) //8
  (0 24.7487 -24.7487) //9
  (0 35 0) //10
  (0 -70 0) //11
  (0 0 -70) //12
  (0 70 0) //13
  (0 -125 0) //14
  (0 0 -125) //15
  (0 125 0) //16
  (0 -250 0) //17
  (0 0 -250) //18
  (0 250 0) //19

  (960 -70 0) //20
  (960 0 -70) //21
  (960 70 0) //22
  (960 -125 0) //23
  (960 0 -125) //24
  (960 125 0) //25
  (960 -250 0) //26
  (960 0 -250) //27
  (960 250 0) //28

  (1000 0 0) //29
  (1000 -21 0) //30
  (1000 -14.85 -14.85) //31
  (1000 0 -21) //32
  (1000 14.85 -14.85) //33
  (1000 21 0) //34
  (1000 -35 0) //35
  (1000 -24.7487 -24.7487) //36
  (1000 0 -35) //37
  (1000 24.7487 -24.7487) //38
  (1000 35 0) //39
  (1000 -45 0) //40
```

```

(1000 0 -45) //41
(1000 45 0) //42
(1000 -125 0) //43
(1000 0 -125) //44
(1000 125 0) //45
(1000 -250 0) //46
(1000 0 -250) //47
(1000 250 0) //48
(1050 0 0) //49
(1050 -21 0) //50
(1050 -14.85 -14.85) //51
(1050 0 -21) //52
(1050 14.85 -14.85) //53
(1050 21 0) //54
(1050 -35 0) //55
(1050 -24.7487 -24.7487) //56
(1050 0 -35) //57
(1050 24.7487 -24.7487) //58
(1050 35 0) //59
(1050 -45 0) //60
(1050 0 -45) //61
(1050 45 0) //62
(1050 -125 0) //63
(1050 0 -125) //64
(1050 125 0) //65
(1050 -250 0) //66
(1050 0 -250) //67
(1050 250 0) //68
(1100 -125 0) //69
(1100 0 -125) //70
(1100 125 0) //71
(1100 -250 0) //72
(1100 0 -250) //73
(1100 250 0) //74
(1100 0 0) //75
(1100 -21 0) //76
(1100 -14.85 -14.85) //77
(1100 0 -21) //78
(1100 14.85 -14.85) //79
(1100 21 0) //80
(1100 -125 0) //81
(1100 -88.3883 -88.3883) //82
(1100 0 -125) //83
(1100 88.3883 -88.3883) //84
(1100 125 0) //85
(1100 -250 0) //86
(1100 -176.777 -176.777) //87
(1100 0 -250) //88
(1100 176.777 -176.777) //89
(1100 250 0) //90
(1500 0 0) //91
(1500 -21 0) //92
(1500 -14.85 -14.85) //93
(1500 0 -21) //94

```

```

(1500 14.85 -14.85) //95
(1500 21 0) //96
(1500 -125 0) //97
(1500 -88.3883 -88.3883) //98
(1500 0 -125) //99
(1500 88.3883 -88.3883) //100
(1500 125 0) //101
(1500 -250 0) //102
(1500 -176.777 -176.777) //103
(1500 0 -250) //104
(1500 176.777 -176.777) //105
(1500 250 0) //106

```

);

blocks

```

(
  hex(0 1 2 3 29 30 31 32)(6 6 125)simpleGrading(1 1 1) //0
  hex(0 3 4 5 29 32 33 34)(6 6 125)simpleGrading(1 1 1) //1
  hex(1 6 7 2 30 35 36 31)(5 6 125)simpleGrading(0.8 1 1) //2
  hex(2 7 8 3 31 36 37 32)(5 6 125)simpleGrading(0.8 1 1) //3
  hex(3 8 9 4 32 37 38 33)(5 6 125)simpleGrading(0.8 1 1) //4
  hex(4 9 10 5 33 38 39 34)(5 6 125)simpleGrading(0.8 1 1) //5
  hex(14 15 12 11 23 24 21 20)(12 18 120)simpleGrading(1 1 1) //6
  hex(15 16 13 12 24 25 22 21)(12 18 120)simpleGrading(1 1 1) //7
  hex(17 18 15 14 26 27 24 23)(12 28 120)simpleGrading(1 1 1) //8
  hex(18 19 16 15 27 28 25 24)(12 28 120)simpleGrading(1 1 1) //9
  hex(23 24 21 20 43 44 41 40)(12 18 6)simpleGrading(1 1 1) //10
  hex(24 25 22 21 44 45 42 41)(12 18 6)simpleGrading(1 1 1) //11
  hex(26 27 24 23 46 47 44 43)(12 28 6)simpleGrading(1 1 1) //12
  hex(27 28 25 24 47 48 45 44)(12 28 6)simpleGrading(1 1 1) //13
  hex(29 30 31 32 49 50 51 52)(6 6 10)simpleGrading(1 1 0.75) //14
  hex(29 32 33 34 49 52 53 54)(6 6 10)simpleGrading(1 1 0.75) //15
  hex(30 35 36 31 50 55 56 51)(5 6 10)simpleGrading(0.8 1 0.75) //16
  hex(31 36 37 32 51 56 57 52)(5 6 10)simpleGrading(0.8 1 0.75) //17
  hex(32 37 38 33 52 57 58 53)(5 6 10)simpleGrading(0.8 1 0.75) //18
  hex(33 38 39 34 53 58 59 54)(5 6 10)simpleGrading(0.8 1 0.75) //19
  hex(40 41 37 35 60 61 57 55)(12 3 10)simpleGrading(1 1 0.75) //20
  hex(41 42 39 37 61 62 59 57)(12 3 10)simpleGrading(1 1 0.75) //21
  hex(43 44 41 40 63 64 61 60)(12 18 10)simpleGrading(1 1 0.75) //22
  hex(44 45 42 41 64 65 62 61)(12 18 10)simpleGrading(1 1 0.75) //23
  hex(46 47 44 43 66 67 64 63)(12 28 10)simpleGrading(1 1 0.75) //24
  hex(47 48 45 44 67 68 65 64)(12 28 10)simpleGrading(1 1 0.75) //25
  hex(66 67 64 63 72 73 70 69)(12 28 10)simpleGrading(1 1 1.333) //26
  hex(67 68 65 64 73 74 71 70)(12 28 10)simpleGrading(1 1 1.333) //27
  hex(75 76 77 78 91 92 93 94)(6 6 45)simpleGrading(1 1 1) //28
  hex(75 78 79 80 91 94 95 96)(6 6 45)simpleGrading(1 1 1) //29
  hex(81 82 77 76 97 98 93 92)(6 24 45)simpleGrading(1 1 1) //30
  hex(82 83 78 77 98 99 94 93)(6 24 45)simpleGrading(1 1 1) //31
  hex(83 84 79 78 99 100 95 94)(6 24 45)simpleGrading(1 1 1) //32
  hex(84 85 80 79 100 101 96 95)(6 24 45)simpleGrading(1 1 1) //33
  hex(86 87 82 81 102 103 98 97)(6 28 45)simpleGrading(1 1 1) //34
  hex(87 88 83 82 103 104 99 98)(6 28 45)simpleGrading(1 1 1) //35

```

```

    hex(88 89 84 83 104 105 100 99)(6 28 45)simpleGrading(1 1 1) //36
    hex(89 90 85 84 105 106 101 100)(6 28 45)simpleGrading(1 1 1) //37
);

edges
(
    arc 6 7 (0 -32.3358 -13.3939)
    arc 7 8 (0 -13.3939 -32.3358)
    arc 8 9 (0 13.3939 -32.3358)
    arc 9 10 (0 32.3358 -13.3939)
    arc 35 36 (1000 -32.3358 -13.3939)
    arc 36 37 (1000 -13.3939 -32.3358)
    arc 37 38 (1000 13.3939 -32.3358)
    arc 38 39 (1000 32.3358 -13.3939)
    arc 11 12 (0 -49.49747 -49.49747)
    arc 12 13 (0 49.49747 -49.49747)
    arc 14 15 (0 -88.38835 -88.38835)
    arc 15 16 (0 88.38835 -88.38835)
    arc 17 18 (0 -176.777 -176.777)
    arc 18 19 (0 176.777 -176.777)
    arc 20 21 (960 -49.49747 -49.49747)
    arc 21 22 (960 49.49747 -49.49747)
    arc 23 24 (960 -88.38835 -88.38835)
    arc 24 25 (960 88.38835 -88.38835)
    arc 26 27 (960 -176.777 -176.777)
    arc 27 28 (960 176.777 -176.777)
    arc 35 37 (1000 -24.74874 -24.74874)
    arc 37 39 (1000 24.74874 -24.74874)
    arc 40 41 (1000 -31.8198 -31.8198)
    arc 41 42 (1000 31.8198 -31.8198)
    arc 43 44 (1000 -88.38835 -88.38835)
    arc 44 45 (1000 88.38835 -88.38835)
    arc 46 47 (1000 -176.777 -176.777)
    arc 47 48 (1000 176.777 -176.777)
    arc 55 56 (1050 -32.3358 -13.3939)
    arc 56 57 (1050 -13.3939 -32.3358)
    arc 57 58 (1050 13.3939 -32.3358)
    arc 58 59 (1050 32.3358 -13.3939)
    arc 55 57 (1050 -24.74874 -24.74874)
    arc 57 59 (1050 24.74874 -24.74874)
    arc 60 61 (1050 -31.8198 -31.8198)
    arc 61 62 (1050 31.8198 -31.8198)
    arc 63 64 (1050 -88.38835 -88.38835)
    arc 64 65 (1050 88.38835 -88.38835)
    arc 66 67 (1050 -176.777 -176.777)
    arc 67 68 (1050 176.777 -176.777)
    arc 69 70 (1100 -88.38835 -88.38835)
    arc 70 71 (1100 88.38835 -88.38835)
    arc 72 73 (1100 -176.777 -176.777)
    arc 73 74 (1100 176.777 -176.777)
    arc 81 82 (1100 -115.4849 -47.8354)
    arc 82 83 (1100 -47.8354 -115.4849)
    arc 83 84 (1100 47.8354 -115.4849)
    arc 84 85 (1100 115.4849 -47.8354)

```

```

arc 86 87 (1100 -230.9699 -95.6709)
arc 87 88 (1100 -95.6709 -230.9699)
arc 88 89 (1100 95.6709 -230.9699)
arc 89 90 (1100 230.9699 -95.6709)
arc 97 98 (1500 -115.4849 -47.8354)
arc 98 99 (1500 -47.8354 -115.4849)
arc 99 100 (1500 47.8354 -115.4849)
arc 100 101 (1500 115.4849 -47.8354)
arc 102 103 (1500 -230.9699 -95.6709)
arc 103 104 (1500 -95.6709 -230.9699)
arc 104 105 (1500 95.6709 -230.9699)
arc 105 106 (1500 230.9699 -95.6709)

```

```
);
```

```
boundary
```

```
(
```

```
inlet
```

```
{
```

```
type patch;
faces
```

```
(
```

```
(0 1 2 3)
```

```
(0 3 4 5)
```

```
(1 6 7 2)
```

```
(2 7 8 3)
```

```
(3 8 9 4)
```

```
(4 9 10 5)
```

```
);
```

```
}
```

```
topOutlet
```

```
{
```

```
type patch;
faces
```

```
(
```

```
(17 18 15 14)
```

```
(18 19 16 15)
```

```
(14 15 12 11)
```

```
(15 16 13 12)
```

```
);
```

```
}
```

```
bottomOutlet
```

```
{
```

```
type patch;
faces
```

```
(
```

```
(91 92 93 94)
```

```
(91 94 95 96)
```

```
(97 98 93 92)
```

```
(98 99 94 93)
```

```
(99 100 95 94)
```

```
(100 101 96 95)
```

```
(102 103 98 97)
```

```
(103 104 99 98)
```

```
(104 105 100 99)
```

```

        (105 106 101 100)
    );
}
externalWalls
{
    type wall;
    faces
    (
        (17 18 27 26)
        (18 19 28 27)
        (26 27 47 46)
        (27 28 48 47)
        (46 47 67 66)
        (47 48 68 67)
        (66 67 73 72)
        (67 68 74 73)
        (86 87 103 102)
        (87 88 104 103)
        (88 89 105 104)
        (89 90 106 105)
    );
}
internalWalls
{
    type wall;
    faces
    (
        (6 7 36 35)
        (7 8 37 36)
        (8 9 38 37)
        (9 10 39 38)
        (12 11 20 21)
        (13 12 21 22)
        (21 20 40 41)
        (22 21 41 42)
        (40 41 37 35)
        (41 42 39 37)
    );
}
target
{
    type wall;
    faces
    (
        (49 50 51 52)
        (49 52 53 54)
        (55 56 51 50)
        (56 57 52 51)
        (57 58 53 52)
        (58 59 54 53)
        (60 61 57 55)
        (61 62 59 57)
        (63 64 61 60)
        (64 65 62 61)
    );
}

```

```

        (63 64 70 69)
        (64 65 71 70)
        (75 76 77 78)
        (75 78 79 80)
        (81 82 77 76)
        (82 83 78 77)
        (83 84 79 78)
        (84 85 80 79)
    );
}
center1
{
    type cyclic;
    neighbourPatch center2;
    faces
    (
        (0 1 30 29)
        (1 6 35 30)
        (11 14 23 20)
        (14 17 26 23)
        (20 23 43 40)
        (23 26 46 43)
        (29 30 50 49)
        (30 35 55 50)
        (35 40 60 55)
        (40 43 63 60)
        (43 46 66 63)
        (63 66 72 69)
        (75 76 92 91)
        (76 81 97 92)
        (81 86 102 97)
    );
}
center2
{
    type cyclic;
    neighbourPatch center1;
    faces
    (
        (5 0 29 34)
        (10 5 34 39)
        (16 13 22 25)
        (19 16 25 28)
        (25 22 42 45)
        (28 25 45 48)
        (34 29 49 54)
        (39 34 54 59)
        (42 39 59 62)
        (45 42 62 65)
        (48 45 65 68)
        (68 65 71 74)
        (80 75 91 96)
        (85 80 96 101)
        (90 85 101 106)
    );
}

```

```

    );
}
patch12
{
    type patch;
    faces
    (
        (72 73 70 69)
        (73 74 71 70)
    );
}
patch21
{
    type patch;
    faces
    (
        (86 87 82 81)
        (87 88 83 82)
        (88 89 84 83)
        (89 90 85 84)
    );
}
patch34
{
    type patch;
    faces
    (
        (35 36 56 55)
        (36 37 57 56)
        (37 38 58 57)
        (38 39 59 58)
    );
}
patch43
{
    type patch;
    faces
    (
        (35 37 57 55)
        (37 39 59 57)
    );
}
);

mergePatchPairs
(
    (patch12 patch21)
    (patch34 patch43)
);

```

A.3. Tablas Validación

A.3.1. DPM 5 mm

Tabla A.1: Datos Validación de datos 5.1.3 para 5 mm de distancia

U (A. Lopez)	U	Error % (Vel.)	Ang. (A. Lopez)	Ang.	Error % (Ang.)
6,519148936	6,117181578	6,165948374	87,12765957	86,9787154	0,170949356
6,246808511	6,150995691	1,533788327	84,63829787	84,31987386	0,376217409
6,195744681	6,207379573	0,187788431	81,19148936	80,72410099	0,575661784
6,229787234	6,296063913	1,063867459	77,93617021	76,96595394	1,244885746
6,434042553	6,396242345	0,587503236	74,10638298	73,14084097	1,30291342
6,536170213	6,431047309	1,60832567	70,08510638	69,01620869	1,525142429
6,757446809	6,374443054	5,667876717	65,68085106	64,14945477	2,331571934
6,791489362	6,251155944	7,95603717	60,31914894	58,55184091	2,929928657

Tabla A.2: Datos re-validación de datos Fig. 5.13 para 5 mm de distancia

U (A. Lopez)	U	Error % (Vel.)	Ang. (A. Lopez)	Ang.	Error % (Ang.)
6,519148936	6,113505175	6,222342285	87,12765957	86,96475396	0,186973478
6,246808511	6,148466687	1,574273057	84,63829787	84,2751343	0,429077121
6,195744681	6,217029893	0,343545657	81,19148936	80,72401975	0,575761844
6,229787234	6,316413131	1,39051133	77,93617021	77,03102259	1,161396077
6,434042553	6,411200563	0,355017709	74,10638298	73,17265486	1,259983393
6,536170213	6,440238694	1,467702269	70,08510638	69,01618938	1,525169984
6,757446809	6,389910061	5,438988393	65,68085106	64,16378908	2,309747749
6,791489362	6,266317807	7,732789178	60,31914894	58,52568872	2,973285021

A.3.2. DPM 25 mm

Tabla A.3: Datos Validación de datos 5.1.3 para 25 mm de distancia

U (A. Lopez)	U	Error % (Vel.)	Ang. (A. Lopez)	Ang.	Error % (Ang.)
6,382978723	6,095584921	4,502502912	85,98726115	86,9854822	1,160894116
6,34893617	6,126891687	3,497349429	81,97452229	84,68630563	3,308080677
6,365957447	6,194689236	2,690376314	77,19745223	81,40416474	5,449289313
6,485106383	6,275212583	3,236551372	73,37579618	77,95583324	6,241890781
6,417021277	6,277898461	2,168027961	69,36305733	74,22078872	7,003340944
6,195744681	6,036027109	2,577859164	65,54140127	69,70208604	6,348177925
5,838297872	5,691295103	2,517904582	59,8089172	63,86871623	6,787949388
5,480851064	5,340806952	2,555152658	52,1656051	57,37130361	9,979177858

Tabla A.4: Datos re-validación de datos Fig. 5.13 para 25 mm de distancia

U (A. Lopez)	U	Error % (Vel.)	Ang. (A. Lopez)	Ang.	Error % (Ang.)
6,382978723	6,163978691	3,431000515	85,98726115	87,11942406	1,316663534
6,34893617	6,214158397	2,122840267	81,97452229	84,51477737	3,098834864
6,365957447	6,283192058	1,300124763	77,19745223	81,04925673	4,989548732
6,485106383	6,373987502	1,713447308	73,37579618	77,50170264	5,622980161
6,417021277	6,393645119	0,364283623	69,36305733	73,65786115	6,191774109
6,195744681	6,161873745	0,546680628	65,54140127	68,99492307	5,269221784
5,838297872	5,811947899	0,451329721	59,8089172	62,91415554	5,191932044
5,480851064	5,483070306	0,04049084	52,1656051	56,1743038	7,684562836

A.4. Imágenes

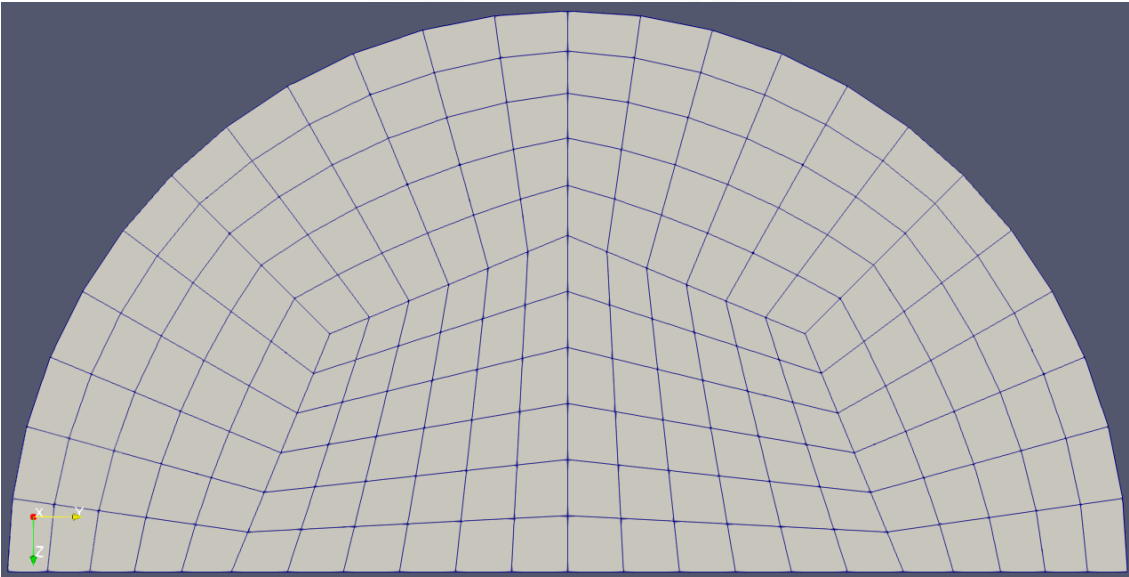
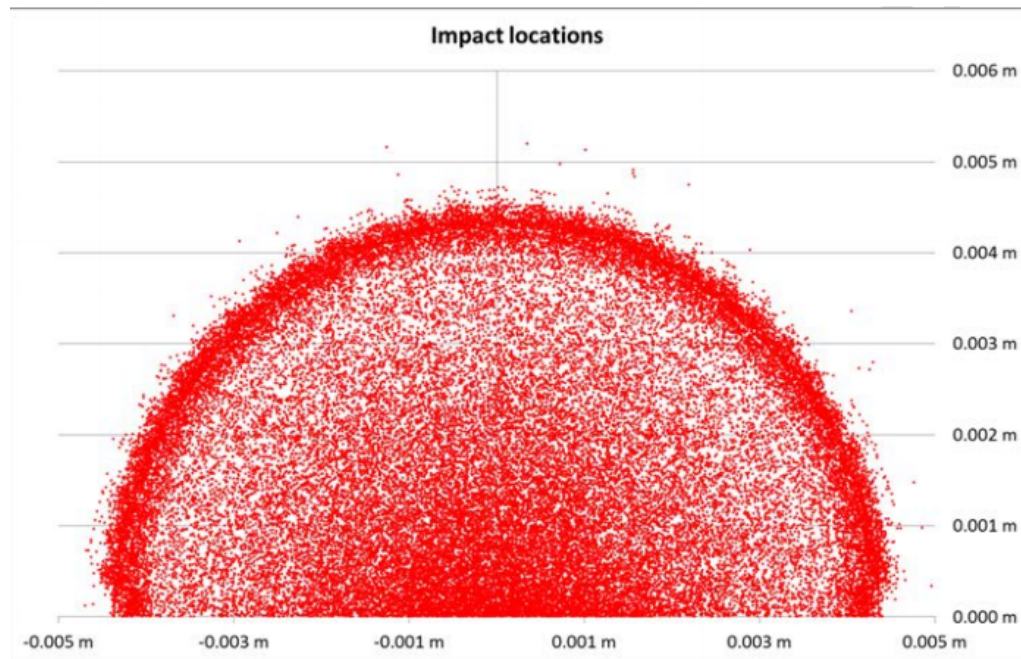
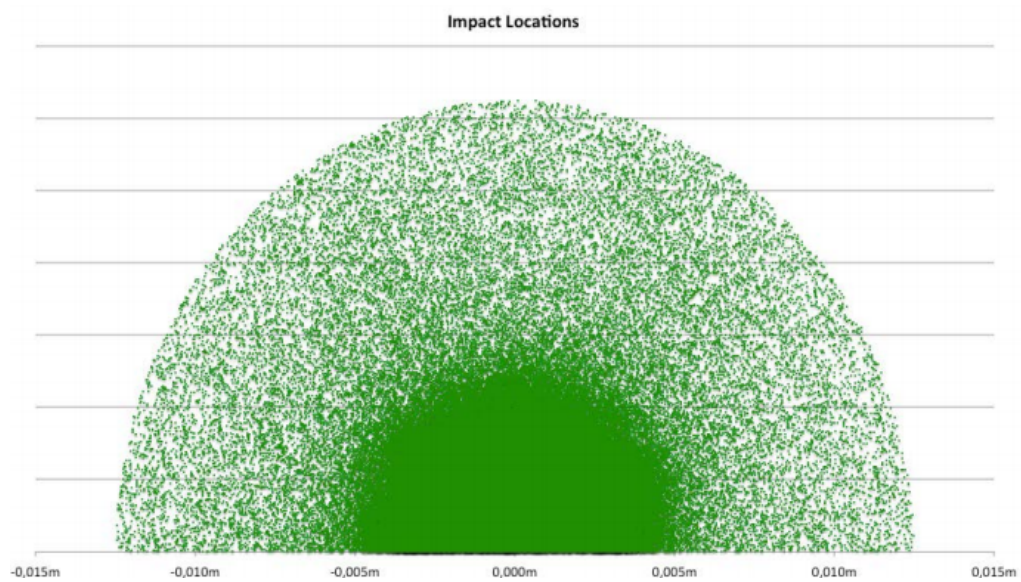


Figura A.1: Mallado centro del dominio



(a) Impactos para distancia 5 mm



(b) Impactos para distancia 25 mm

Figura A.2: Gráficos de ubicación de impactos sobre superficie de muestra obtenidos por A. Lopez en OpenFOAM

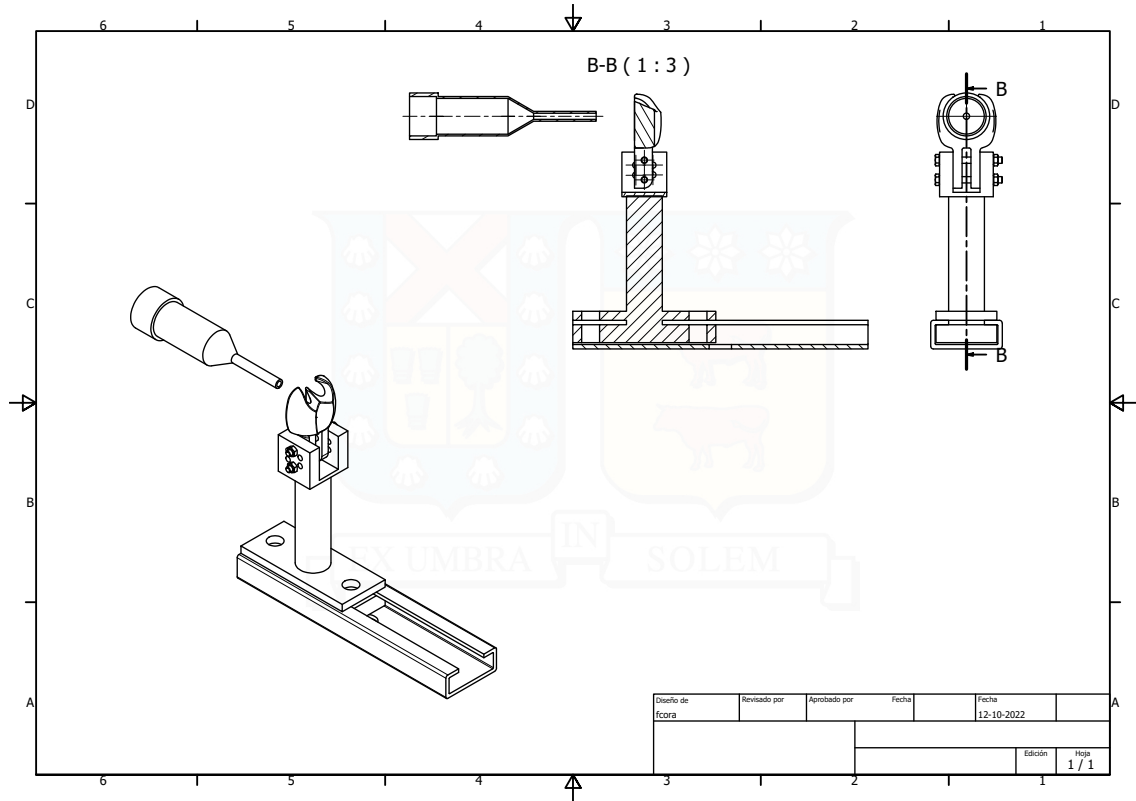


Figura A.3: Vista ISO de diseño conceptual de portador de pala Pelton
(Fuente: Elaboración propia en Autodesk Inventor)

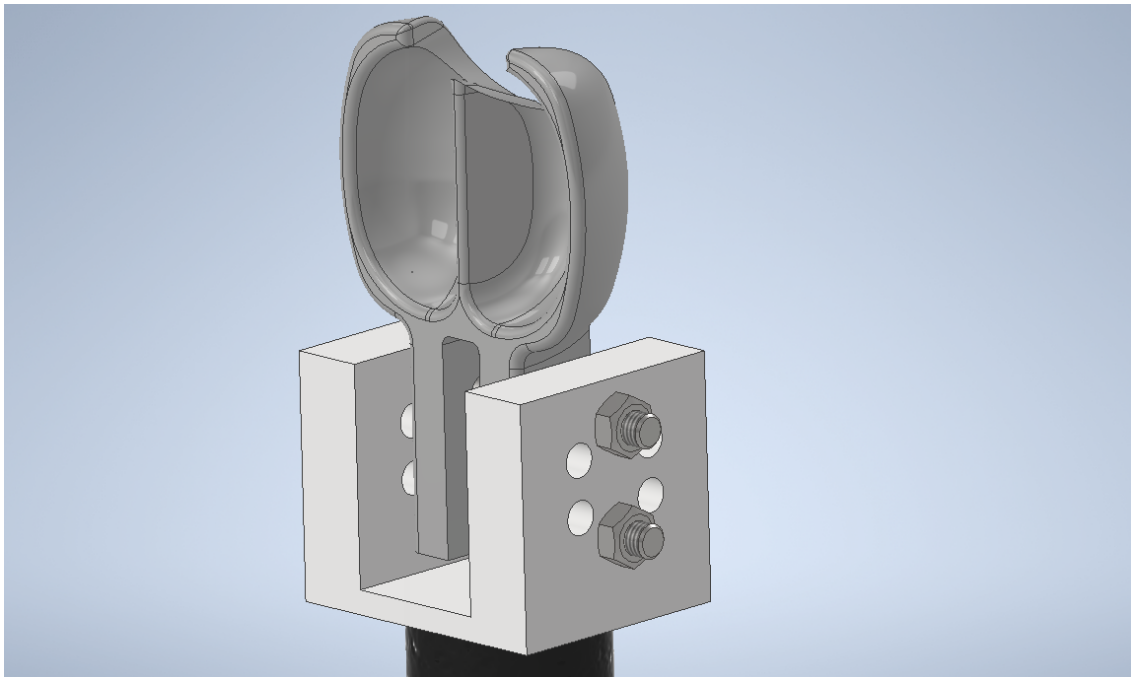


Figura A.4: Vista en 3D cabezal portador de pala Pelton
(Fuente: Elaboración propia en Autodesk Inventor)

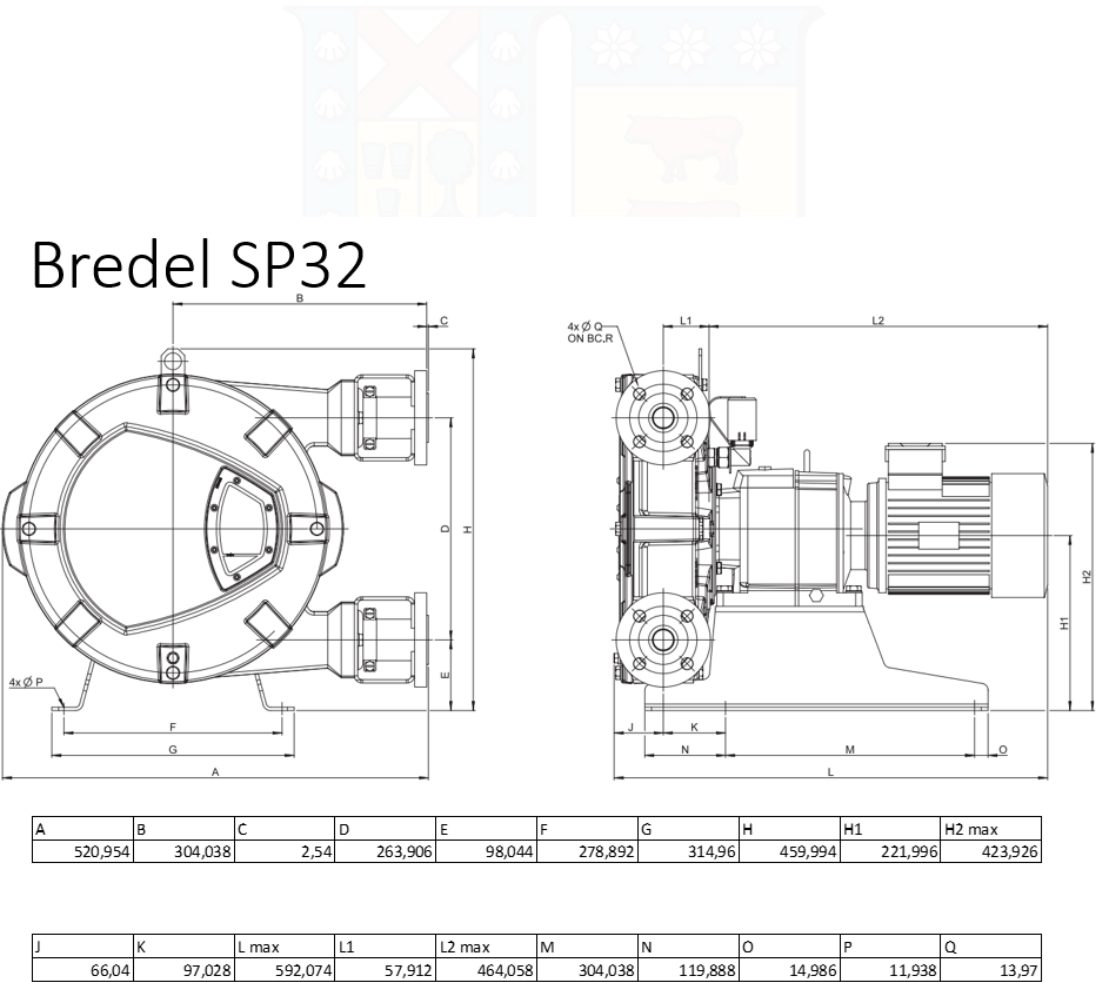


Figura A.5: Dimensiones bomba Bredel SP32
(Fuente: Catálogo Bredel, (donde se encuentran las medidas en pulgadas))

Bredel SP40

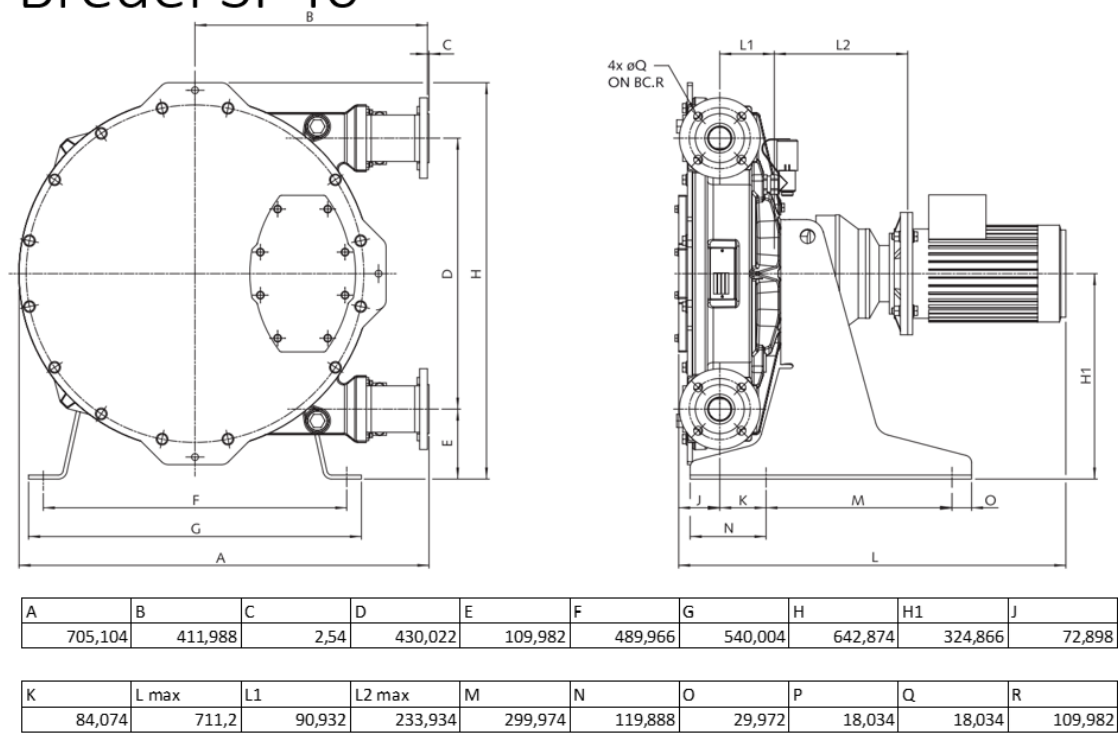


Figura A.6: Dimensiones bomba Bredel SP40
(Fuente: Catálogo Bredel, (donde se encuentran las medidas en pulgadas))