Università degli Studi di Genova

Scuola Politecnica

Corso di Laurea in Ingegneria Meccanica

Valutazione delle Performance di uno Strumento CFD Open Source per lo Sviluppo di un Energy Harvester

—

Tesi di Laurea

Relatore
Prof. A. BOTTARO

Candidato:
ALBERTO ROSSI

Correlatore:
Dr. S. OLIVIERI

Anno Accademico 2013/2014

Sommario

L'*energy harvesting* è un processo che consente di recuperare l'energia presente nell'ambiente, quale ad esempio il calore scaricato da cicli termodinamici, l'energia magnetica dovuta alla trasmissione di onde radio o i moti vibrazionali di strutture. Il compito di catturare e convertire questi potenziali in energia elettrica è affidato a dispositivi denominati *energy harvester*.

L'obiettivo di questa tesi è quello di valutare le prestazioni del codice di calcolo *OpenFOAM* per studiare il comportamento di un energy harvester che sfrutta fenomeni aeroelastici per produrre potenza elettrica.

Per studiare il problema si adotta un modello bidimensionale, dove una piastra rigida e sottile è ancorata tramite una molla ideale ed investita da un flusso uniforme incomprimibile governato dalle equazioni di Navier-Stokes.

Ai fini di una prima validazione, è stata condotta un'analisi di *grid dependency* per determinare la risoluzione minima della mesh per cui si ha una soluzione non dipendente da essa, nella situazione in cui l'ala è fissa per diversi angoli di attacco.

Successivamente si è simulato il funzionamento vero e proprio del dispositivo per alcuni casi già precedentemente studiati con il codice *Overture*. I risultati di OpenFOAM sono incoraggianti in quanto diverse particolarità del sistema dinamico sono colte, in analogia a precedenti studi.

Da questo lavoro si conclude che OpenFOAM risulta un codice idoneo a proseguire le ricerche su tale energy harvester, sia per approfondire gli aspetti teorici del problema che per simulazioni maggiormente corrispondenti agli esperimenti in galleria del vento.

Abstract

Energy harvesting is the process that allows to recover energy from the environment, such as heat released by a thermodynamic cycle, magnetic energy due to radio wave transmissions or vibrational motions. The task of the energy harvester is to capture and convert these forms of energy into electricity.

The aim of this thesis is to assess the performance of the CFD code *OpenFOAM* in the study of an experimental energy harvester which is based on aeroelastic interactions.

For this purpose, we choose a two-dimensional model, where a thin flat plate is anchored with a spring obeying to Hooke's law, and is immersed in an incompressibile and uniform flow governed by the Navier-Stokes equations.

For a first validation, we study the static configuration where the wing is fixed, for several angles of attack, performing a grid dependency analysis to estimate the lower acceptable resolution of the mesh ensuring a reliable numerical solution.

Subsequently we solve the full model, allowing the wing to move, for some cases already investigated with another numerical tool, *Overture*: OpenFOAM results are encouraging since the main features of the wing motion are captured, similarly to previous studies.

We conclude that OpenFOAM is a suitable candidate to continue the research activities within the project, both for investigating theoretical aspects and for performing simulations closer to wind tunnel experiments.

Ringraziamenti

Desidero ringraziare il Professor Alessando Bottaro per avermi dato l'opportunità di lavorare a questa tesi che mi ha dato molti strumenti sicuramente utili per il mio futuro.

Un ringraziamento lo devo sicuramente al mio correlatore l'Ingegner Stefano Olivieri che mi ha seguito passo dopo passo nella stesura di questa tesi, portando una pazienza più che infinita e dispensadomi sempre utili consigli che porterò sempre nel mio bagaglio personale.

Un grazie lo devo alla mia famiglia che è stata capace di starmi vicino nonostante i miei silenzi.

Un ringraziamento lo devo alla mia fidanzata Camilla che sempre mi ha supportato durante questi mesi, proponendosi più volte di correggere la sintassi personale della tesi ma che, per conservare la relazione, le ho impedito di fare.

Un ultimo ringraziamento lo devo fare ai miei amici conosciuti qua all'Università con i quali questi primi tre anni sono stati più leggeri grazie alle risate che ci siamo fatti.

Indice

Elenco delle tabelle

Elenco delle figure

Capitolo 1

Stato dell'arte dell'*energy harvesting*

L'*energy harvesting* (tradotto letteralmente"mietere energia") è un processo che consente di produrre energia elettrica a partire da qualunque sorgente energetica che lo possa permettere, tramite dispositivi denominati *energy harvester*, capaci di estrarre e convertire potenza da gradienti termici, moti vibrazionali o onde radio di minima entità [25].

Date le piccole energie in gioco, per massimizzare la resa degli *energy harvester* è necessario rendere questi dispositivi i più semplici e compatti possibili. Inoltre, per ottenere potenze elettriche superiori al microWatt, si sta studiando la capacità di combinare diversi dispositivi tra loro, rendendo così attraenti forme di energia finora non utilizzate per lo scarso contenuto energetico. Un'ulteriore caratteristica degli *energy harvester* è quella di non basare il principio di funzionamento su un ciclo termodinamico, dove, nei casi ideali, il rendimento è limitato dal rendimento di Carnot, che è tanto più elevato quanto è più elevata la temperatura massima del ciclo. Visto che la maggior parte delle sorgenti che si utilizzano nell'ambito dell'energy harvesting opera in condizioni ambientali, il rendimento che si avrebbe sarebbe molto scarso.

Ciò che differenzia l'energy harvesting dalla produzione elettrica tradizionale mediante fonti rinnovabili sono gli utilizzatori finali e le potenze sviluppate. L'energy harvesting, date le sorgenti che utilizza, sviluppa potenze dell'ordine del microWatt ed ha l'obiettivo di alimentare piccoli congegni elettronici. La produzione tradizionale con fonti rinnovabili invece è stata concepita per generare elevate potenze, dell'ordine del MegaWatt, per utenze domestiche e industriali. Prima di entrare nel merito del lavoro di questa tesi, si descrivono alcuni scenari in cui l'energy harvesting può essere applicato con successo.

1.1 Energia termica

Il principio utilizzato in questo caso è lo sfruttamento di un gradiente termico tra due conduttori per ricavare energia elettrica. Questo fenomeno fu osservato in primis da Seebeck nel 1821, il quale dimostrò che in un circuito composto da due metalli differenti, le cui giunzioni giacciono a temperature diverse, si crea una circolazione di corrente alla chiusura del circuito.

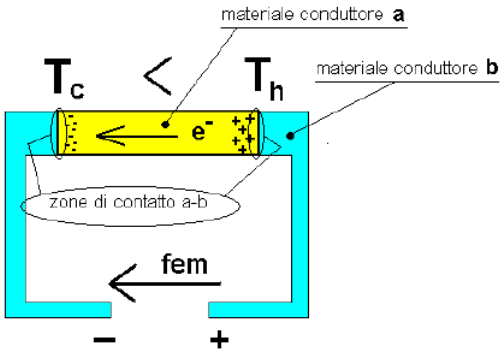


Figura 1.1: Illustrazione effetto Seebeck.

Dall'immagine 1.1 possiamo apprezzare come le cariche libere (elettroni) nel conduttore a vengano trascinate dal flusso termico dalla giunzione calda (T_h) alla giunzione fredda (T_c). Lo squilibrio di temperatura permette dunque di generare una forza elettromotrice. Questo circuito prende il nome di termocoppia. Il cosiddetto coefficiente di Seebeck $\alpha = \frac{dV}{dT}$ [V / ℃] varia per ogni materiale:

$$f.e.m. = \alpha \triangle T \ \ [V]. \tag{1.1}$$

Da questa formula si intuisce la forte influenza del coefficiente di Seebeck sulla forza elettromotrice, che dipende essenzialmente dal tipo di materiale costituente la termocoppia (Fig. 1.2). In natura, il massimo valore che è possibile trovare per α, è dell'ordine del μV/℃. Non è dunque possibile estrarre un' ingente potenza elettrica da un singolo dispositivo. Ciò quindi ci fa capire che per produrre energia in maniera considerevole è necessario usufruire di molte termocoppie collegate in serie.

La potenza termica generata è direttamente proporzionale alla conducibilità termica del materiale, la quale è pari al rapporto tra il flusso termico per unità di lunghezza e la differenza di temperatura che provoca il flusso termico. Questa è una proprietà che dipende dal materiale e che caratterizza le sue specifiche di conduttore o isolante. L'utilizzo di materiali con forti capacità di conduzione termica è altamente sconsigliato, in quanto il rendimento di conversione, pari al rapporto tra potenza elettrica prodotta

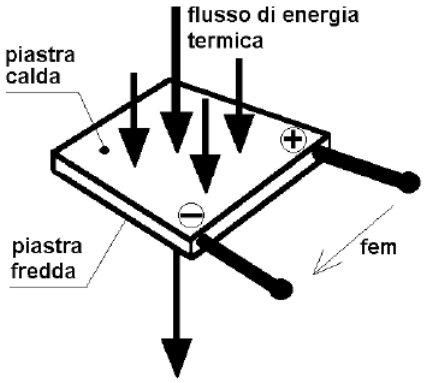


Figura 1.2: Illustrazione termocoppia.

e potenza termica fornita ($\frac{P_{el}}{Q}$), risulta penalizzato per la presenza di un elevato flusso termico. Il rendimento può essere riscritto anche come [9]:

$$\eta = \frac{T_h + T_c}{T_h} \frac{\sqrt{1+ZT}-1}{\sqrt{1+ZT}+\frac{T_h}{T_c}} Z = \frac{\alpha^2}{RkA} \tag{1.2}$$

dove

- T: è la temperatura media di funzionamento. [°K]
- α: è il coefficiente di Seebeck. [V/°C]
- R: è la resistenza elettrica del carico. [Ω]
- K: è la conducibilità termica del materiale. [W/mK]
- A: è la superficie di scambio termico. [m^2]
- Z: è un parametro adimensionale che dipende strettamente dal materiale.

Questa conclusione toglie spazio quindi all'impiego di tutti i materiali metallici, poichè possiedono alta conducibilità termica. Le ricerche svolte in questo campo hanno suggerito l'impiego di materiali semiconduttori, i quali hanno una conducibilità minore rispetto ai metalli e hanno una resa migliore al variare della temperatura media di lavoro (T_{avg}). Esistono quindi termocoppie composte da materiali diversi per ottenere migliori prestazioni per ogni range di temperatura, come il Bismuto-Tellurio (T_{avg}=90°C), Silicio-Germanio (T_{avg}=700°C) e molte altre. Lasciando momentaneamente le caratteristiche fisiche della termocoppia ritorniamo al funzionamento della termocoppia.

Si è detto che per garantire l'operatività è necessario che il conduttore principale possieda delle cariche libere, affinchè sia generata corrente elettrica circolante nello stesso verso del flusso termico. Abbiamo poi aggiunto che è sconsigliato utilizzare conduttori metallici, perciò si utilizzeranno dei semiconduttori, i quali, senza alcuna modifica, hanno caratteristiche di isolanti. Per far sì che questi conducano elettricità è necessario drogarli, cioè aggiungere atomi di elementi diversi i quali modificano la banda di conduzione del semiconduttore, che prima era satura. Si distinguono due metodi per drogare il materiale: il metodo di tipo N e quello di tipo P. Il primo tipo di drogaggio consiste nell'aggiungere elettroni, in modo tale da utilizzare la banda di conduzione successiva a quella che prima era satura, mentre il tipo P toglie elettroni dalla banda di conduzione che prima era satura, utilizzando proprio questa per condurre. A parità di flusso termico, i due drogaggi tendono quindi a creare due correnti di verso opposto. Se si crea una disposizione che mette le termocoppie in serie elettricamente e termicamente in parallelo (Fig. 1.3), i due flussi elettrici vengono a sommarsi. Questo sistema è già ampiamente disponibile sul mercato e permette di raggiungere potenze necessarie al funzionamento di piccoli dispositivi elettronici. Diamo in conclusione alcuni dati tecnici di generatori termoelettrici in produzione.

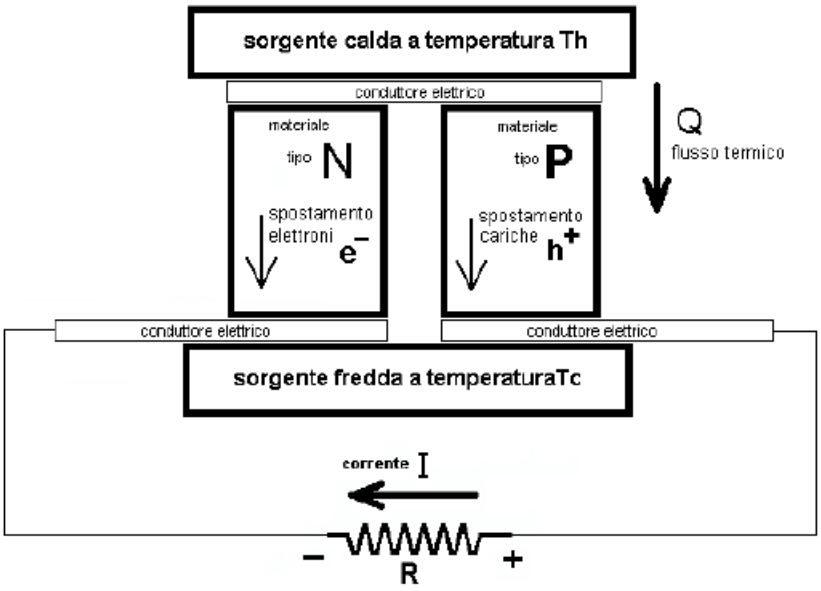


Figura 1.3: Schema termocoppia collegata in parallelo termicamente ed in serie elettricamente.

Prendiamo per esempio un modulo composto da 71 termocoppie con

drogaggio N-P collegate in serie, sottoposte a un gradiente termico di 120 °C. L'area occupata è pari a 56.38 cm^2 con spessore di 0.5 cm. Questo modulo è capace di generare una potenza elettrica 8.28 W a fronte di 324.76 W termici forniti [12]. Il rendimento quindi sarà molto basso, pari a 2.55%;, d'altro canto questo sistema non necessita praticamente di nessuna manutenzione, dato il suo funzionamento statico, e permette un recupero energetico che sarebbe andato perso, con una spesa limitata.

1.2 Energia cinetica

Questo metodo si basa sulla produzione energetica a partire dalle vibrazioni di un corpo. È forse il sistema che, preso singolarmente, produce le più basse potenze, ma ha il vantaggio che è applicabile a un numero elevatissimo di casi come mostra tabella esemplificativa 1.4, la quale è riferita a movimenti che effettuiamo ogni giorno (Fig. 1.4):

Scenario	$\overline{P}$
Taking a book off a shelf	<10 μW
Putting on reading glasses	<10 μW
Reading a book	<10 μW
Writing with a pencil	10–15 μW
Opening a drawer	10–30 μW
Spinning in a swivel chair	<10 μW
Opening a building door	<1 μW
Shaking an object	>3,000 μW

Figura 1.4: Esempi di potenza ricavabili da movimenti consueti.

Esistono diversi modi per estrarre energia da un sistema di questo tipo, ne citiamo solo due che sono i più diffusi. Il primo utilizza il principio dell'induzione elettromagnetica, il quale si serve di dispositivi a massa inerziale (Fig. 1.5) che sfruttano le oscillazioni di una massa sismica per produrre energia. Difatti, se si avvolge una bobina metallica attorno a questa massa, e viene immersa nel campo magnetico generato da un magnete permanente, con il suo movimento, per la legge di Faraday, si ottiene una variazione del flusso magnetico, con induzione di una forza elettromotrice indotta:

$$f.e.m = \frac{\triangle\phi}{\triangle t} \ [V], \tag{1.3}$$

dove $\triangle\phi$ è la variazione del flusso del campo magnetico concatenato alla bobina.

Noto il principio di funzionamento, bisogna trovare la frequenza di risonanza del sistema per massimizzare la potenza ricavata, in quanto, in questa condizione, si ottengono le massime oscillazioni del corpo e quindi maggiori

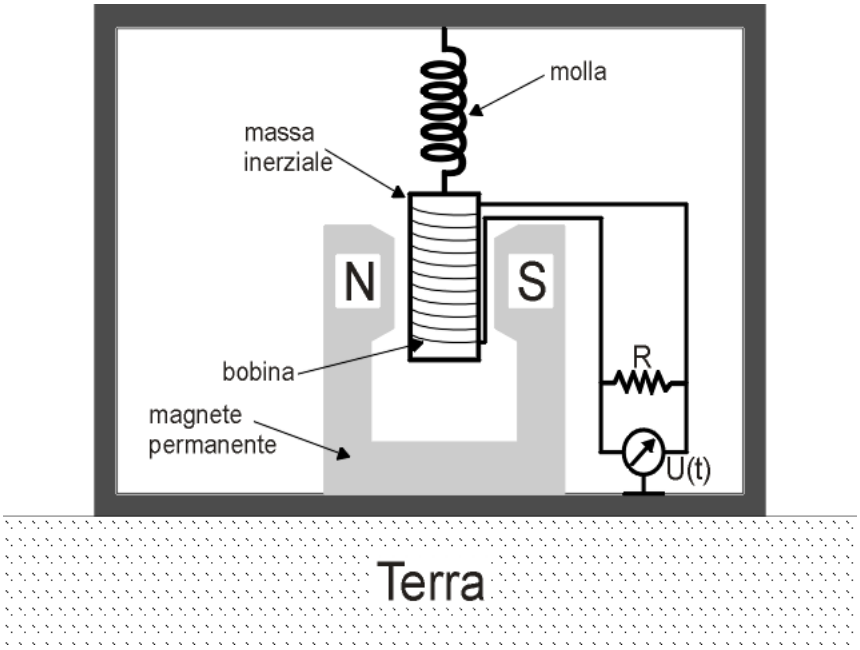


Figura 1.5: Schema di un generatore elettrico a massa inerziale.

variazioni del campo magnetico concatenato alla bobina. Questo particolare sistema permette di ricavare energia da ogni corpo che possiede un moto vibratorio, producendo elettricità con solo l'ausilio di un magnete permanente e una bobina.

Il secondo metodo che citiamo per produrre energia elettrica è il fenomeno piezoelettrico. La piezoelettricità è la capacità di un materiale di generare una tensione elettrica a seguito di una deformazione meccanica. Inoltre è valido anche il contrario, ovvero a partire da una sollecitazione dovuta a un campo elettrico si produce una deformazione meccanica sul cristallo [20]. I materiali che hanno una struttura cristallina asimmetrica, come il quarzo, presentano già l'effetto piezoelettrico, mentre materiali piezoceramici devono essere trattati artificialmente per essere polarizzati, poiché hanno una disposizione atomica simmetrica, non presentando polarità. Per curiosità in natura esistono trentadue reticoli cristallini e solo dodici di questi presentano un comportamento piezoelettrico naturale. Aiutandoci con la figura 1.6 ed adottando le stesse notazioni degli assi, cerchiamo di capire il fenomeno,

Definiamo due coefficienti che sono propri dei materiali piezoelettrici [17]:

$$d = \frac{quantità\ di\ carica\ sviluppata}{stress\ meccanico\ applicato}\ [\frac{C}{N}]; \tag{1.4}$$

$$g = \frac{campo\ elettrico\ sviluppato}{stress\ meccanico\ applicato}\ [\frac{Vm}{N}]; \tag{1.5}$$

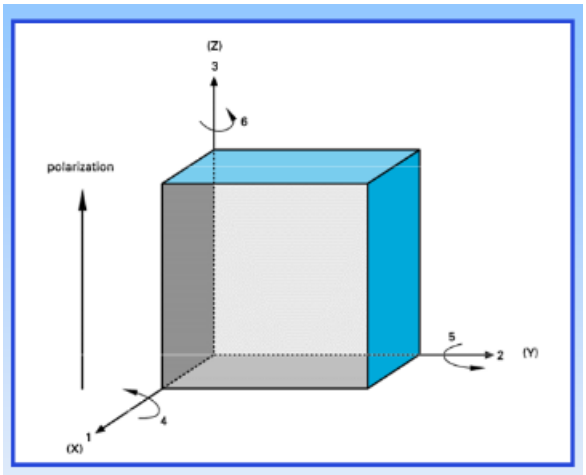


Figura 1.6: Schematizzazione di una cella di un cristallo.

Si sono introdotti tali coefficienti perché ogni corpo ha la sua direzione principale di polarizzazione. Attraverso questi parametri riusciamo ad identificare lungo quali direzioni i fenomeni piezoelettrici si manifestano in maniera più evidente. Infatti, in analogia al tensore delle tensioni, si può costruire una matrice 3x3, avente per elementi i coefficienti d_{ij} (i=1,3,j=1,3) e g_{kh} (k=1,3 h=1,3). Un altro parametro di primaria importanza è il coefficiente di accoppiamento k definito come:

$$k_{ij} = \frac{W_i^e}{W_j^m}. \tag{1.6}$$

Il coefficiente di accoppiamento esprime il rapporto tra la potenza elettrica immagazzinata secondo l'asse i-esimo e la potenza meccanica fornita secondo l'asse j-esimo. L'efficienza di conversione di un elemento piezoelettrico, sollecitato alla sua frequenza di risonanza, è dato dalla seguente formula:

$$\eta = \frac{\frac{k^2}{2(1-k^2)}}{\frac{Qk^2+1}{2Q(1+k^2)}}, \tag{1.7}$$

che evidenzia come il rendimento sia fortemente influenzato dal coefficiente k e da Q, che è il fattore di qualità del generatore [18].

1.3 Energia eolica

Un altro campo di forte interesse è quello eolico. Infatti il vento rappresenta una fonte rinnovabile inesauribile, in quanto prodotto da gradienti termici all'interno dell'atmosfera. La tecnologia più utilizzata e più diffusa è quella

della pala eolica (Fig. 1.7), in quanto copre intervalli di potenza che vanno dalle centinaia di Watt, sino ai MegaWatt. Con questi dispositivi si sfrutta l'energia del vento per far ruotare delle pale, fissate a un mozzo, le quali sono collegate a un generatore elettrico che è in grado di produrre energia elettrica. La potenza generata dipende proporzionalmente dalla velocità al cubo del vento (u), dall'area intercettata dalle pale con il flusso (A) e dalla densità dell'aria (ρ):

$$P = \frac{1}{2} A \rho u^3 \tag{1.8}$$

Come è intuibile, non è possibile assorbire completamente tutta l'energia del vento, poichè ciò presupporrebbe che a valle dell'aerogeneratore il vento abbia velocità nulla. La teoria di Betz mostra che al massimo è sfruttabile il 59.3% di energia di una massa d'aria che attraversa il dispositivo. Ogni turbina possiede una sua curva di potenza, la quale riporta la potenza generata in funzione della velocità del vento. Attraverso questo grafico (Fig. 1.8) è possibile capire quale pala possa produrre maggiore energia in sicurezza, data la velocità media del vento in una determinata zona [5].

Figura 1.7: Illustrazione di un parco eolico off shore situato in Giappone.

Questo è il metodo classico di produzione di energia elettrica sfruttando l'effetto di una corrente fluida. Garantisce la generazione di grandi quantitativi di potenza a scapito di grosse dimensioni di impianto, sia in termini di dispositivi che di spazio effettivo occupato. Un metodo meno invasivo per produrre energia dal vento, che si sta indagando nell'energy harvesting, è utilizzare dispositivi che sfruttano i principi dell'aeroelasticità.

Un esempio di energy harvester che utilizza questo principio è mostrato in figura 1.9; esso si trova nel dipartimento di Fisica dell'Università di Genova. La struttura del dispositivo è molto semplice essendo costituito da un'ala incollata a un supporto cilindrico, a cui sono collegati degli elastomeri, aventi un estremo collegato a telaio. Per estrarre energia elettrica si

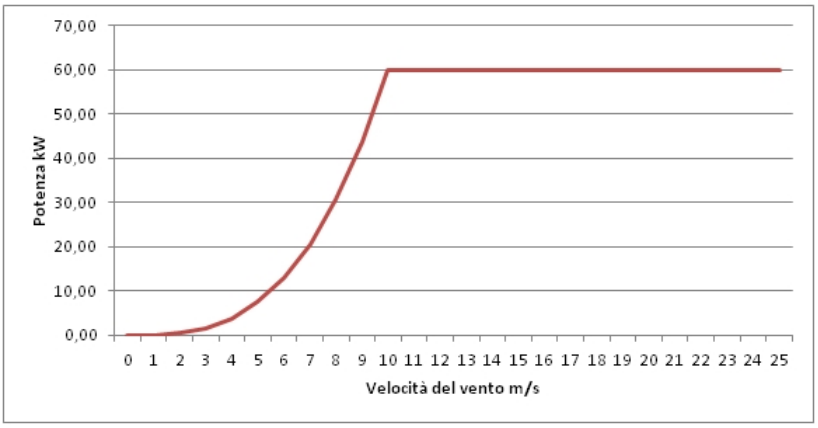


Figura 1.8: Esempio curva di potenza di una turbina eolica, con velocità di *cut in* pari a circa 2.5 m/s e *cut off* 10 m/s.

possono utilizzare due metodi. Il primo sfrutta il principio di Faraday che è applicabile se si inseriscono nel dispositivo dei magneti permanenti fissati al telaio e due bobine ferromagnetiche agli estremi del supporto cilindrico; il moto oscilattorio dell'ala, investita da una corrente fluida, produrrà una forza elettromotrice grazie alla variazione del flusso del campo magnetico che sperimentano le bobine. Il secondo metodo invece prevede l'utilizzo di elastomeri piezoelettrici, i quali essendo sottoposti a stress meccanici per le oscillazioni dell'ala, producono elettricità grazie al fenomeno piezoelettrico.

Nel prosieguo di questa tesi si forniranno ulteriori informazioni su questo energy harvester, capace di produrre una potenza elettrica grazie all'interazione con il flusso, con possibile applicazione all'alimentazione di reti WINS (Wireless Integrated Network Sensor).

1.4 Wins

Una possibile applicazione delle tecnologie derivanti dall'energy harvesting è l'alimentazione delle reti WINS (Wireless Integrated Network Sensor). Si tratta di reti di sensori *wireless* collegate a un nodo principale, denominato *sink*, che provvede a trasmettere la totalità dei dati ad un'unità centrale (Fig. 1.10) [11]. La funzione degli *energy harvester* è quella di fornire un'alimentazione permanente a tutti i sensori della rete.

Le applicazioni delle WINS sono molteplici, ad esempio:

- Monitoraggio di moti ondosi. Si può pensare di distribuire un certo numero di sensori in una zona oceanica rilevando gli spostamenti che subiscono.

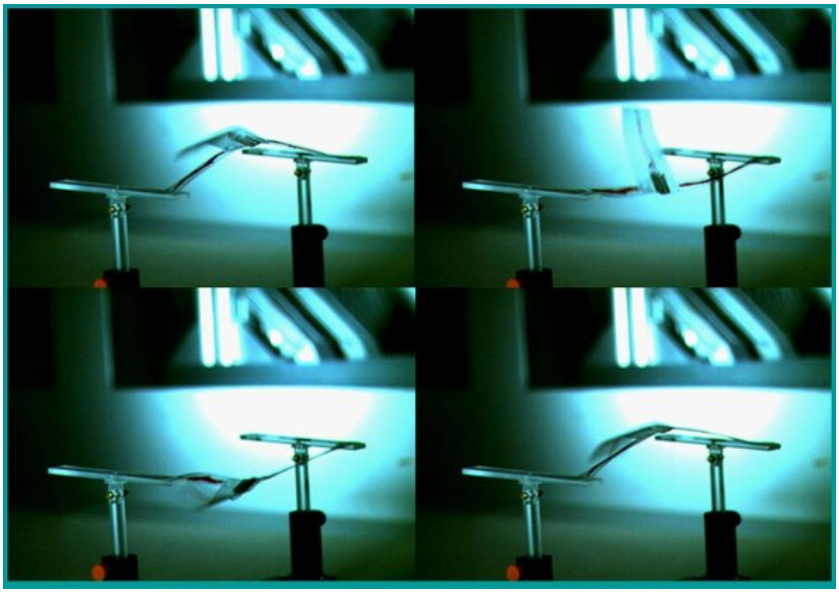


Figura 1.9: Energy harvester presente al dipartimento di Fisica dell'Università di Genova che sfrutta i principi dell'aeroelasticità.

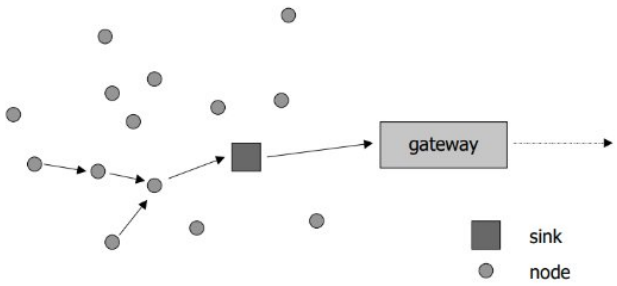


Figura 1.10: Esempio di rete WINS. I sensori trasmettono informazioni fra loro e al nodo principale.

- Rilevamento di incendi in zone boschive grazie al posizionamento di sensori dotati di termometri. I segnali vengono inviati poi all'unità principale, che valuta istantaneamente se sono presenti eventuali anomalie, accelerando le operazioni di soccorso.
- Monitoraggio della qualità dell'aria di aree urbane o dei fumi di una centrale termoelettrica.
- Controllo della psicrometria dell'aria all'interno di un edificio, rilevando i parametri fisici e modulando di conseguenza il consumo di energia elettrica dovuto alla climatizzazione.

Per rendere efficienti queste reti, bisogna cercare di rendere indipendente il sistema di alimentazione di ogni sensore. Infatti utilizzando degli accumulatori chimici (batterie), potrebbero essere interrotte le comunicazioni tra dispositivi a causa dell'esaurimento delle stesse. Utilizzando invece le tecnologie dell'energy harvesting è possibile garantire una continuità del servizio, dato che gli *energy harvester* funzionano grazie alle forme di energia presenti nell'ambiente. Le potenze che vengono assorbite dai sensori sono in linea con quelle che possono essere fornite dagli *energy harvester*. Una possibile applicazione del dispositivo analizzato in questa tesi è proprio quello di fornire alimentazione illimitata nel tempo ai sensori di una rete WINS.

Capitolo 2

Oggetto e scopo della tesi

2.1 Descrizione del dispositivo

L'oggetto di questa tesi riguarda lo studio tramite CFD di un *energy harvester* che, investito da una corrente fluida, sfrutta il principio del *fluttering* per produrre energia elettrica.

Il dispositivo è situato nel Dipartimento di Fisica dell'Università di Genova ed è al centro del progetto *FLEHAP* (*fluttering energy harvesting for autonomous power*). L'*energy harvester* è composto dall'insieme di un'ala (composta da foil e supporto cilindrico), due o quattro elastomeri fissati ad un'asta metallica attorno alla quale l'ala è libera di ruotare. Al momento si stanno studiando diverse disposizioni e configurazioni di questi componenti, al fine di individuare quella che permetta un funzionamento ottimale. I vari assetti del dispositivo si differenziano tra loro per la dimensione della corda e della larghezza dell'ala, per il posizionamento del punto di attacco e per le proprietà degli elastomeri.

Per l'estrazione di energia sono allo studio due diversi metodi [6]. Il primo metodo per ricavare energia è mostrato in figura 2.1, dove il sistema è composto dall'ala incollata al supporto cilindrico, da quattro elastomeri, che hanno un estremo collegato a un appoggio fisso e l'altro all'asta cilindrica. Sono presenti altri due appoggi fissi su cui sono montati due magneti permanenti, atti a produrre un campo magnetico. L'elettricità viene prodotta sfruttando il principio di Faraday, grazie all'inserimento di due bobine, poste ai due estremi dell'appoggio cilindrico, che muovendosi solidalmente con l'ala, sperimentano una variazione del flusso del campo magnetico, producendo una forza elettromotrice, proporzionale alla frequenza della variazione di campo magnetico indotto.

É importante dunque studiare il moto di tale dispositivo, analizzando la frequenza di oscillazione. Il secondo metodo invece utilizza elastomeri capacitivi che, con le loro contrazioni e dilatazioni durante il funzionamento del dispositivo, sono in grado di produrre elettricità sfruttando l'effetto

piezoelettrico; in quest'ultima configurazione non sono presenti i magneti permanenti e le bobine.

Per completezza si forniscono le dimensioni e le proprietà dell'*energy harvester* per un caso indicativo dai test sperimentali. L'ala ha uno spessore di circa $0.1mm$, la corda di $40mm$ e la larghezza di $60mm$. Il materiale polimerico utilizzato per l'ala è polivinil acetato ed ha una densità di 1.17 g/cm^3, portando il peso complessivo dell'ala mediamente a 0.35 g.

Normalmente l'ala è fissata, tramite incollaggio, ad un supporto cilindrico in poliacido lattico con densità pari a 1.25 g/cm^3 e lungo 75 mm, al cui interno è inserita un'asta di rame di egual lunghezza di un 1 mm di diametro. Quest'asta ha la funzione di permettere all'ala di compiere liberamente rotazioni intorno al proprio asse [19].

Al momento la potenza elettrica ricavabile dal sistema è dell'ordine del microWatt ma si ipotizza che aumentando le dimensioni aumenti anche la potenza prodotta.

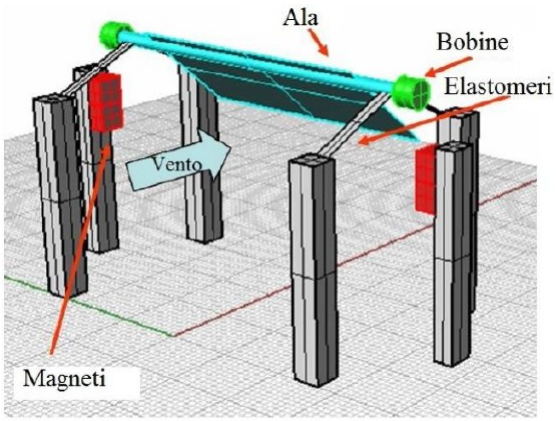


Figura 2.1: Illustrazione del dispositivo presente nel laboratorio di Fisica dell'Università di Genova.

2.2 Scopo del lavoro

Per cogliere al meglio ogni dettaglio della dinamica dell'*energy harvester*, è importante affiancare all'indagine sperimentale anche un'analisi mediante CFD (*Computational Fluid Dynamics*), con la quale è possibile eseguire simulazioni numeriche corrispondenti ai casi reali.
Il codice di calcolo che si utilizzerà è *OpenFOAM*, il quale sta suscitando un crescente interesse nell'ambito ingegneristico sia accademico che industriale, date le sue potenzialità, il suo carattere *open source* e la possibilità di personalizzazione. Lo scopo di questa tesi è quello di valutare in modo preliminare l'affidabilità di tale codice al fine di svolgere uno studio completo

sul dispositivo in oggetto. OpenFOAM è un software molto versatile scritto in C++, con cui si possono scrivere in maniera abbastanza intuitiva algoritmi di risoluzione utilizzando un' implementazione del C++ denominato *equation mimicking*.

Le simulazioni numeriche rappresentano un primo approccio al problema, poichè l'obiettivo è valutare l'affidabilità del codice di calcolo OpenFOAM. Si attueranno dunque alcune semplificazioni:

- I casi studiati sono bidimensionali, perchè dai dati sperimentali si è osservato che le oscillazioni che compie il dispositivo avvengono principalmente lungo l'asse x e z. Ciò permette, in prima approssimazione, di studiare il caso in *2D*, alleggerendo l'onere computazionale.

- Non si tengono conto degli eventuali disturbi del flusso da parte dei sostegni fissi e del supporto dell'ala.

- Si assume di modellare gli elastomeri come delle molle che seguono la legge di Hooke.

Le simulazioni svolte con OpenFOAM riprodurranno il funzionamento statico e dinamico del dispositivo. Con le prime si studieranno casi in cui l'ala rimarrà fissa a diversi angoli di attacco, semplificando lo studio del problema ai fini di una prima validazione. Le simulazioni dinamiche avranno invece lo scopo di studiare principalmente i movimenti dell'ala comparando gli esiti numerici ottenuti con i risultati di un altro software CFD.

Un problema sempre presente nel mondo computazionale è che la soluzione finale può risultare dipendente dalla costruzione dello spazio discretizzato (*mesh*), dove le equazioni di governo vengono risolte. In questa tesi si mostrerà come si conduce un'analisi di *grid dependency*, che è un'operazione che deve essere sempre svolta prima di mostrare i risultati conclusivi di una simulazione. La grid dependency consiste nel condurre simulazioni su *mesh* sempre più precise e valutare quando le soluzioni ottenute non subiscono più evidenti variazioni. Quando si trova che per un gruppo di maglie di calcolo il risultato è indipendente dalla risoluzione della mesh, si può assumere come affidabile la mesh più grezza di questo insieme. La grid dependency verrà condotta sia per i casi statici che dinamici al fine di valutare la capacità di OpenFOAM di fornire una soluzione accurata. Si svolgeranno anche delle simulazioni con un altro *meshatore*, *cfMesh*, diverso da quello di default di OpenFOAM (*blockMesh* e *snappyHexMesh*). Si è voluto condurre questa indagine perchè *cfMesh* ha la capacità di generare griglie direttamente bidimensionali e si sono valutate le differenze che sussistono con il *meshatore* di base di OpenFOAM.

La tesi sarà organizzata come segue. Nel capitolo 3 saranno forniti dei cenni sulle forze aerodinamiche e sulla teoria dell'aeroelasticità, per comprendere al meglio la fisica del problema. Nel capitolo 4 si descriveranno le

potenzialità della CFD. Nel capitolo 5.1 si descriverà il funzionamento del codice OpenFOAM. Nel capitolo 6 si mostreranno e verranno commentati i risultati ottenuti con mesh statica e l'ala fissa disposta a diversi angoli di attacco, verificando che OpenFOAM fornisca risultati accettabili in linea con la fisica del problema. Nel capitolo 7 si presenteranno i risultati con griglia dinamica e si discuterà del comportamento dell'*energy harvester* soggetto a diverse condizioni iniziali. Infine nel capitolo 8 si trarranno le conclusioni finali del lavoro illustrando gli sviluppi futuri del dispositivo.

Capitolo 3

Cenni di aerodinamica e fluttering

Prima di descrivere le simulazioni sul dispositivo, vengono forniti alcuni cenni sulle forze aerodinamiche e sulla teoria della aeroelasticità necessari per capire meglio la fisica del problema.

3.1 Forze aerodinamiche

Le forze aerodinamiche nascono dall'interazione tra il flusso e il profilo che viene investito da tale flusso, e dipendono dalla distribuzione di pressione (p) e lo sforzo tangenziale (τ) sulla superficie del corpo, entrambi dovuti alla presenza di un fluido che lambisce il profilo. Ambedue hanno come dimensioni una forza su unità d'area, la distribuzione di pressione ha come direzione di azione sempre la normale alla superficie in ogni punto, mentre lo sforzo tangenziale ha direzione tangenziale al corpo in ogni suo punto. Se si integrano queste grandezze su tutta la superficie del profilo, si ottiene una forza risultante (R), avente due componenti: portanza o *lift* (L) che è la componente perpendicolare alla velocità del flusso indisturbato, resistenza o *drag* (D) che è la componente parallela alla velocità del flusso indisturbato (Fig. 3.1) [4].

Per ali a basso angolo di attacco, si può ritenere la portanza dovuta alla differenza di pressione tra l'estradosso e l'intradosso di un profilo per unità di superficie. Adottando una geometria opportuna si può far si che tale corpo si libri oppure venga schiacciato verso il terreno. La resistenza invece è la componente della forza risultante diretta in verso opposto alla direzione del moto. É composta da tre termini: la resistenza di forma, dovuta alla geometria del corpo e alla distribuzione della pressione, la resistenza d'attrito, dovuta alla viscosità del fluido, e la resistenza indotta, dovuta alla differenza di pressione tra intradosso ed estradosso. Quest'ultima resistenza è presente solo su ali tridimensionali. L'insieme di questi tre termini porta a calcolare

la resistenza totale di un corpo.

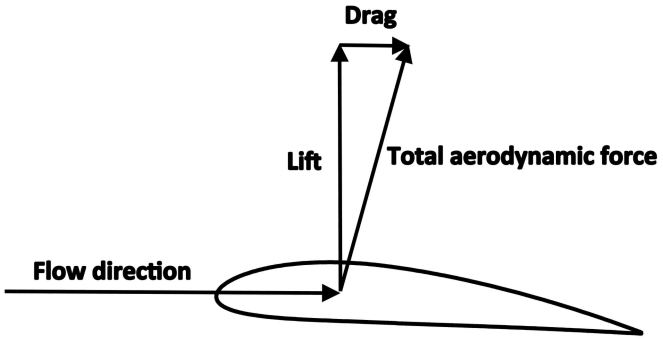


Figura 3.1: Forze aerodinamiche agenti su un profilo.

Una volta determinate le forze di resistenza e portanza si possono determinare i coefficienti adimensionali di resistenza (C_d) e portanza (C_l) definiti come:

$$C_d = \frac{D}{\frac{1}{2}\rho_f A U^2} \tag{3.1}$$

$$C_l = \frac{L}{\frac{1}{2}\rho_f A U^2} \tag{3.2}$$

dove:

- ρ_f: densità del fluido $[kg/m^3]$
- U: velocità indisturbata del fluido $[m/s]$
- A: area del profilo proiettata in direzione normale al flusso $[m^2]$

Nei problemi aerodinamici si utilizzano questi coefficienti per ridurre il numero di parametri in gioco e compattare le equazioni. Nello studio della dinamica del problema dell'energy harvester si farà riferimento a questi due coefficienti.

3.2 Strato limite

Attraverso le potenzialità di OpenFOAM sarà possibile analizzare il campo di moto e di pressione del flusso. La zona più delicata da osservare, come si è detto, è quella prossima alla superficie del corpo. Infatti questa regione è sede di fenomeni di creazione di strato limite.

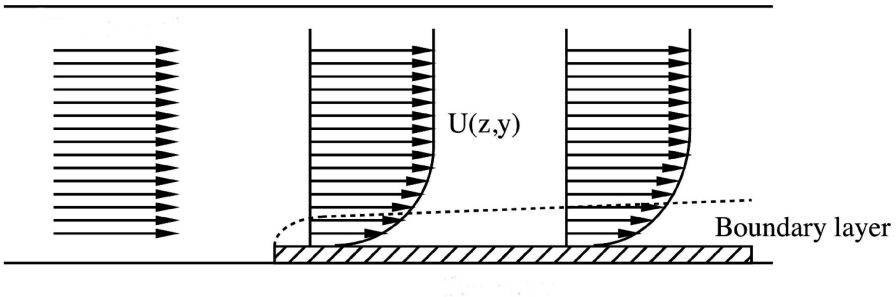


Figura 3.2: Evoluzione strato limite su una piastra.

Lo strato limite è quella porzione di spazio in cui la velocità del flusso locale si discosta dalla velocità del flusso indisturbata (Fig. 3.2). Questo è dovuto alla viscosità che possiede la corrente fluida. La viscosità è una proprietà di un fluido che permette di opporre resistenza a sforzi tangenziali esterni. Questa grandezza è la causa della formazione di forze viscose, che sono responsabili dell'aumento di resistenza al moto del profilo [7]. L'analisi precisa di questa zona risulta fondamentale ai fini della determinazione delle forze aerodinamiche. Infatti lo sforzo tangenziale resistente, per un fluido incomprimibile e Newtoniano nel caso di un fluido che lambisce una lastra piana, è pari a:

$$\tau = \mu \frac{\partial u_x}{\partial y} \tag{3.3}$$

dove:

- μ: viscosità dinamica. $[kgm^{-1}s^{-1}]$
- u_x: componente di velocità lungo l'asse x. $[m/s]$
- y: direzione lungo l'asse y. $[m]$

Si possono avere due tipi di strato limite: strato limite laminare o turbolento. Entrambi sono sede di forze dissipative, nel caso laminare comunque la direzione del flusso segue il contorno del corpo, mentre nel caso turbolento sono presenti fenomeni legati al distacco della vena fluida dalla parete.

3.3 Aeroelasticità

L'aeroelasticità è lo studio della mutua interazione tra le forze inerziali, elastiche ed aerodinamiche agenti su di un corpo esposto ad una corrente fluida [8].

Un fenomeno aerolastico molto comune e di indagine attuale in aerodinamica è il *flutter*. Questo accade quando l'ala non è sufficientemente rigida

da riuscire a opporsi alla flessione o torsione dovuta a una corrente fluida che la investe, facendo variare l'angolo di attacco della struttura con conseguente modifica delle forze aerodinamiche. Nel caso che lo smorzamento offerto dalla struttura non risulti sufficientemente elevato, il *flutter* può diventare un fenomeno distruttivo se la deflessione del corpo e la forza esercitata dal fluido diventano in fase fra loro, facendo andare in risonanza la struttura. Un esempio di *flutter* distruttivo è il caso del crollo del ponte Tacoma Narrow nel 1940. L'incidente avvenne a causa delle vibrazioni aeroelastiche autoeccitate; infatti la forza del vento influì sul moto della struttura che a sua volta modificò l'azione del flusso, causando deformazioni al ponte per torsione e flessione. Si raggiunsero condizioni tali che le oscillazioni divennero instabili invece che essere smorzate dalla struttura, portando alla distruzione del ponte.

Ci sono altri campi di interesse, come quello dell'energy harvesting, in cui il fenomeno è desiderato: l'instabilità induce in questo caso oscillazioni autosostenute e di ampiezza finita, con la conseguente possibilità di estrarre energia.

Capitolo 4

La fluidodinamica computazionale (CFD)

4.1 Introduzione alla CFD

La CFD (Computational Fluid Dynamics) è una tecnica che permette di modellizzare e simulare a calcolatore numerosi sistemi che coinvolgono un flusso di un generico fluido. I campi di applicazione sono vasti e spaziano dall'aerodinamica alla modellistica ambientale fino al campo biologico ed astronomico. La diffusione di questa tecnica è dovuta al fatto che grazie a un computer è possibile svolgere test per conoscere il comportamento di un sistema, per esempio nel caso aerodinamico volti a valutare il coefficiente di resistenza e portanza di un profilo alare, senza il bisogno di costruire modelli da testare nella galleria del vento.

È una frontiera con la quale si possono abbattere i costi dovuti ad esperimenti pratici, i quali, per taluni scenari, possono rivelarsi molto onerosi, in termini di costo e tempo, e non svolgibili in completa sicurezza. La CFD presuppone però che l'utente abbia solide basi di fluidodinamica e analisi numerica, in modo da poter interpretare al meglio i risultati e creare un modello virtuale attendibile. Il principale utilizzo della CFD è la risoluzione delle equazioni di Navier-Stokes, modellate a seconda del tipo di fluido che si sta considerando, le quali permettono di calcolare il campo di pressioni e velocità di un fluido. La risoluzione di queste equazioni per via analitica è limitata a soli pochi casi, che presuppongono il flusso laminare. Per geometrie non banali e per moti turbolenti è assai frequente adottare un approccio per via numerica, in quanto le equazioni di governo sono non lineari. È bene specificare che le equazioni di Navier-Stokes si riferiscono a un modello continuo nel tempo e nello spazio, che non può essere riprodotto attraverso l'uso di un calcolatore, perciò si ricorre alla discretizzazione del dominio, sia spaziale che temporale.

Lo spazio viene suddiviso tramite una griglia di calcolo (*mesh*), attraver-

so la quale si definisce dove le incognite devono essere calcolate.
Esistono due tipi principali di mesh che si possono utilizzare per discretizzare il dominio: strutturate e non strutturate [10]. Le mesh strutturate (fig. 4.1) hanno la caratteristica che le famiglie di linee, generate a partire dagli assi del sistema di riferimento che compongono la griglia di calcolo, si intersecano solo una volta, creando quindi solo blocchi esaedrici. Se questi assi di riferimento sono puramente rettilinei si parla di mesh strutturata cartesiana. Ogni punto è dunque facilmente numerabile ordinatamente, essendo molto simile a un piano cartesiano. Questa struttura molto regolare permette l'agevole soluzione delle equazioni per geometrie semplici, essendo che lo spazio fisico è coincidente con lo spazio computazionale. Per quest'ultima ragione i tempi necessari per la generazione della griglia sono relativamente più veloci rispetto all'uso di altre mesh [23]. Tuttavia è richiesto all'utente un maggior lavoro per la loro creazione.

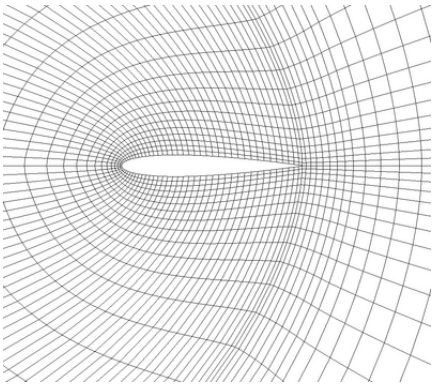


Figura 4.1: Esempio di mesh strutturata.

Le mesh strutturate cartesiane regolari non sono comunque molto utilizzate, poichè non garantiscono un'ottima densità di punti nodali dove è necessario, per esempio in prossimità del corpo. Si preferisce utilizzare piuttosto griglie strutturate body fitted.

Per geometrie complicate, le mesh di tipo strutturato non sono molto consigliate, in quanto non si riesce a controllare e a rendere uniformi le distanze tra i punti nodali della griglia. Si avranno così zone dove le incognite verranno calcolate in maniera più fitta, giungendo a una soluzione precisa, mentre in altre si avranno meno informazioni, data la minore densità di punti di calcolo, che renderà poco credibile il risultato. Si ha dunque un problema di disuniformità per geometrie complesse, il quale può essere ovviato utilizzando mesh non strutturate. La peculiarità di queste griglie risiede nel fatto che i blocchi che la formano possono avere sia forma esaedrica che tetraedrica, adattandosi meglio alla geometria del corpo. Hanno inoltre il grande vantaggio di essere generate automaticamente attraverso algoritmi. Lo svantaggio è però che si perde la possibilità di numerare ordinatamente

i nodi delle mesh, rendendo più lento lo svolgimento del sistema algebrico, in quanto questo perde la sua struttura regolare che si aveva nel caso delle mesh strutturate.

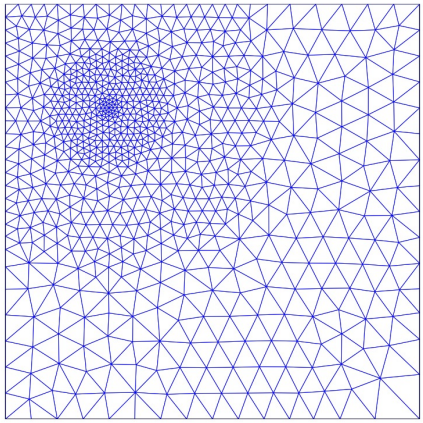


Figura 4.2: Esempio di mesh non strutturata.

Dopo che si è valutata la migliore mesh da adottare per lo studio del problema, il passo successivo sarà quello di discretizzare le equazioni, che nel nostro caso di fluido incomprimibile risultano:

$$\begin{cases} \rho \frac{D\mathbf{U}}{Dt} = \rho \mathbf{f} - \nabla p + \mu \nabla^2 \mathbf{U}, \\ \nabla \cdot \mathbf{U} = 0. \end{cases} \tag{4.1}$$

Il passaggio da continuo a discreto è svolto tramite l'operazione di *meshing* per la discretizzazione spaziale, e dall'adozione di un *time step* per la discretizzazione temporale che determina la precisione dell'andamento della soluzione (fig. 4.3).

Esistono diversi metodi di discretizzare le equazioni quali il metodo degli elementi finiti, i volumi finiti e le differenze finite. Il metodo dei volumi finiti ha come punto di partenza la conservazione della massa e della quantità di moto in forma integrale di un sistema, che viene applicato a ogni volume di controllo che suddivide l'intero dominio. Al centro di ogni cella vengono calcolati tutti i valori del problema. Il principio di questo metodo è di valutare la variazione delle grandezze del flusso tra i diversi centroidi. La linearizzazione del problema è ottenuta mediante formule quadratiche che approssimano i vari integrali di volume e superficie che definiscono la trattazione completa. Questo metodo ha il vantaggio di essere facilmente intuibile e programmabile e lo svantaggio di approssimare con difficoltà leggi di ordine superiore al secondo grado.

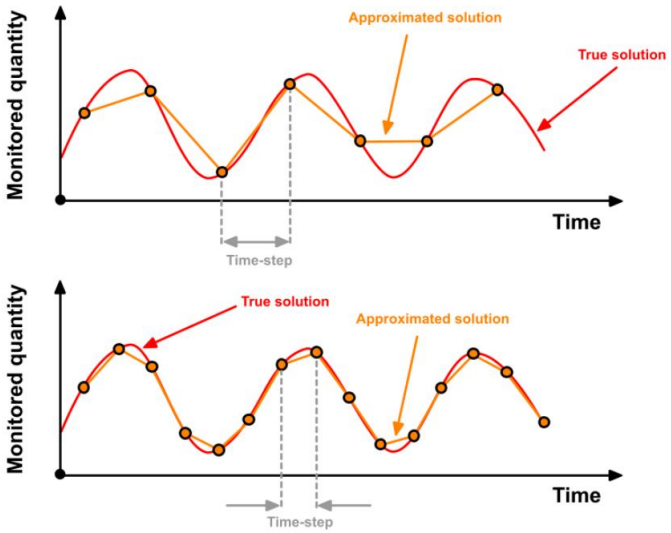


Figura 4.3: Errore di discretizzazione in funzione del time step

Il metodo delle differenze finite è il metodo più antico e più facile da usare su geometrie semplici. In questo caso il punto di partenza sono le equazioni di conservazione nella loro forma differenziale. Tale metodo prevede di associare ad ogni punto nodale della griglia un'equazione differenziale, sostituendo a ogni derivata, intesa come limite del rapporto incrementale, il rapporto incrementale stesso. Il risultato sarà avere per ogni nodo un'equazione algebrica. Il vantaggio di questo schema è che è molto semplice ed efficace su griglie strutturate non troppo complesse dove esiste una relazione ben definita tra spazio fisico e computazionale, lo svantaggio è che non garantisce soluzioni precise su mesh non strutturate o complicate.

Nel metodo degli elementi finiti si divide il dominio in elementi non strutturati, i quali garantiscono la continuità della soluzione e la conservazione delle grandezze del flusso. Ogni equazione, derivante dalla legge di conservazione del moto, associata a ogni elemento finito, è moltiplicata per una funzione peso, prima di essere linearizzata. Il vantaggio di questa tecnica è la capacità di essere applicata a qualsiasi mesh, dall'altra parte è che crea delle matrici di equazioni linearizzate molto più complesse rispetto agli altri due metodi [22].

Una volta scelto il metodo di discretizzazione, bisogna decidere come risolvere le equazioni, usando un metodo diretto o iterativo. La prima categoria è più semplice concettualmente ma applicabile a pochi casi non eccessivamente complessi, mentre la seconda richiede un minor tempo di processamento ma risulta più complicato da scrivere. Il metodo iterativo presuppone la creazione di un algoritmo, capace di trovare le incognite di un'equazione o di un sistema di equazioni a partire da un valore iniziale di tentativo, at-

traverso un *loop* di passaggi matematici che si ripete in maniera sistematica. Ad ogni ciclo si calcola un nuovo valore dell'incognita sempre più accurato; il procedimento termina a discrezione dell'utente, che impone una differenza minima che deve esistere tra le soluzioni trovate tra due cicli sucessivi.

Esistono numerosi codici di calcolo CFD che risolvono le equazioni di bilancio dei moti fludi ma tutti seguono la seguente sequenza di analisi del problema: *pre-processing*, *solving*, *post-processing*. Analizziamo adesso ogni fase.

4.1.1 Pre-processing

La fase di pre-processamento è il momento in cui si mette a punto tutto ciò che è necessario per creare la simulazione.

Il primo passo fondamentale di questo processo è la creazione del corpo che si sta studiando, mediante l'uso di un software CAD. Una geometria errata o imprecisa si ripercuote sulla soluzione finale, la quale si discosterà dal caso reale tanto più la geometria è malfatta. Gli errori da evitare sono per esempio: presenza di spigoli vivi, facce mancanti, sovrabbondanza di superfici connesse ad un'unica superficie oppure facce disallineate.

Il secondo passo è la costruzione della mesh, che può essere strutturata o non strutturata.

La decisione riguardo la griglia dipende principalmente dalla geometria che si adotta, se è poco complicata è conveniente creare una mesh strutturata, così da avere più sotto controllo la disposizione delle celle, garantendo una sufficiente accuratezza. Se invece la geometria risulta complicata è preferibile usare una mesh non strutturata, che può essere creata in modo veloce con un algoritmo, garantendo una buona uniformità di punti nodali nel dominio. Per far si che si crei una mesh che non amplifichi gli errori numerici, è bene rispettare alcuni parametri di qualità fondamentali, che ora si elencano.
La non-ortogonalità tra due celle adiacenti deve essere quanto più evitata. Questo difetto si misura in gradi e si manifesta nella rappresentazione di superfici curve usando poche celle. Nell'esempio riportato in figura 4.4 la non-ortogonalità è valutata come l'angolo tra il vettore che unisce due centroidi e il vettore ortogonale alla faccia in comune.

La *skewness* (fig. 4.5) è lo scostamento del vettore che unisce due centroidi dal centro della faccia. Nella figura f rappresenta il centro della faccia e $\triangle$ la distanza del vettore che unisce i due centroidi dal centro faccia.

L'*aspect ratio* (fig. 4.6), definito come il rapporto tra la dimensione più grande con quella più piccola della cella, deve essere quasi unitario. Questo parametro deve essere monitorato a dovere soprattutto nelle zone dove sono presenti layer prismatici, i quali sono inseriti per infittire localmente la mesh. Può accadere che, non ponderando a dovere il numero e le dimensioni dei layers, il valore dell'*aspect ratio* risulti troppo elevato, creando problemi a livello risolutivo.

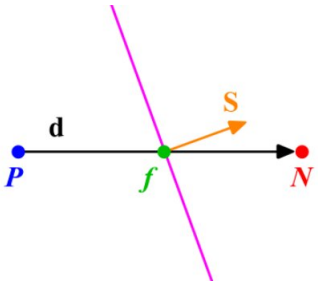


Figura 4.4: Esempio di non-ortogonalità.

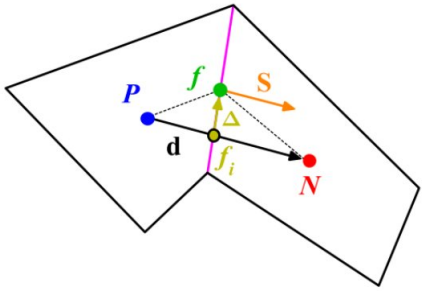


Figura 4.5: Esempio di *skewness*.

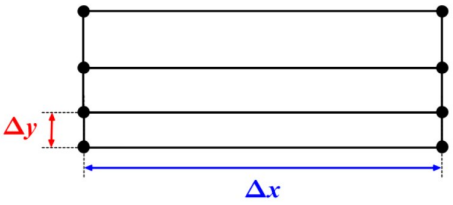


Figura 4.6: Esempio di *aspect ratio* dal valore troppo elevato.

L'ultimo parametro principale è parente dell'*aspect ratio*, ed è l'*expansion rate* (fig. 4.7), definito come la distanza tra due celle adiacenti. É buona norma adottare valori dell'*expansion rate* minori di 1.2. Anche questo parametro deve essere monitorato in presenza di layers prismatici.

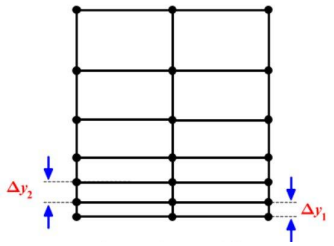


Figura 4.7: Esempio di valutazione dell'*expansion ratio.*

Tutte queste imperfezioni fanno risolvere in maniere più imprecisa le equazioni di bilancio del flusso attraverso i vari volumi di controllo [13]. Esistono poi altri parametri di qualità che possono essere monitorati in fase di creazione della mesh.

Successivamente bisogna definire tutte le proprietà termofisiche del fluido considerato, le grandezze di cui si vuole sapere il valore, specificare le condizioni iniziali e le condizioni al contorno in corrispondenza di determinate zone del dominio.

4.1.2 Solving

In questa fase si sceglie o si crea il solutore più adatto alla risoluzione del sistema di equazioni, che ha lo scopo di fornire le operazioni da svolgere al calcolatore. La struttura del *solver* dipende principalmente dal tipo di problema che si sta affrontando, per esempio se il flusso può essere considerato incomprimibile, se il fluido è bifase o se è prevalentemente in regime supersonico ed ogni tipologia di problema possiede un *solver* ottimale [2].

4.1.3 Post-processing

L'ultima fase del problema è il post-processamento, nella quale si analizzano tutti i risultati della simulazione. Si ha disponibile la valutazione numerica dei valori dei residui della soluzione, delle forze, del campo di velocità, temperature, specificati nella fase iniziale di pre-process ed anche la visualizzazione grafica che può essere utile in taluni casi. Questa fase finale di post-processamento è fondamentale dal punto di vista di verifica, infatti dai risultati ottenuti è possibile esaminare quanto i risultati trovati si discostino dalle soluzioni finali condotte con altri codici o con altri metodi [24].

Definita questa panoramica sugli aspetti fondamentali della CFD, nel prossimo capitolo si forniscono alcuni riferimenti sulla struttura del codice

di calcolo *OpenFOAM* che si utilizzerà in questa tesi e come è stata svolta la fase di pre processamento, utilizzando gli strumenti di default del codice.

Capitolo 5

OpenFOAM e implementazione del problema

5.1 Introduzione a OpenFOAM

OpenFOAM è un software CFD, open source, sviluppato dalla società OpenCFD Ltd (Gruppo ESI) e rilasciato sotto lincenza GNU GPL, che possiede pacchetti utili alla risoluzione dei problemi fluidodinamici. OpenFOAM è scritto in C++ ma possiede un *domain specific language* denominato *equation mimicking*, il quale si appoggia al C++, ma è sviluppato in modo che le entità che si vogliono rappresentare, quali funzioni od operatori matematici, sono facilmente rappresentabili attraverso una sintassi molto diretta, come possiamo apprezzare dal seguente esempio, relativo all'implementazione dell'equazione della quantità di moto [13]:

$$\frac{\partial \rho \mathbf{U}}{\partial t} + \nabla \cdot \phi \mathbf{U} - \mu \nabla^2 \mathbf{U} = -\nabla p \tag{5.1}$$

che nel codice si scrive come:

```
  fvm: :ddt(rho, U)
+ fvm: :div(phi, U)
- fvm: :laplacian(mu, U)
  ==
- fvc: :grad(p)
```

Un altro motivo della sua rapida diffusione nel mondo ingegneristico è dovuto al fatto che questo software permette l'integrazione con molti programmi CAD e meshatori, inoltre, aspetto non poco importante, è che questo codice ha un carattere *open source*. OpenFOAM possiede delle librerie che comprendono tutti gli oggetti utili a creare degli eseguibili. Le librerie contengono informazioni su vettori, tensori, condizioni al contorno, metodi di discretizzazione e modelli termofisici o viscosi. Tutte queste entità permettono di creare due tipi di applicazioni: *solvers* e *utilities*.

I solvers sono preposti alla risoluzione delle equazioni di bilancio della termofluidodinamica, mentre le *utilities* sviluppano la parte di pre e post processing. Esempi di *utilities* possono essere meshatori (*blockMesh* o *cfMesh*), oppure programmi per la visualizzazione grafica della soluzione (*paraView*). Per eseguire queste applicazioni è sufficiente lanciarle da terminale Linux.

OpenFOAM fa uso di dizionari, cioè file testuali che specificano le impostazioni delle applicazioni attraverso l'utilizzo di *keywords*, per delineare il comportamento delle utilities, come nell'esempio seguente;

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       pointVectorField;
    location    "0";
    object      pointDisplacement;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

dimensions      [0 1 0 0 0 0 0];

internalField   uniform (0 0 0);

boundaryField
{
    body
    {
        type            calculated;
        value           uniform (0 0 0);
    }

    front
    {
        type            empty;
    }

    back
    {
        type            empty;
    }

    ".*"
    {
        type            fixedValue;
        value           uniform (0 0 0);
    }
}


// ************************************************************************* //
```

Un caso OpenFOAM, per funzionare correttamente, deve avere una propria cartella (fig. 5.1), in cui sono allocate le seguenti sottocartelle. La prima è *constant*, in cui all'interno sono presenti i file inerenti alla geometria e la sottocartella *polyMesh*. Questa sottocartella contiene tutte le informazioni sulla griglia di calcolo che adesso si illustrano.

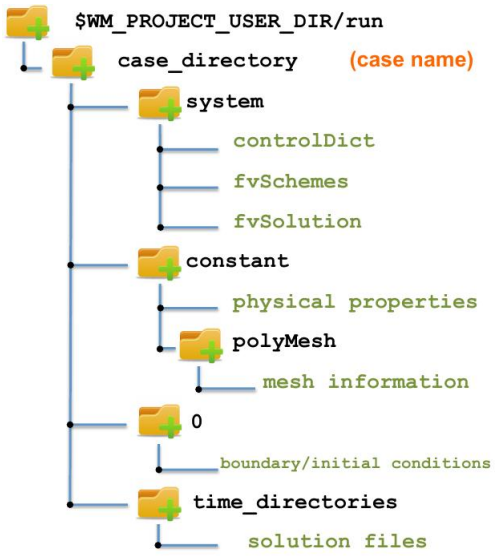


Figura 5.1: Organizzazione cartelle di un caso OpenFOAM

L'oggetto più piccolo presente nella mesh è il punto, che è definito attraverso un vettore. Tutti i punti sono inseriti in una lista, ed a ogni punto appartiene un numero. Non è ammesso che due punti siano identificati dallo stesso vettore, quindi dallo stesso numero nella lista.
Le facce sono invece un insieme ordinato di punti, dove si può definire il lato (*edge*) come il segmento che unisce due punti contigui. Il vettore normale alla cella è poi definito secondo la regola della mano destra (fig. 5.2). Le facce possono essere facce interne o facce di contorno: se connettono due celle adiacenti si dicono interne, se sono invece contigue alle regioni del confine del dominio si dicono di contorno.

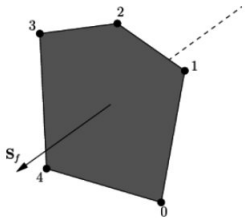


Figura 5.2: Faccia di una cella ottenuta a partire da una lista ordinata di punti

A partire dalle facce si possono definire le celle come insieme di *faces*. Le celle devono coprire l'intero dominio, essere convesse, cioè il centroide non può essere al di fuori della cella stessa e devono essere chiuse. Una volta creata la griglia con *blockMesh* o qualsiasi altro meshatore compatibile con OpenFOAM, la sotto cartella *polyMesh* si arricchirà di file testuali che descrivono tutte queste entità, che sono le grandezze basilari per generare una griglia di calcolo.

La seconda è *system*, nella quale sono presenti tre file: *controlDict*, *fvSchemes*, *fvSolution*. Nel *controlDict*, vengono specificati tutti i parametri temporali, come il tempo di inizio e fine integrazione, il $\triangle t$ di integrazione e il numero di Courant massimo. Sempre in questo file, nella sezione *function object*, sono presenti tutte le specifiche delle grandezze che si vogliono calcolare in aggiunta alle incognite principali. In *fvSchemes* viene specificato lo schema numerico che si adotta. Infine in *fvSolution* si specifica come deve essere risolto il sistema di equazioni discretizzato e le tolleranze di iterazione che si vogliono applicare.
Con questi tre dizionari si delinea quindi il modo di operare del *solver*.

L'ultima cartella è 0 dove vengono specificate le condizioni iniziali e al contorno relative alle grandezze incognite.

Si può illustrare molto semplicemente come può essere lanciato un calcolo su OpenFOAM. Da terminale ci si mette all'interno della cartella del caso e si digita *blockMesh* per creare la griglia. OpenFOAM legge i dati del file *blockMeshDict* e crea la griglia di calcolo. Successivamente, sempre da terminale, si chiama il solutore che si è scelto o creato il quale risolve le equazioni della fluidodinamica. L'ultimo passaggio è visualizzare graficamente i risultati ottenuti con l'applicazione *paraFoam* [2]. É chiaro che è possibile complicare il problema da studiare aggiungendo altre operazioni nella fase di pre-processing, in modo tale da creare una griglia più fitta nelle zone in cui serve, per esempio.

Nel paragrafo successivo si mostreranno tutti i procedimenti che sono stati utilizzati per creare il caso del dispositivo da studiare.

5.2 Pre-processamento del problema

In questa sezione si spiega come è stata condotta la fase iniziale del problema, illustrando la creazione della geometria e della mesh.

5.2.1 Geometria

La prima operazione che deve essere svolta per impostare una simulazione è la creazione della geometria che rappresenta l'ala dell'energy harvester. Nel nostro caso è stata sviluppata grazie alla piattaforma open-source *SALOME*, con cui è possibile creare strutture sia tramite interfaccia grafica (*GUI*)

che tramite script, programmati nel linguaggio *Python*, rendendo possibile l'automazione e la parametrizzazione delle procedure [1].

La geometria è esportata nel formato *.stl* (*Standard Triangulation Language*), in cui la superficie è approssimata come un insieme di triangoli.

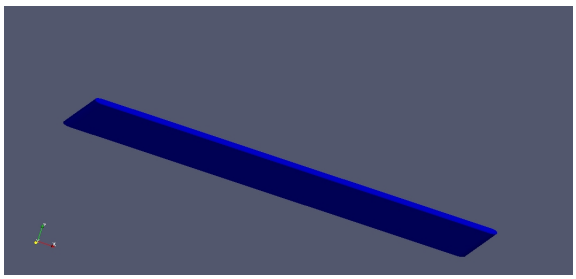


Figura 5.3: Visuale *3D* dell'ala usata per simulare i casi con i *meshatori* di default di OpenFOAM.

Fissata la lunghezza della corda del profilo è stato scelto un *aspect ratio* pari a 80, cioè con spessore pari a $0.0125c$.

5.2.2 Creazione mesh

Per creare le mesh si è scelto inizialmente di impiegare le *utility* di default di OpenFOAM che sono *blockMesh* e *snappyHexMesh*. Con il primo si costruisce la griglia di base, definendo in pratica la risoluzione massima della mesh, mentre con il secondo si eseguono raffinamenti locali per poter cogliere al meglio la soluzione in determinate zone del dominio. Un aspetto da rimarcare è che *snappyHexMesh* lavora solo con mesh *3D* sia in input che output. Per lo studio che si sta svolgendo è necessaria anche l'applicazione *extrudeMesh* che permette il passaggio dal *3D* al *2D*. Si spiega ora in maniera più accurata come lavorano queste *utilities*.

BlockMesh Mediante l'utility *blockMesh* si genera una griglia di calcolo attraverso una serie di istruzioni specificate nel dizionario *blockMeshDict*, situato nella cartella *constant/polyMesh*. Il vantaggio di questa applicazione è che ha un dizionario molto semplice da impostare ma ha il grande limite che è possibile generare soltanto mesh con una distribuzione di celle regolare. Le informazioni che sono contenute nel *blockMeshDict* sono: l'unità di misura del dominio, i limiti spaziali del dominio, il numero e le dimensioni dei blocchi che compongono la mesh e le informazioni sulle regioni sui contorni del dominio, le cosiddette *boundary*. Queste ultime sono molto importanti da definire ai fini della simulazione, poichè le caratteristiche di queste zone incidono sui calcoli computazionali. L'esecuzione delle istruzioni del *blockMeshDict*, porta alla creazione della griglia intesa come un insieme di punti, linee, facce e celle [2]. Utilizzando solo *blockMesh*, come si può vedere (Fig.

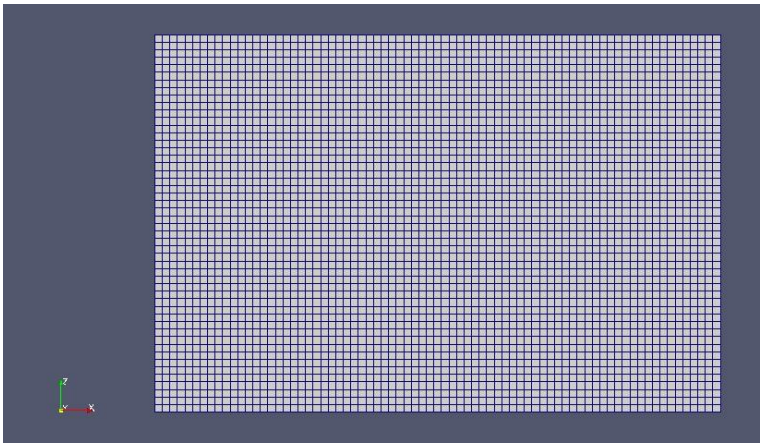


Figura 5.4: Mesh generata con blockMesh.

5.4), ci consente di fare solo una mesh di base che ha una distribuzione di punti nodali regolare. Per infittire la griglia di calcolo nelle zone utili, come possono essere quelle vicine alla geometria oppure nella regione a valle del corpo, si può agire creando diversi blocchi con risoluzioni diverse, a seconda delle esigenze.

Questa soluzione normalmente non viene adottata poichè presuppone un lavoro più lungo da parte dell'utente e non garantisce risultati sempre accettabili. Si preferisce utilizzare per il raffinamento della griglia l'utility *snappyHexMesh*.

SnappyHexMesh Questa applicazione è in grado di generare mesh *3D*, contenente celle esaedriche adattate al contorno di una geometria in formato .stl. Ciò che è richiesto è una mesh di sfondo, creabile con *blockMesh* o un altro meshatore, costituita da solamente celle esaedriche, aventi l'*aspect ratio* prossimo all'unità vicino alla superficie e il lato di ogni cella deve intersecare la geometria al massimo una sola volta.

Anche questa *utility* lavora grazie a degli input specificati nel dizionario *snappyHexMeshDict*. Tale applicazione ha disponibile tre operazioni di *meshing*, le quali possono essere scollegate una dall'altra: *castellation*, *snapping* e l'aggiunta di layers prismatici [14], ciascuna di queste tre operazioni lavora con un sotto-dizionario dove vengono specificate le entrate che definiscono le impostazioni di perfezionamento della mesh.

La fase di *castellation* fa uso del *cell splitting*, ovvero la divisione di una singola cella in più celle. Questa fase ha il compito di eseguire il raffinamento della mesh nell'intorno della geometria. La suddivisione dei volumi dipende dal livello di *refinement* che si impone (Fig. 5.5), il quale coincide con l'esponente n della relazione seguente [2]:

$$\triangle_i = \frac{\triangle_0}{2^n}, \tag{5.2}$$

dove $\triangle_0$ è la risoluzione della griglia creata con *blockMesh* e $\triangle_i$ è la risoluzione della zona di griglia dove si è eseguito il processo di raffinamento. Si dovrà specificare inoltre se questa operazione dovrà essere sviluppata all'interno, all'esterno o a una distanza prefissata dalla superficie. Inoltre verranno eliminate tutte le celle contenute per più del 50% del loro volume all'interno o all'esterno del corpo. Sono presenti altre entrate nel sotto-dizionario quali il numero massimo di celle ammesso e il numero di strati di celle tra due zone con diverso livello di raffinamento.

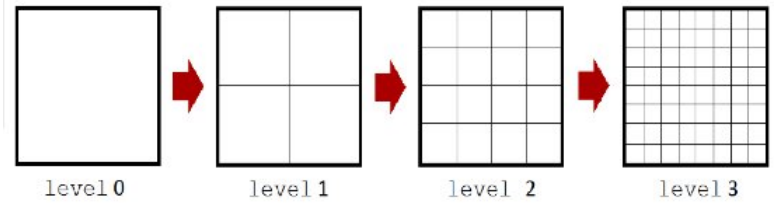


Figura 5.5: Visualizzazione del livello di raffinamento.

É possibile inoltre creare raffinamenti non solo intorno al corpo, ma anche in altre zone della mesh, specificando le coordinate di tale regione (Fig. 5.7). L'operazione di *castellation* può risultare non sufficiente a creare una buona mesh, se la geometria presenta delle curvature. Infatti la sola operazione di *splitting* non è in grado di far seguire alle celle l'andamento del profilo, rendendo necessario eseguire il secondo passo, lo *snapping*.

Questa fase ha il compito di rendere la griglia il più possibile simile al file geometrico, smussando tutte le zone frastagliate create nella fase del *castellation*. Le operazioni che si compiono in maniera iterativa in questa fase sono: lo spostamento dei vertici, creati durante la fase di *castellation*, sulla superficie del file .stl, l'analisi dei vertici che possono creare problemi dal punto di vista della qualità della mesh e la loro riduzione, fintantochè non si sono raggiunti i parametri di qualità prestabiliti. Infine se si è interessati allo studio di fenomeni di strato limite sul corpo, è possibile aggiungere strati di layers prismatici, specificandone il numero e lo spessore, che rendono uniforme e fitta la mesh proprio in prossimità della superficie. Anche questa fase è svolta iterativamente sino a che i criteri di qualità della griglia risultino soddisfatti (Fig. 5.6).

Si è osservato sperimentalmente che la dinamica dell'ala si sviluppa prevalentemente in due dimensioni e visto che questa analisi rappresenta un primo approccio al problema, per non allungare inutilmente i tempi computazionali, si è scelto di utilizzare una mesh *2D*, creabile con l'applicazione *ExtrudeMesh*. Anche questa utility lavora con un dizionario, *extrudeMesh-*

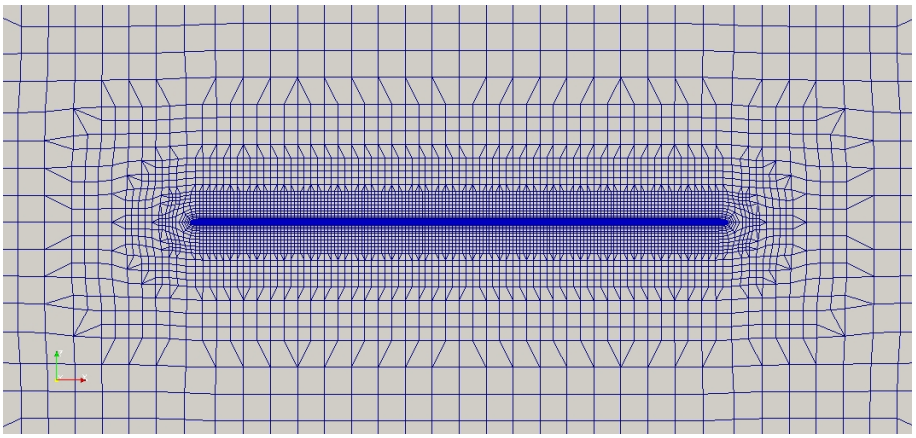


Figura 5.6: Raffinamento mesh solo su geometria utilizzando *castellation*, *snapping* e layers prismatici.

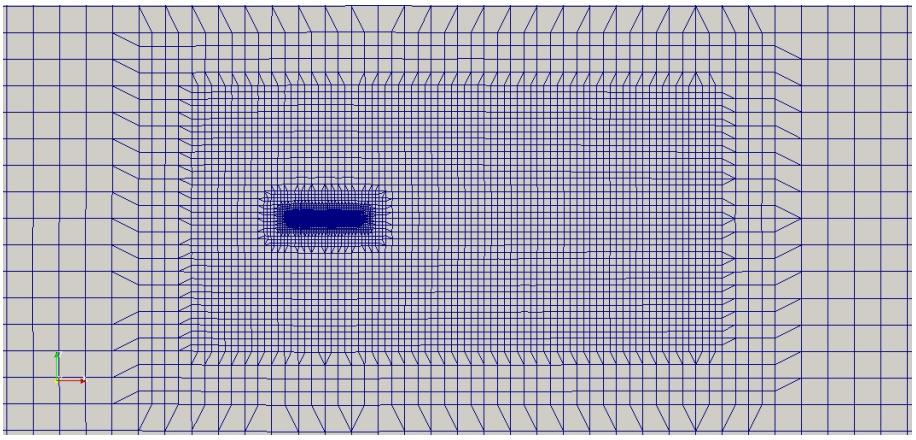


Figura 5.7: Raffinamento mesh su geometria e in un box.

Dict, dove deve essere specificato lo spessore che deve avere la nuova griglia. Nel nostro caso si è scelto spessore unitario.

5.2.3 cfMesh

In questa tesi si vuole testare un ulteriore *meshatore*, per simulare i casi statici, che ha il vantaggio di creare mesh direttamente bidimensionali. *cfMesh* è un pacchetto di librerie aggiuntivo per creare mesh, integrabile con tutte le versioni più recenti di OpenFOAM. Questo *meshatore* permette di creare griglie sia *3D* che *2D*. Ciò si rivela molto utile per il nostro caso nel quale sono necessarie solo due dimensioni.
Le mesh create precedentemente in origine erano sempre tridimensionali, poiché *snappyHexMesh* fornisce in output una mesh *3D*, solo successivamente con l'utility *extrudeMesh* era possibile ottenere il caso corrispondente *2D*.

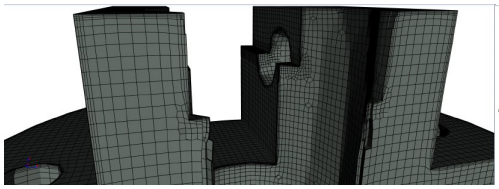
I file di input da fornire ai due *meshatori* sono totalmente differenti. Per *cfMesh* è un file di estensione *.stl* o *.fms* che rappresenta l'intero dominio mentre per *snappyHexMesh* la geometria era solo il corpo da studiare. Inoltre con *cfMesh* è necessario solo un comando da *shell* che esegue le istruzioni riportate nell'opportuno dizionario *meshDict*, invece con gli altri *meshatori* erano necessari più comandi e più dizionari per completare la griglia.
Nel nostro caso l'intera geometria è stata creata grazie alla piattaforma *SALOME*, con il quale sono state riprodotte le *boundary patch*, il tutto è stato salvato con estensione *.stl* e poi convertito nel formato *.fms*, più consono per l'utilizzo di *cfMesh*, tramite l'*utility surfaceFeatureEdges*.

Creato il file geometrico si possono usare diverse *utility* per creare la mesh. Per griglie *3D* si può utilizzare *cartesianMesh* o *tetMesh*, la prima crea griglie con celle esaedriche, mentre la seconda crea mesh con celle tetraedriche. Nel nostro caso invece si utilizza *cartesian2DMesh* che lavora con geometrie in input a forma di *ribbon* [15] (Fig. 5.8).

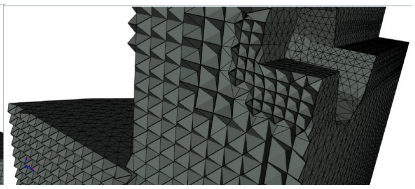
Definita la geometria e il tipo di griglia che si vuole creare, si passa alla compilazione del *meshDict*, nel quale si specificano le diverse risoluzioni che si vogliono avere all'interno del dominio. Nel nostro caso, si sono chiarite le dimensioni massime ammissibili delle celle in tutto il dominio, la dimensione delle celle lungo le *patch*, il numero di strati e le proprietà dei layer prismatici e infine tutte le caratteristiche dei *refinement box*.

Ciò che *cfMesh* e *snappyHexMesh* hanno in comune è comunque il metodo di lavoro durante il raffinamento della griglia, poichè entrambi si basano prima sul principio di *cell splitting* (Fig. 5.9 e 5.10), per aumentare il numero di celle in una determinata zona, approssimando al meglio il contorno di una geometria data e poi dell'eventuale *snapping* ed aggiunta di layers prismatici.

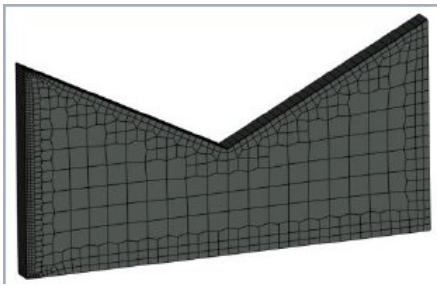
Ciò che si può dire a proposito dell'utilizzo di questo *meshatore* è che, nel nostro caso, il tempo del processamento della griglia è notevolmente minore rispetto all'uso di *snappyHexMesh*, in quanto quest'ultimo impiega un minuto per creare la mesh mentre *cfMesh* impiega 5 secondi. Questo è molto utile in



(a) Dettaglio di una mesh generata con *cartesianMesh*.



(b) Dettaglio di una mesh generata con *tetMesh*.



(c) Dettaglio di una mesh generata con *cartesian2DMesh*.

Figura 5.8: Griglie generabili con *cfMesh*.

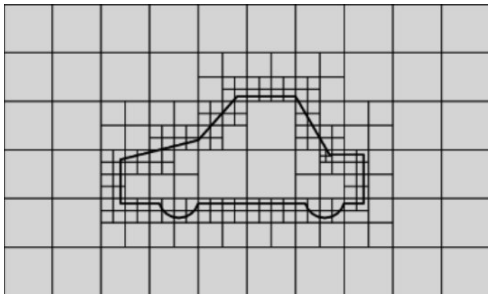
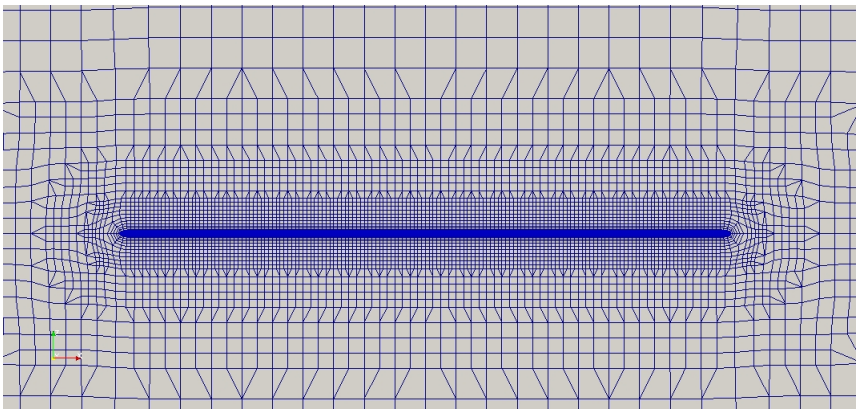
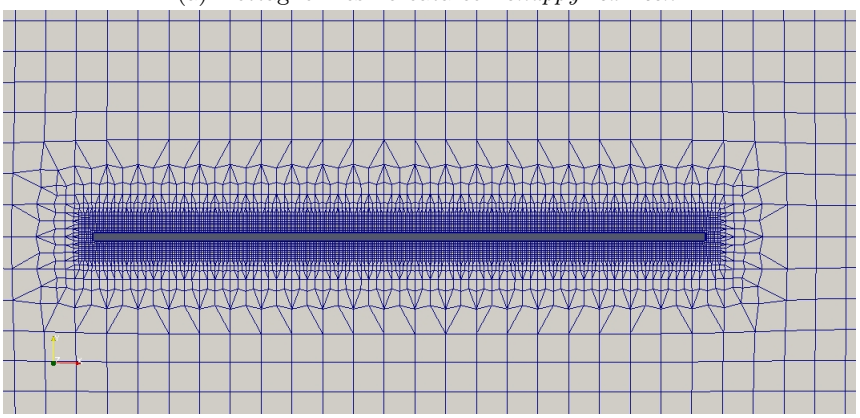


Figura 5.9: Esempio di *cell splitting* su una geometria qualsiasi.



(a) Dettaglio mesh creata con *snappyHexMesh*.



(b) Dettaglio mesh creata con *cfMesh*.

Figura 5.10: Confronto del raffinamento attorno al corpo con i due diversi *meshatori*.

fase di preparazione della griglia poichè si può vedere praticamente in modo istanteneo le modifiche che si sono apportate. Inoltre le impostazioni per regolare le caratteristiche della mesh sono facilmente controllabili attraverso l'uso di un singolo dizionario, non rendendo troppo complicato il processo di *meshing*.

Capitolo 6

Analisi di *grid dependency* per casi statici

Prima di studiare la vera e propria dinamica del dispositivo, ci si concentrerà sulla situazione in cui l'ala è mantenuta ferma, al fine di poter valutare le performance del codice in assenza di ulteriori complessità caratteristiche del problema.

Le simulazioni sono state svolte con numero di *Reynolds* (Re) pari a 1000 usando un fluido Newtoniano e si è fatta l'ipotesi di flusso incomprimibile. Il numero di Reynolds è adimensionale ed esprime il rapporto tra le forze di inerzia e le forze viscose:

$$Re = \frac{Uc}{\nu} \tag{6.1}$$

- U: la velocità del fluido. $[m/s]$
- c: lunghezza della corda del corpo. $[m]$
- ν: viscosità cinematica del fluido. $[m^2/s]$

Sotto le ipotesi introdotte nel paragrafo 2.2 sono state simulate diverse configurazioni al variare dell'angolo di attacco.

Le equazioni di governo del fluido sono state risolte con il solutore *pimpleFoam*, adatto per la soluzione di flussi incomprimibili ed instazionari.

Gli schemi numerici utilizzati per risolvere le equazioni discretizzate sono mostrati nell'Appendice A.

Si ricorda che le simulazioni sono finalizzate a valutare se OpenFOAM può essere considerato un codice di calcolo affidabile per proseguire gli studi sul dispositivo.

6.1 Metodo di indagine

Prima di mostrare i risultati trovati si spiega in cosa consiste effettivamente l'analisi di grid dependency e come la si utilizza per verificare l'affidabilità del codice di calcolo OpenFOAM.

Come si è ripetuto più volte in questa tesi, la soluzione di una simulazione CFD presenta una forte dipendenza dalla mesh creata, quindi per controllare i propri risultati è opportuno simulare lo stesso caso su griglie gradualmente sempre più accurate, al fine di cogliere la dipendenza della soluzione dalla risoluzione della mesh.

Per eseguire l'analisi di grid dependency si può scegliere di monitorare un'entità prettamente numerica quali possono essere i residui della soluzione. Il residuo r è una quantità diversa da zero che risulta sempre presente ad ogni fine iterazione a causa della non completa precisione del metodo iterativo, come spiegato nel paragrafo 4.1, ed è definito come:

$$r = Ax' - B \tag{6.2}$$

con x' soluzione dell'equazione ed A e B coefficienti.

Questo parametro è dunque indice dell'accuratezza con la quale sono state risolte le equazioni. La minimizzazione del residuo risulta comunque una condizione non sufficiente a validare la mesh presa in considerazione. L'analisi del residuo deve essere perciò accompagnata da un risultato di carattere fisico del problema. Se i residui, valutati per le diverse griglie, si assestano in un range di valori simili e i valori dei risultati finali sono plausibili, si può concludere che si è raggiunta la massima precisione di calcolo e la soluzione finale non mostra particolari segni di dipendenza dalla risoluzione della mesh, non rendendo necessario l'utilizzo di griglie più raffinate.

Un altro metodo di conduzione dell'analisi di grid dependency consiste nel monitorare l'andamento nel tempo delle grandezze fisiche caratteristiche del problema per le diverse mesh. Utilizzando grafici è possibile rendersi conto se i vari risultati ottenuti con le diverse griglie convergono ad un'unica soluzione, oppure discostano fra loro di una quantità non accettabile [16].

In questa tesi si utilizzerà l'analisi di grid dependency monitorando l'andamento delle grandezze fisiche del problema nel tempo, mentre la valutazione del residuo avrà un ruolo più marginale. Se dai risultati che si ottengono si evince che le soluzioni si dimostrano indipendenti dalla mesh utilizzata e i valori trovati risultano plausibili, OpenFOAM può essere considerato un codice idoneo a proseguire gli studi sul dispositivo.

6.2 Descrizione Mesh

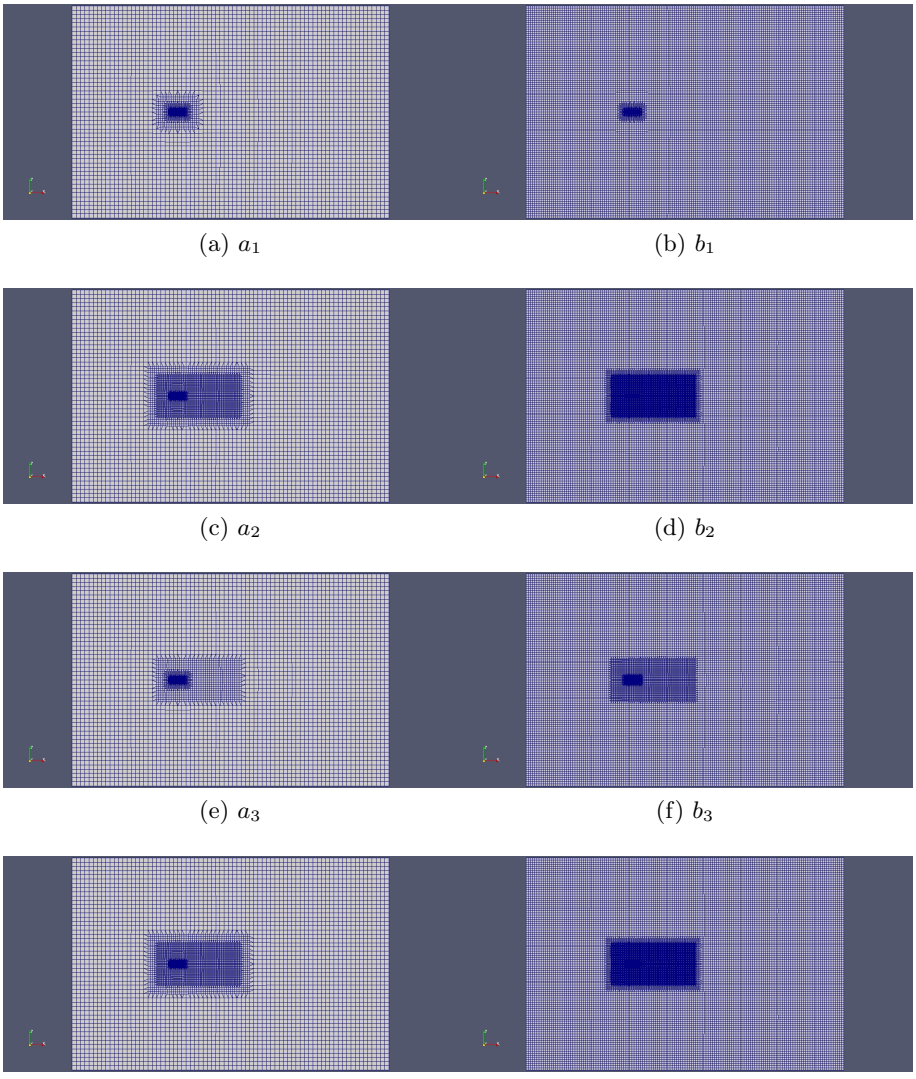
Si passa adesso alla descrizione di ogni mesh (Fig. 6.1), in cui tutte le misure fornite sono relative alla lunghezza della corda c. Le otto griglie hanno la di-

mensione del dominio fissata (-10c 0 -10c) : (20c 0.4c 10c) ma si distinguono tra loro per la risoluzione di base e l'inserimento di box di raffinamento. Le mesh denotate con la lettera a hanno risoluzione di base pari a 0.4c, mentre le griglie b pari a 0.2c. Con un pedice numerico invece si definiscono le caratteristiche dei box di raffinamento, utili a infittire localmente la mesh dove è necessario.
Il box_1 ha estensione (-2c 0 -2c) : (6c 0.1c 2c), in questa regione è inclusa la geometria e una zona abbastanza ampia nell'intorno del profilo in modo da assicurare una soluzione sufficientemente precisa. Il box_2 ha estensione (-0.7c 0 -0.25c) : (0.8c 0.1c 0.25c), ha dimensioni decisamente più ridotte del precedente in quanto ha il compito di infittire la griglia solo in prossimità dell'ala.

Le mesh con pedice 1 non possiederanno alcun box, il pedice 2 sarà attribuito alle griglie con solo il box_1, avente livello di raffinamento 2, il pedice 3 indicherà le griglie possedenti il box_1 con livello 1 e box_2 con livello 3, infine i casi 4 con box_1 aventi livello 2 e box_2 con livello 3.
Da questa prima descrizione si possono definire sinteticamente le caratteristiche di ogni mesh:

- a_1: risoluzione di base 0.4c, nessun *refinement box*.
- a_2: risoluzione di base 0.4c, *refinement* box_1 livello 2.
- a_3: risoluzione di base 0.4c, *refinement* box_1 livello 1, box_2 livello 3.
- a_4: risoluzione di base 0.4c, *refinement* box_1 livello 2, box_2 livello 3.
- b_1: risoluzione di base 0.2c, nessun *refinement box*.
- b_2: risoluzione di base 0.2c, *refinement* box_1 livello 2.
- b_3: risoluzione di base 0.2c, *refinement* box_1 livello 1, box_2 livello 3.
- b_4: risoluzione di base 0.2c, *refinement* box_1 livello 2, box_2 livello 3.

Ciò che si può osservare immediatamente dall'elenco delle mesh, è che tutti i componenti della famiglia di griglie a hanno il lato della cella massimo pari a 0.4c, mentre i casi b invece lo hanno pari a 0.2c. Questo fatto è dovuto all'utilizzo di blockMesh, con il quale si sono create griglie con un valore della risoluzione della griglia di base costante.
Il lato minimo di cella di tutte le mesh a è pari a $\frac{0.4c}{2^6} = \frac{1}{160}c$, mentre i casi b hanno lato minimo pari a $\frac{0.2c}{2^6} = \frac{1}{320}c$, cioè rispettivamente lo 0.625% e lo 0.313% della corda. Questa costanza della dimensione minima delle celle è dovuta all'utilizzo di snappyHexMesh, il quale lavora sui processi di raffinamento a partire dalle griglie di sfondo, che sono di tipo a o b. Si hanno dunque il doppio dei punti su cui è possibile calcolare la soluzione nei casi b rispetto ai casi a, sulla superficie della geometria.



(a) a_1 (b) b_1

(c) a_2 (d) b_2

(e) a_3 (f) b_3

(g) a_4 (h) b_4

Figura 6.1: Rappresentazione delle mesh utilizzate per l'analisi di *grid dependency*.

Tabella 6.1: Dimensioni e numero delle celle delle mesh utilizzate, dove $\triangle_{max}$ è la risoluzione della mesh di base, $\triangle_{min}$ è la risoluzione in prossimità della geometria, $\triangle_1$ è il lato delle celle all'interno del box_1, $\triangle_2$ è il lato delle celle all'interno del box_2.

mesh	$\triangle_{max}$	$\triangle_{min}$	$\triangle_1$	$\triangle_2$	n° celle
a_1	$40\%c$	$0.625\%c$	-	-	9274
a_2	$40\%c$	$0.625\%c$	$20\%c$	-	12178
a_3	$40\%c$	$0.625\%c$	$20\%c$	$5\%c$	9646
a_4	$40\%c$	$0.625\%c$	$10\%c$	$5\%c$	12178
b_1	$20\%c$	$0.313\%c$	-	-	25444
b_2	$20\%c$	$0.313\%c$	$10\%c$	-	37516
b_3	$20\%c$	$0.313\%c$	$10\%c$	$2.5\%c$	27946
b_4	$20\%c$	$0.313\%c$	$5\%c$	$2.5\%c$	37756

I casi denotati con il pedice 2 hanno lati delle celle del box_1 pari a $0.1c$ nei casi a, $0.05c$ nei casi b, cioè il 10% della corda e il 5% della corda. I casi con pedice 3 hanno il box_1 con celle di lato pari a $0.2c$ nei casi a e $0.1c$ nei casi b, cioè il 20% e il 10% della corda, mentre il box_2 con lato $0.05c$ per a e $0.025c$ per b, ovvero il 5% e il 2.5% della corda. Infine i casi con pedice 4 avranno il box_1 con lato pari a $0.1c$ nei casi a e $0.05c$ per i casi b, mentre il box_2 uguale al caso con pedice 3.
Nella tabella 6.1 vengono riportate in maniera sintentica tutte le dimensioni caratteristiche delle mesh utilizzate.

6.3 Configurazione a 0°

Avendo ora chiara la caratteristica di ogni mesh e le informazioni principali sulla simulazione, riportiamo una tabella che mostra i risultati dei coefficienti aerodinamici calcolati alla fine della simulazione e i grafici relativi ai valori assunti nel tempo:

Dalla Tabella 6.2 si può immediatamente notare che i risultati ottenuti appartengono tutti allo stesso ordine di grandezza. Invece dai grafici dei coefficienti aerodinamici plottati in funzione del tempo (Fig. 6.2 e Fig. 6.3), si può apprezzare che per tutti i casi si ha una soluzione che converge a un dato valore. La soluzione è dunque stabile, ma per capire più chiaramente se c'è o meno indipendenza dei risultati finali dalla risoluzione di griglia è necessaria un'analisi più approfondita. A livello informativo, il transitorio iniziale che si osserva è dovuto prettamente a fenomeni numerici che non verranno discussi in questo lavoro.

Aiutandoci con i valori finali dei coefficienti riportati in tabella 6.2, valutiamo se i risultati trovati a fine simulazione possano essere ritenuti accet-

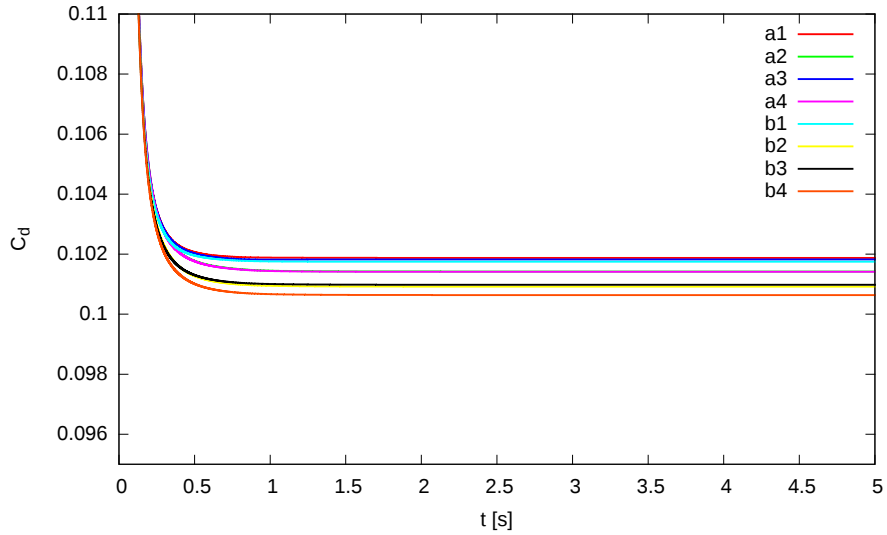


Figura 6.2: C_d in funzione del tempo per il caso a 0°.

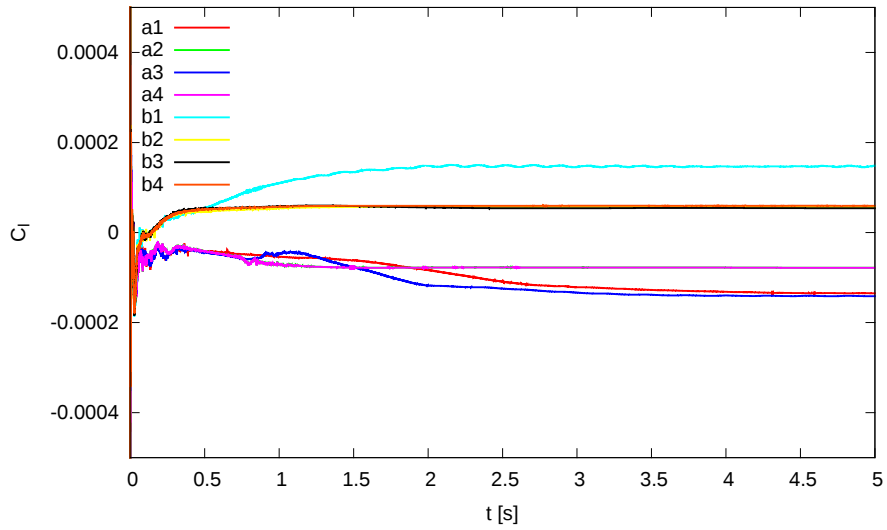


Figura 6.3: C_l in funzione del tempo per il caso a 0°.

mesh	C_d	C_l
a_1	1.01874 x 10^{-1}	-13.4940 x 10^{-5}
a_2	1.01422 x 10^{-1}	-7.82189 x 10^{-5}
a_3	1.08230 x 10^{-1}	-14.1424 x 10^{-5}
a_4	1.01411 x 10^{-1}	-7.87764 x 10^{-5}
b_1	1.01750 x 10^{-1}	14.8666 x 10^{-5}
b_2	1.00918 x 10^{-1}	5.69054 x 10^{-5}
b_3	1.00984 x 10^{-1}	5.87680 x 10^{-5}
b_4	1.00636 x 10^{-1}	6.04700 x 10^{-5}

Tabella 6.2: Tabella con i risultati giunti a convergenza di tutte le mesh per il caso 0°.

tabili dal punto di vista fisico e se discostano tra loro di una quantità non accettabile al variare della mesh.

Essendo il dispositivo in posizione fissa con angolo di attacco zero gradi ed con il flusso avente direzione solo lungo l'asse della corda, teoricamente il coefficiente di portanza dovrebbe assumere il valore nullo.

Questa comunque è solo un'indicazione, poichè il caso che è stato simulato si discosta dalle ipotesi di questa teoria. Infatti la geometria creata presenta degli smussi al *leading edge* ed al *trailing edge*, i quali non sono presenti nella lastra della soluzione teorica, inoltre l'ala non presenta dimensioni infinite come vorrebbe la teoria.

Dal grafico del C_l, plottato in valore assoluto, in funzione del tempo, possiamo apprezzare che il risultato finale assume in ogni caso valori del tutto plausibili, in linea con la teoria, dato che tutte le mesh rilevano un coefficiente di portanza praticamente nullo.

Partendo dall'analisi delle griglie a, si può notare che si possono considerare solo due valori del C_l trovati, date le minime differenze che sussistono. Infatti le mesh a_1 e a_3 valutano un divario del coefficiente di portanza del 4% tra loro del valore di a_3, mentre a_2 con a_4 discostano fra loro dello 0.7% del valore di a_4. Inoltre se si considerano i valori calcolati da a_3 ed a_2 è presente uno scostamento del 44% del valore valutato da a_3. La differenza tra a_1 e a_4 è pressochè identica. La valutazione dell'indipendenza dei risultati in funzione della risoluzione della mesh potrebbe apparire difficoltosa, poichè all'aumentare della raffinatezza della griglia le soluzioni si assestano in un range di valori non decrescente.
Ciò non toglie comunque il fatto che la famiglia di griglie a colga un risultato accettabile, in linea con la teoria.

Per la famiglia b l'analisi è più semplice, in quanto gli scostamenti tra le varie mesh risultano meno marcate. L'unica differenza più sostanziale si ha passando dalla griglia senza refinement a quella con refinement box, dove il divario tra b_1 e b_2 è pari al 60% del valore di b_1. Invece lo scostamento

medio tra i casi 2, 3, 4 è pari a circa al 3% del valore di b_4. Questo risultato mostra come i casi con refinement box diano risultati molto simili fra loro con il variare della risoluzione della mesh in prossimità della geometria.

I valori ottenuti del C_l possono ritenersi soddisfacenti dal punto di vista dell'analisi di grid dependency, dato che l'ordine delle fluttuazioni del risultato rimane sull' ordine del 10^{-4} sia per i casi b che a. Risultano più affidabili comunque i componenti della famiglia b con refinement box, in quanto forniscono risultati più simili fra loro.

Sul coefficiente di resistenza possiamo appoggiarci sul risultato teorico della teoria della lastra piana, che afferma:

$$C_d = \frac{1.328}{\sqrt{Re}} \tag{6.3}$$

con Re numero di Reynolds.

Nel caso in esame bisognerà moltiplicare per due tale valore, in quanto il corpo è investito dal fluido su due facce. Quindi il risultato teorico è:

$$C_d = 2\frac{1.328}{\sqrt{1000}} = 0.084 \tag{6.4}$$

I risultati delle simulazioni mostrano diferenza tra risultato computazionale e teorico di circa del 16%, tale scostamento è imputabile alle dimensioni non infinite della geometria e dal bordo d'attacco smussato. Possiamo ritenerci comunque soddisfatti per il calcolo di entrambi i coefficienti aerodinamici, affermando che OpenFOAM, in questo caso, riesce a trovare valori plausibili con la fisica reale del problema.

Ritornando all'analisi di grid dependency, si nota che l'ordine di grandezza del C_d è uguale per tutte le griglie, stabile su valori nell'intorno di 10^{-1}. Lo scostamento massimo che si ha tra i risultati trovati è tra la griglia a_1 e b_4, ovvero la più grezza con la più fine, ed è pari all' 1.2% del valore fornito da a_1. La presenza del box di raffinamento, in questo caso, non fa variare significativamente il valore del C_d, in quanto per a_1 e a_2 si ha un divario pari a 0.44% del valore di a_1, mentre per b_1 e b_2 una differenza di 0.82% rispetto al valore di a_1. Per i casi a, aventi refinement box, lo scostamento maggiore lo si registra tra i casi a_3 e a_4 che vale 0.4% del valore di a_3, mentre per i casi b, il massimo scostamento è tra b_2 e b_4 che misura 1.1% di b_1.
Dal monitoraggio del coefficiente di resistenza, si può apprezzare come ci sia una più sostanziale indipendenza del risultato dalla risoluzione della griglia rispetto al coefficiente di portanza, sia per i casi a che b, avendo una banda di incertezza pari mediamente all' 1% del valore più elevato trovato.

Combinando i risultati delle analisi dei due coefficienti, si può affermare che le mesh dei casi b con refinement box forniscono risultati affidabili e dipendenti dalla risoluzioni di pochi punti percentuali. Inoltre griglie più spinte

del caso b_4 non permetterebbero di avere un rapporto *soluzione migliore - tempo di simulazione* soddisfacente.

Concludiamo quindi che da questa prima analisi statica con l'ala fissata ad angolo d'attacco 0°, le mesh della famiglia b con refinement box mostrano una maggiore affidabilità rispetto alla famiglia a, sia nel calcolo del C_l sia nel calcolo del C_d, date le differenze esigue che sussistono tra i vari valori trovati. Un'ultima osservazione riguardo questo caso a 0° può essere fatta sulla valutazione dei profili di velocità sulla superficie del dispositivo. Da quest'analisi ci si potrebbe aspettare nuovamente una migliore affidabilità dei casi con refinement box, come accaduto per l'analisi dei coefficienti aerodinamici, in quanto queste mesh possiedono un numero maggiore di punti. Come si può vedere dal grafico 6.4, il quale confronta i profili di velocità del fluido valutati da a_1 e b_4, che gli andamenti della velocità sono pressochè identici sia per la mesh più grezza e sia per quella più fine. Sulla superficie del corpo, come si era detto, si era imposto un livello di raffinamento pari a 6 per tutte le griglie considerate. Essendo che la famiglia di mesh a ha risoluzione doppia della famiglia b, i casi b possiedono il doppio dei punti nodali sulla geometria rispetto ai casi a. Nonostante sussista questa differenza di dimensione delle celle, ciò non influisce sul risultato finale, evidenziando che i profili di velocità sono colti in maniera precisa anche dai casi a. Anche attraverso il monitoraggio di questa grandezza, si può concludere che le mesh mostrano risultati non dipendenti dalla risoluzione.

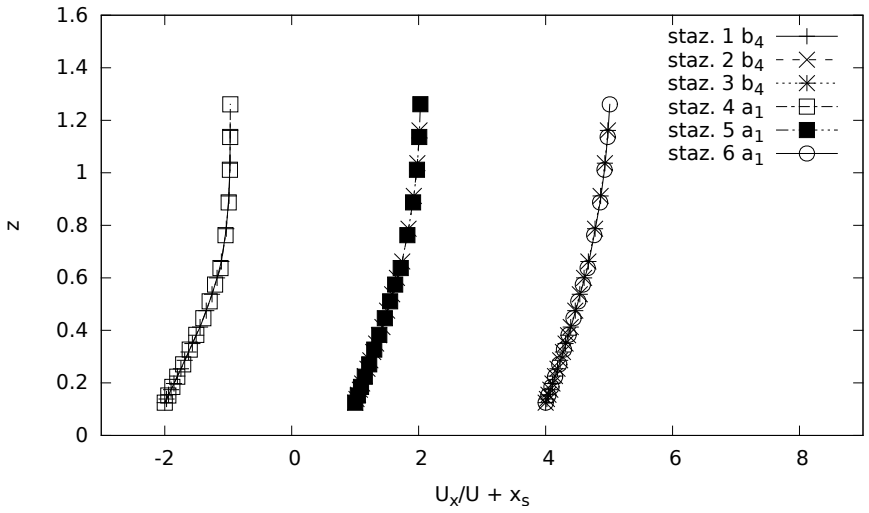


Figura 6.4: Illustrazione dei profili di velocità sulla superficie dell'ala. Sulle ordinate è riportata la coordinata z, mentre in ascissa il valore della velocità adimensionalizzata con la velocità indisturbata del fluido, valutata su un'ascissa costante. L'intervallo di estensione dell'ala lungo x è [-5:5].

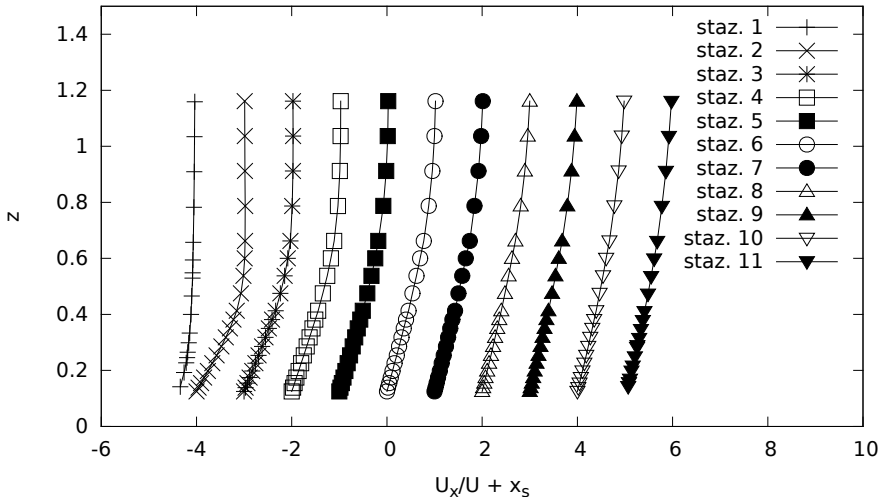


Figura 6.5: Valutazione dello strato limite sulla superficie dell'ala che si estende lungo x da [-5:5], valutato dalla griglia b_4.

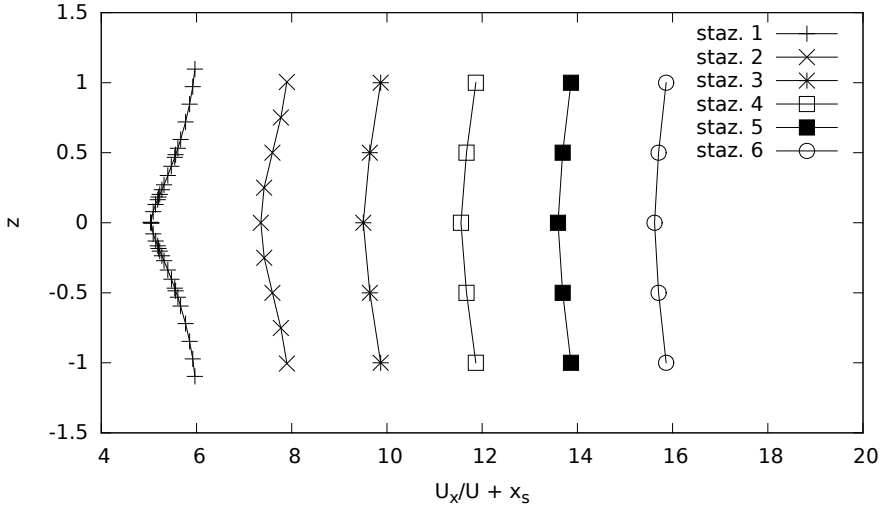


Figura 6.6: Valutazione della scia in coda all'ala che si estende lungo x da [-5:5], valutato dalla griglia b_4.

Questa analisi sul confronto dei profili di velocità (Fig. 6.4) è stata svolta tramite l'utility *Sample* che ha la funzione di campionare lungo una linea i valori trovati di un'incognita del problema. Lavora tramite l'uso del dizionario *SampleDict* dove bisogna specificare la direzione e l'estensione della linea su cui deve essere eseguito il campionamento. Si riportano anche i profili di velocità (Fig. 6.5) valutati tra il bordo di ingresso e di uscita con stazioni distanziate di $\frac{1}{10}c$, e la valutazione della scia in coda al profilo (Fig. 6.6). Si può osservare che lo spessore dello strato limite cresce avvicinandosi al *trailing edge* e il difetto di velocità risulta quasi assente a una distanza pari c.

6.3.1 Risultati cfMesh

In questa sezione si mostreranno i risultati dei coefficienti aerodinamici trovati con *cfMesh* e si confronteranno con gli esiti ottenuti con *snappyHexMesh*. Per testare questo *meshatore* si condurranno solo due casi b_2 e b_4, che hanno mesh molto simili ai casi costruiti con *snappyHexMesh*, l'unica differenza sostanziale risiede nella forma attribuita all'ala che possiede degli spigoli vivi al bordo di attacco e di uscita invece che essere smussata.

Si può apprezzare dalla figura 6.7 che il coefficiente di resistenza è stato calcolato in maniera pulita sia da b_2 che b_4 e il risultato è giunto a un valore stabile, con uno scostamento tra i due casi pari al 2% del valore di b_4, mentre il coefficiente di portanza possiede del rumore numerico che non permette di valutare un'esatta quantità del C_l (Fig. 6.8). Questo disturbo è dovuto alla presenza degli spigoli vivi della geometria o la risoluzione sulla superficie non risulta sufficiente a cogliere al meglio la soluzione. L'entità del rumore risulta comunque molto tenue in quanto l'ampiezza di oscillazione del coefficiente è pari a un ordine di 10^{-6}; è piuttosto visibile perchè la scala del C_l è molto vicina allo zero. Chiaramente il picco che si nota a circa a un secondo non è dovuto a un fenomeno fisico ma numerico.

Sempre dalla figura 6.7 si può apprezzare come il C_d, calcolato da *cfMesh*, risulti superiore del 4% rispetto al coefficiente valutato da b_4 di *snappyHexMesh*. Questo è un risultato plausibile poichè il profilo utilizzato ha una forma meno aerodinamica per la presenza degli spigoli vivi, che aumentano il contributo del *pressure drag*.
Riguardo al confronto del C_l (Fig. 6.8) invece è possibile fare solo un discorso qualitativo, a causa del rumore numerico presente. Le griglie costruite con *cfMesh* valutano un coefficiente di portanza inferiore dell'80% del valore fornito dalle mesh eseguite con *snappyHexMesh*. Questo fatto è dovuto nuovamente alla forma aerodinamica del corpo.
Le soluzioni finali possono essere ritenute comunque plausibili, poichè è stato rilevato un coefficiente di resistenza maggiore di quello rilevato dalle mesh con *snappyHexMesh* per motivi di forma, mentre il coefficiente di portanza è prossimo a zero come suggerisce la teoria. Quindi anche utilizzando questo

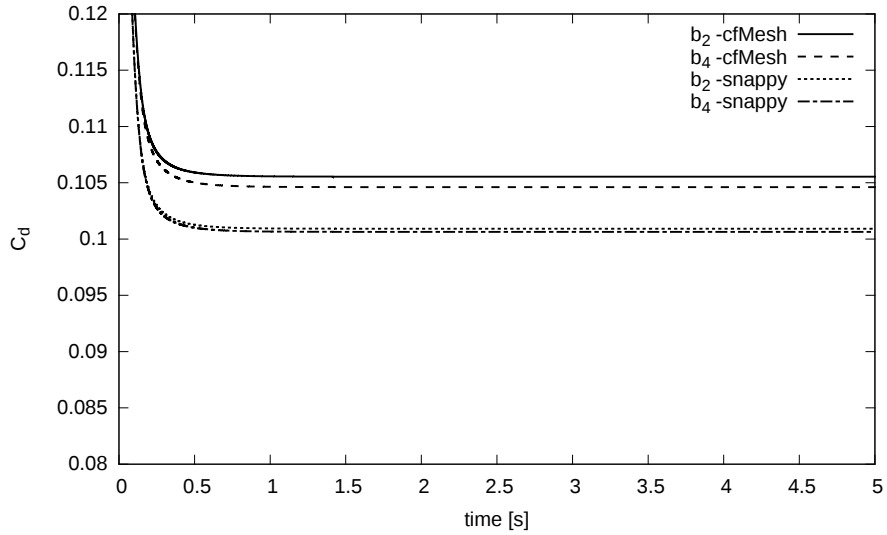


Figura 6.7: Confronto del coefficiente di resistenza trovato con *cfMesh* e *snappyHexMesh* per il caso a 0°.

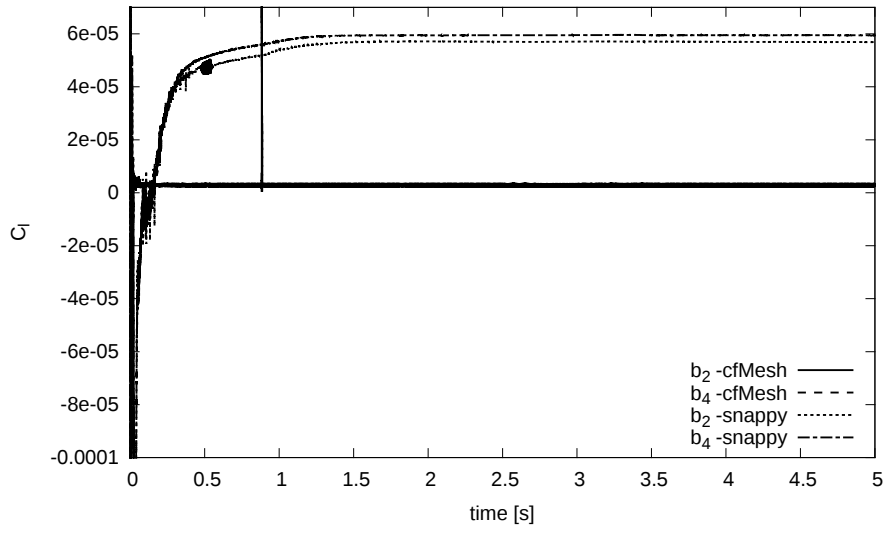


Figura 6.8: Confronto del coefficiente di portanza trovato con *cfMesh* e *snappyHexMesh* per il caso a 0°.

pacchetto di *meshing* OpenFOAM può essere ritenuto un codice di calcolo affidabile.

6.4 Configurazione a 5°

In questa sezione si simulerà l'ala disposta con angolo di attacco pari a 5°. Si vuole capire se anche per questa situazione il software OpenFOAM riesca a fornire una soluzione affidabile. Il metodo di indagine è sempre lo stesso, si valuterà la dipendenza dei risultati in funzione della risoluzione delle otto mesh presentate nel paragrafo 6.2 . Le grandezze che si monitoreranno saranno nuovamente il C_d e il C_l. Le mesh non saranno ritenute affidabili se i coefficienti assumeranno valori significativamente distanti uno dall'altro. Si riportano nella tabella 6.3 tutti i risultati trovati giunti a convergenza.

Dai grafici si può apprezzare anche che il tempo di simulazione è sufficiente a far sviluppare completamente il flusso sul corpo, infatti i valori dei coefficienti si assestano su valori fissi dopo poco più di un secondo. Ciò che si può notare immediatamente dalla tabella 6.3 è che l'ordine di grandezza dei risultati è lo stesso per ogni mesh, con differenze dell'ordine del millesimo e coefficienti di portanza tutti con segno concorde. Analizziamo ora in maniera più approfondita i valori trovati.

Per quanto riguarda il coefficiente di resistenza si ha che l'introduzione del box di raffinamento non porta a una variazione sensibile della soluzione. Infatti tra il caso a_1 e a_2 si ha uno scostamento pari allo 0.28% di a_1, mentre tra b_1 e b_2 sussite una differenza pari allo 0.65% di b_2. La differenza massima si ha tra i casi a_1 e b_4 che è rilevata pari all'1.2% di a_1. Dal punto di vista della grid dependency ci si può ritenere molto soddisfatti in quanto gli scostamenti percentuali, rispettivamente per le due famiglie di mesh, sono mediamente intorno all'1.4% del valore massimo dell'elemento della famiglia considerata e le differenze assolute sono dell'ordine del millesimo. Dal punto di vista aerodinamico invece si nota che si ha un aumento del 30% del C_d rispetto al caso con ala fissa a 0°. Questo è dovuto principalmente alla nuova configurazione del dispositivo, il quale mostra un'aerea di impatto maggiore rispetto alla direzione del flusso indisturbato, aumentando così il contributo del *pressure drag*.

Il coefficiente di portanza invece mostra un variazione poco più significativa della soluzione se si introduce il box di raffinamento. Infatti tra a_1 ed a_2 c'è uno scostamento del 2% del valore di a_1, mentre tra b_1 e b_2 c'è una differenza pari al 3% del valore di b_1. Lo scostamento massimo si realizza tra la griglia più fine (b_4) e quella più grezza (a_1), che è pari al 4.7% del valore di a_1. Comunque mediamente tra i casi con *refinement box* di entrambe le famiglie, sussiste uno scostamento pari all'1% del valore di a_2 o b_2. Quindi anche per il coefficiente di portanza si può concludere che oltre al caso b_4 sarebbe superfluo spingersi. Dal punto di vista aerodinamico, possiamo ap-

mesh	C_d	C_l
a_1	1.135764 x 10^{-1}	4.699548 x 10^{-1}
a_2	1.132568 x 10^{-1}	4.605648 x 10^{-1}
a_3	1.135805 x 10^{-1}	4.658740 x 10^{-1}
a_4	1.132577 x 10^{-1}	4.605533 x 10^{-1}
b_1	1.129836 x 10^{-1}	4.638450 x 10^{-1}
b_2	1.124290 x 10^{-1}	4.493585 x 10^{-1}
b_3	1.123135 x 10^{-1}	4.542169 x 10^{-1}
b_4	1.122126 x 10^{-1}	4.478197 x 10^{-1}

Tabella 6.3: Risultati dei coefficienti aerodinamici ottenuti a fine simulazione per il caso 5°.

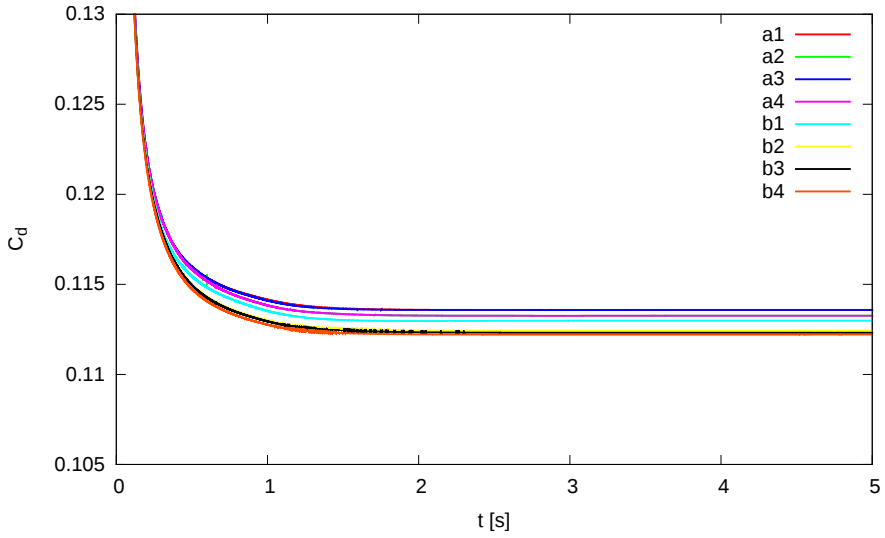


Figura 6.9: Andamento del C_d in funzione del tempo di simulazione dell'ala con angolo di attacco 5°.

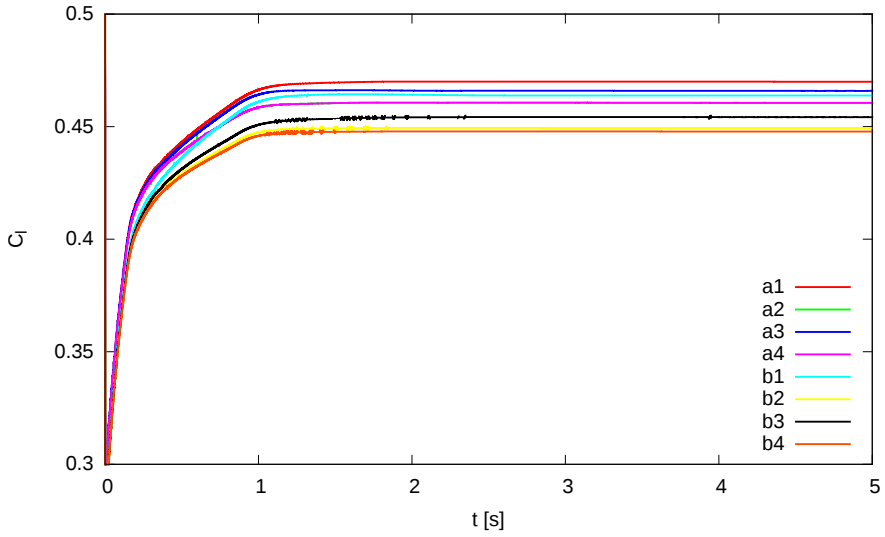


Figura 6.10: Andamento del C_l in funzione del tempo di simulazione dell'ala con angolo di attacco 5°.

prezzare che il C_l è notevolmente aumentato rispetto al caso di ala fissa a 0°. Questo è dovuto alla dipendenza che sussiste tra la portanza e tale angolo di attacco. Il C_l infatti aumenta con l'angolo fino a un certo valore per poi decrescere per inclinazioni maggiori. Ciò è determinato dal fenomeno dello stallo che si realizza quando il punto di separazione dello strato limite si porta sul bordo di attacco del profilo. Nello sviluppo di questo fenomeno il fluido risulta in quantità minore attaccato alle pareti del corpo, esercitando quindi forze non sufficienti a sostenere il profilo.
Anche per il caso a 5° si può affermare che, utilizzando una griglia più fine del caso b_4, si troverebbero risultati molto simili a scapito di tempi di simulazioni molto più lunghi.

6.5 Configurazioni a 20°, 40° e 60°

Al fine di valutare l'andamento dei coefficienti aerodinamici in funzione dell'angolo di attacco si sono condotte tre ulteriori simulazioni con dispositivo fisso a 20°, 40° e 60°, in modo tale da avere un'idea di come l'*energy harvester* reagisca a diverse inclinazioni.

La differenza sostanziale che sussiste con i casi a 0° e 5° è che in questi tre nuovi casi non si trova un valore costante dei coefficienti aerodinamici dopo un certo tempo di simulazione ma si osservano valori variabili (Fig. 6.11 e 6.12). Il motivo dell'oscillazione dei risultati risiede nel fatto che a queste inclinazioni più spinte il flusso diviene instazionario, cioè il campo di pressione e velocità non dipende solamente dalle coordinate spaziali ma anche

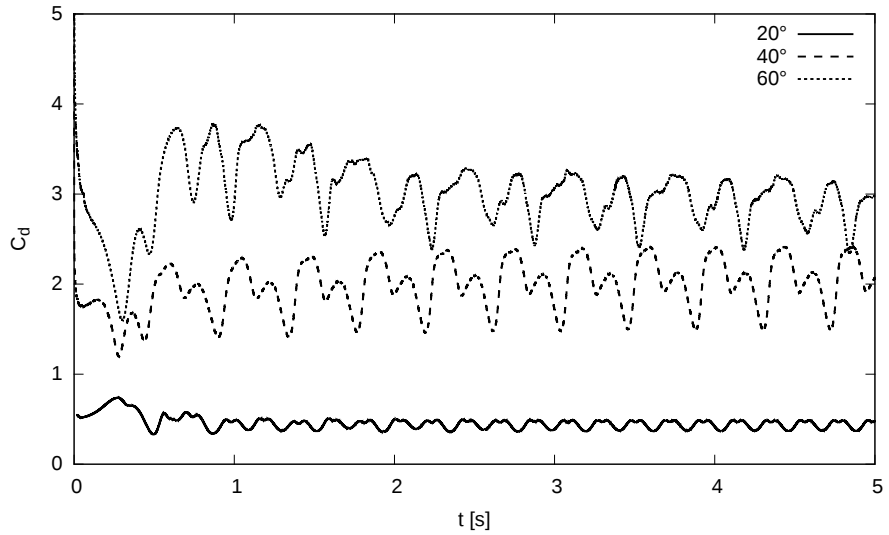


Figura 6.11: Andamento del coefficiente di resistenza in funzione del tempo per i tre casi a 20°, 40° e 60° valutati dalla griglia b_2.

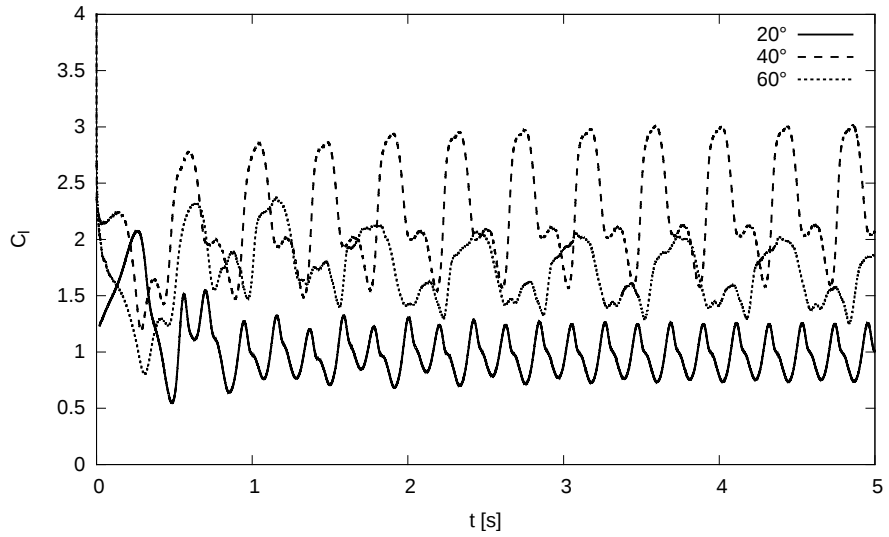


Figura 6.12: Andamento del coefficiente di portanza in funzione del tempo per i tre casi a 20°, 40° e 60° valutati dalla griglia b_2.

dal tempo. Attraverso l'applicazione *paraView* è stato possibile visualizzare graficamente la soluzione e si è rilevato che tutti e tre i casi presentano il fenomeno dello *shedding* (Fig. 6.13 e 6.14), il quale è un fenomeno che consiste nella formazione periodica di vortici a seguito del distacco dello strato limite dall'ala. Il motivo della separazione del flusso sta nel fatto che il fluido non riesce a rimanere attaccato al corpo a causa dell'angolo di attacco troppo elevato.

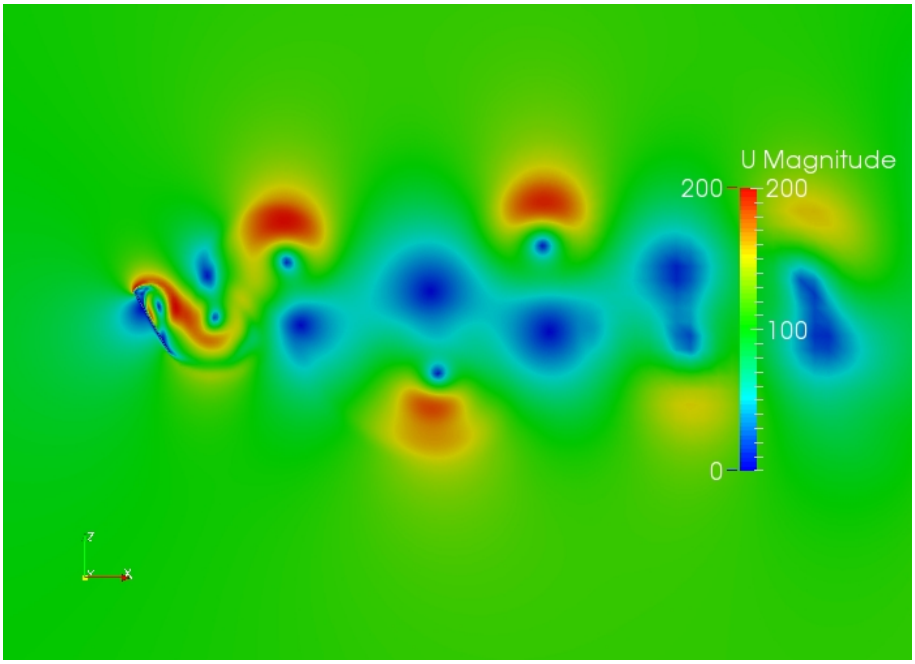


Figura 6.13: Istantanea che raffigura il fenomeno dello *shedding* attraverso la visualizzazione del campo di velocità, da una simulazione svolta con dispositivo a 60°.

Per questi tre nuovi casi non è stata condotta un'accurata analisi di grid dependency poichè lo scopo principale era quello di ottenere ulteriori valori dei coefficienti aerodinamici per ottenere in via preliminare la polare del profilo. Si sono svolte comunque le simulazioni con quattro griglie a_1, a_2, b_1 e b_2 per valutare se le mesh fornivano risultati simili fra loro. Solo per il caso a 40° si nota un marcato scostamento dal valore medio del C_d e del C_l stimato dalla mesh b_2 (Fig. 6.15 e 6.16). Per la mancanza di tempo non si è potuto indagare con griglie più raffinate il motivo di questo risultato.
Dal grafico 6.15 possiamo notare come il coefficiente di resistenza abbia un andamento crescente, poichè aumentando l'angolo di attacco aumenta anche il contributo del *pressure drag*, per la maggior area di impatto tra flusso e corpo che si viene a creare per inclinazioni maggiori e la differenza di pressione tra monte e valle. Dal grafico 6.16 invece viene mostrato che il coefficiente di

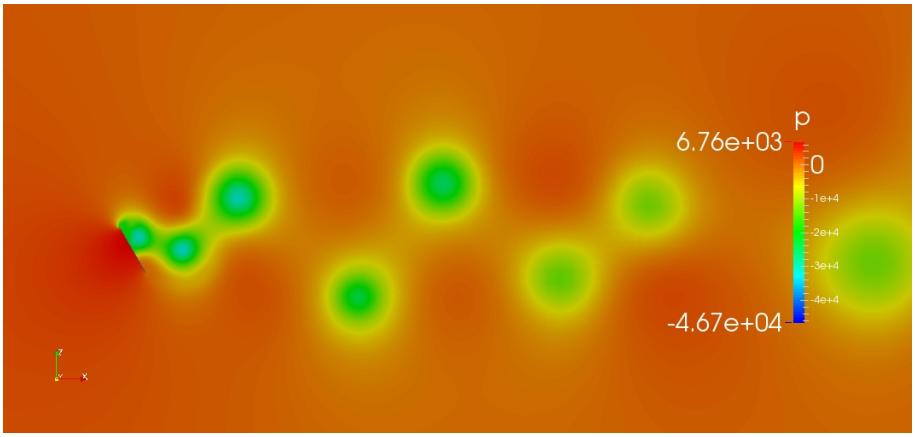


Figura 6.14: Istantanea che raffigura il fenomeno dello *shedding* attraverso la visualizzazione del campo di pressione, da una simulazione svolta con dispositivo a 60°.

Angolo di attacco	C_d	C_l
0°	0.1	7 x 10^{-5}
5°	0.12	4 x 10^{-1}
20°	0.43	0.96
40°	1.5	1.64
60°	2.85	1.6

Tabella 6.4: Risultati dei coefficienti aerodinamici medi ottenuti per diversi angoli di attacco.

portanza possiede un andamento a massimo al variare dell'angolo di attacco, come accade per ogni profilo, inoltre si può stimare che il punto di stallo è attorno ai 50°. A livello generale invece si può apprezzare come per i casi a 0° e 5° i valori dei due coefficienti valutati dalle diverse griglie risultano più simili fra loro, mentre per gli altri casi, dove il flusso è instazionario, le differenze di risultati sono più marcate, risultando dell'ordine del decimo e del centesimo.
Perciò quando si hanno fenomeni di distacco dello strato limite la risoluzione della mesh b_2 risulta insufficiente per cogliere una soluzione precisa, mentre per i casi stazionari b_2 può dimostrarsi sufficiente.

Infine si riporta la polare del profilo (Fig 6.17) con la relativa tabella dei valori medi dei coefficienti aerodinamici (Tab. 6.4). Nel campo aeronautico questo grafico risulta molto importante, poichè descrive come varia il coefficiente di portanza in funzione del coefficiente di resistenza a diversi angoli di attacco. Dal diagramma si può evidenziare che per piccole inclinazioni si hanno forti incrementi del coefficiente di portanza e limitate variazioni del coefficiente di resistenza, mentre per inclinazioni maggiori l'aumento della

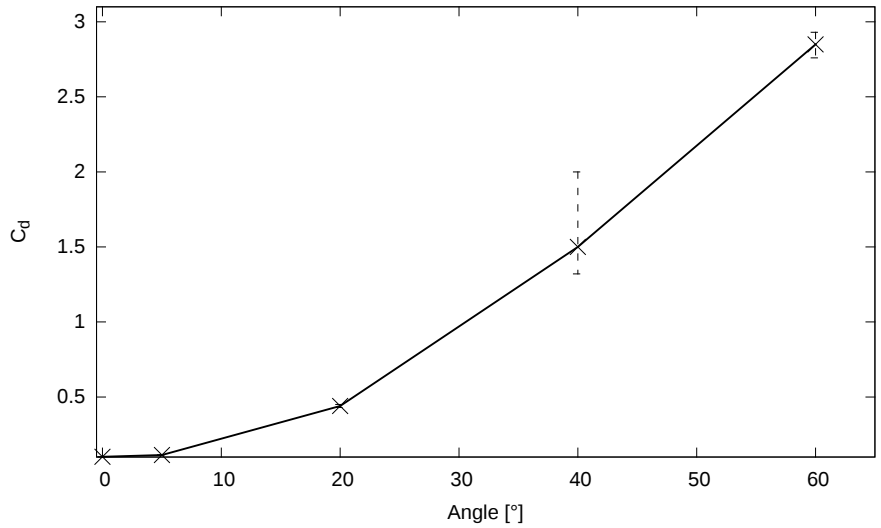


Figura 6.15: Andamento del coefficiente di resistenza in funzione dell'angolo di attacco. Il valore sulla curva rappresenta il valore medio valutato dall'analisi di grid dependency, mentre la linea tratteggiata rappresenta il range dello scostamento massimo e minimo da tale valore medio.

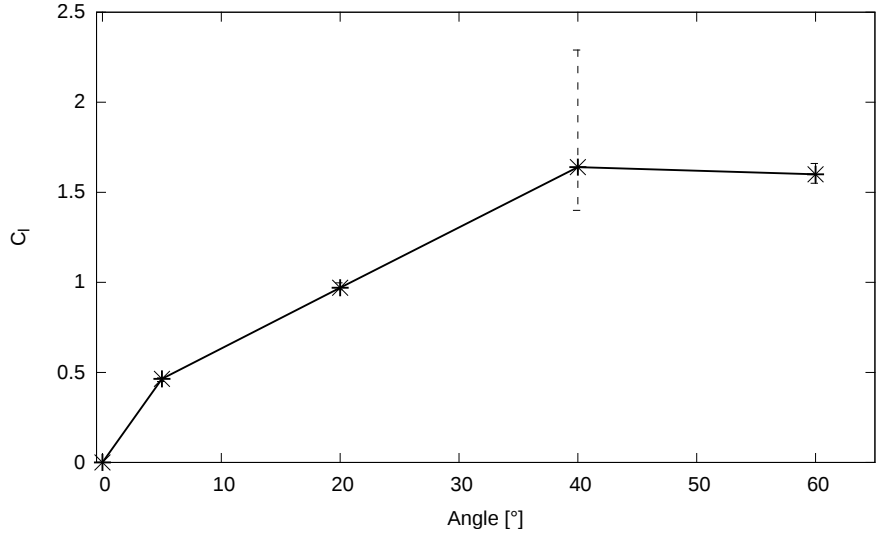


Figura 6.16: Andamento del coefficiente di portanza in funzione dell'angolo di attacco. Il valore sulla curva rappresenta il valore medio valutato dall'analisi di grid dependency, mentre la linea tratteggiata rappresenta il range dello scostamento massimo e minimo da tale valore medio.

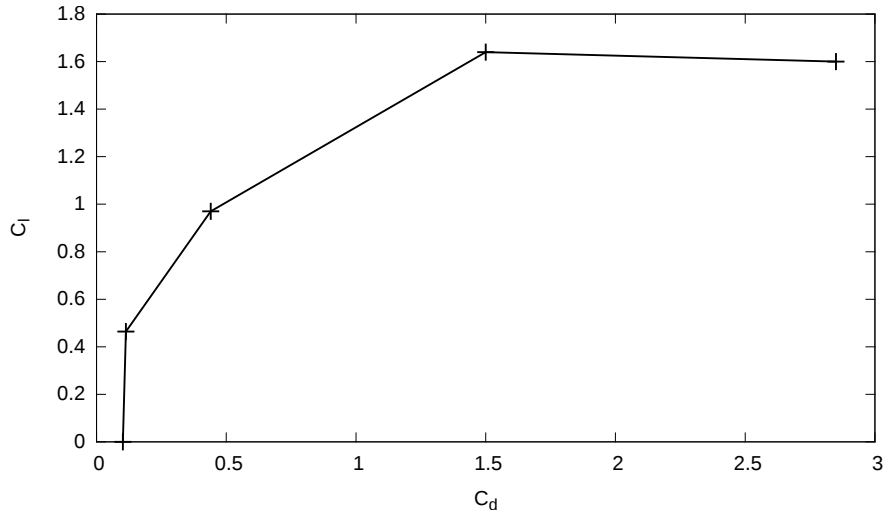


Figura 6.17: Polare del profilo costruita a partire dagli angoli di attacco 0°, 5°, 20°, 40° e 60°.

portanza viene pagato con un forte aumento della resistenza.

6.6 Conclusioni sui risultati statici

Da questo primo approccio con funzionamento statico del dispositivo si può affermare che il codice di calcolo OpenFOAM è in grado di giungere a soluzioni in linea con la fisica del problema.

L'analisi di *grid dependency* mostra che per bassi angoli di attacco, dove il flusso rimane stazionario è ampiamente sufficiente utilizzare una mesh di tipo b_4 per studiare il problema, mentre per i casi con inclinazioni superiore a 20° una griglia di tipo b_2 non garantisce una soluzione del tutto soddisfacente a causa della maggior complessità dei fenomeni fluidodinamici dovuti all'instazionarietà del flusso. *cfMesh* si è mostrato un ottimo *meshatore* capace di costruire griglie in maniera precisa e veloce, fornendo risultati affidabili.

Capitolo 7

Simulazioni del funzionamento reale del dispositivo

In questo capitolo si mostreranno i risultati riguardanti le simulazioni bidimensionali sul vero e proprio funzionamento del dispositivo, conducendo dapprima un'analisi di grid dependency per valutare quale mesh è opportuno utilizzare; in seguito si confronteranno i risultati di tale griglia con le soluzioni ottenute con il codice *Overture*, con cui si sono condotti i primi studi presentati in un recente articolo [21] per un determinato set di parametri importante per il moto oscillatorio del corpo. Le ipotesi chiarite nel paragrafo 2.2 saranno applicate anche a queste simulazioni.

7.1 Impostazione caso dinamico

Al fine di aleggerire la complessità del problema nel modello dinamico, per simulare l'effetto degli elastomeri, si è definita una sola molla equivalente, che rispetta la legge di Hooke, in quanto è risultata essere un'ipotesi semplificativa valida da precedenti lavori [3]. OpenFOAM possiede delle librerie apposite in cui sono modellizzati i vincoli che possono essere applicati a un corpo, i quali sono descritti attraverso dizionari, dove sono presenti le informazioni che devono essere completate dall'utente per simulare l'azione del vincolo. I calcoli svolti dal solutore danno in output risultati utili per svolgere attività di post-process, nel nostro caso sono stati forniti il vettore forza elastica o la lunghezza della molla.

Le equazioni di governo del fluido sono state risolte attraverso il solutore *pimpleDyMFoam*, il quale è un'implementazione di *pimpleFoam* che consente la risoluzione di casi con mesh dinamica. Il processo di risoluzione è più complicato rispetto a un caso con griglia statica, dove si richiedeva solamente la risoluzione del campo di moto e di pressione del fluido. Adesso si possono generare spostamenti a causa delle forze esercitate dal fluido sul corpo sotto

esame, con conseguente movimento dei punti nodali della griglia. Tali forze sono calcolate per mezzo delle equazioni cardinali della dinamica:

$$m\ddot{\mathbf{x}}_G = \mathbf{R}^{(\mathbf{e})} \tag{7.1}$$

- m: massa dell'ala. $[kg]$
- $\ddot{\mathbf{x}}_G$: accelerazione del centro di massa dell'ala. $[m/s^2]$
- $\mathbf{R}^{(\mathbf{e})}$: risultante delle forze aerodinamiche ed elastiche agenti sull'ala. $[N]$

$$I_G\ddot{\theta} = M_G^{(e)} \tag{7.2}$$

- $M_G^{(e)}$: risultante dei momenti dovuti alle forze aerodinamiche ed elastiche agenti sull'ala. $[Nm]$
- I_G : momento di inerzia del corpo rigido valutato rispetto al baricentro g. $[kg\,m^2]$
- $\ddot{\theta}$: accelerazione angolare dell'ala. $[rad/s^2]$

le due equazioni sopra menzionate sono risolte mediante l'impiego del solutore *sixDoFRigidBodyMotion*.

L'impostazione del caso dinamico passa principalmente attraverso la scrittura del dizionario *dynamicMeshDict* dove si sono dovuti specificare la massa dell'ala, i momenti di inerzia relativi ai tre assi principali del corpo, le caratteristiche della molla come la lunghezza a riposo, la rigidezza, la coordinata del punto di fissaggio all'ala e al telaio.

La differenza che sussiste con i casi studiati nel capitolo 6 è che in questi i punti nodali delle mesh non subivano alcun spostamento durante la simulazione per la staticità dell'ala, mentre nelle simulazioni del funzionamento reale del dispositivo è necessario utilizzare mesh dinamiche che permettono alla griglia di seguire gli spostamenti del corpo per ottenere una soluzione accurata.

In Overture si sono usate *overlapping grids*. Queste mesh sono create mediante sovrapposizione di più griglie e durante la simulazione con il corpo in movimento le celle non subiscono deformazioni, consentendo una totale rotazione della geometria. La difficoltà di questo metodo risiede nel creare un'interfaccia tra le due mesh che garantisca una precisione sufficiente in fase di interpolazione delle soluzioni, operazione che non risulta spesso banale.

OpenFOAM non permette ancora la creazione di *overlapping grids*, si è utilizzata dunque un'altra tecnica detta *mesh morphing*, che prevede l'utilizzo di una sola griglia che si deforma in base ai movimenti del corpo. Il *mesh morphing* si basa sulla risoluzione di una equazione alle derivate parziali,

attraverso un algoritmo denominato *SLERP*, la cui è incognita è il campo degli spostamenti dei punti nodali della griglia. Per la risoluzione di tale equazione è necessario specificare le condizioni al contorno, presenti nel file *pointDisplacement*, su tutte le *patch* del dominio. La condizione iniziale del problema prevede che il campo degli spostamenti sia pari a zero, ovvero che inizialmente la mesh risulti indeformata.

Il *mesh morphing* risulta affidabile se i movimenti del corpo sono tali da garantire i parametri di qualità della griglia. Può accadere in alcuni casi che la geometria si sposti fino al bordo del dominio, comprimendo eccessivamente le celle, e non permettendo l'adeguata risoluzione delle equazioni di governo del fluido. La creazione della mesh risulta però più semplice rispetto alle *overlapping grids*, in quanto necessita della creazione di un'unica maglia di calcolo.

Dall'articolo, ove sono presenti i risultati trovati da Overture sul funzionamento dell'ala [21], sono state definite la densità adimensionale e l'inverso della rigidezza adimensionale dell'ala, da cui è possibile ricavarsi le quantità necessarie per il calcolo della massa e dei momenti di inerzia. Nota la densità del fluido e la velocità del flusso indisturbato si definiscono le seguenti relazioni:

$$\rho_w^* = \frac{\rho_w \delta}{\rho c} \tag{7.3}$$

- ρ_w^*: densità adimensionale dell'ala.
- ρ_w: densità dimensionale dell'ala. $[kg/m^3]$
- ρ: densità del fluido. $[kg/m^3]$
- δ: spessore dell'ala. $[m]$
- c: corda dell'ala. $[m]$

$$\frac{1}{k^*} = \frac{\rho U^2}{k} \tag{7.4}$$

- k^*: rigidezza adimensionale della molla.
- k: rigidezza dimensionale della molla. $[N/m]$
- ρ: densità del fluido. $[kg/m^3]$
- U: velocità del flusso indisturbato. $[m/s]$

I risultati dell'articolo che si confronteranno riguardano la configurazione con $1/k^* = 0.35$, $\rho_w^* = 3.5$ e $1/k^* = 0.1$, $\rho_w^* = 20$. Secondo Overture, con il

primo set di parametri l'ala si ottiene un fenomeno di instabilità autosostenuta controllata partendo con un angolo di 0°, mentre il secondo set presenta un fenomeno di instabilità sotto-critica, ovvero si ottiene il fenomeno di *fluttering* autosostenuto solo se il sistema è perturbato inizialmente in maniera opportuna. Le prove con Overture sono state tutte condotte a $Re = 10000$, mentre con OpenFOAM a $Re = 1000$.

Prima di confrontare i risultati dei due codici verrà condotta un'analisi di grid dependency utilizzando il primo come set di parametri ($1/k^* = 0.35$, $\rho_w^* = 3.5$), nella quale si monitoreranno i seguenti dati: l'angolo di attacco dell'ala e la coordinata verticale (parallela alla direzione del flusso indisturbato) ed orizzontale (perpendicolare alla direzione del flusso indisturbato) del punto di attacco della molla sul corpo. Si confronteranno in seguito i risultati riguardanti l'angolo di attacco e il moto nella direzione verticale dell'ala.

7.2 Grid dependency e risultati con $\rho_w^* = 3.5$, $1/k^* = 0.35$

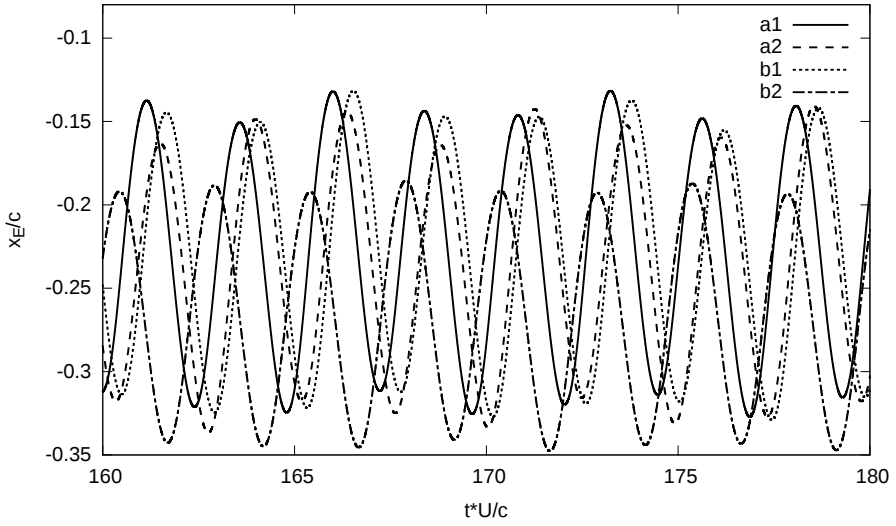
Le prime simulazioni dinamiche sono state condotte con $\rho_w^* = 3.5$, $1/k^* = 0.35$ con ala posizionata ad angolo di attacco 0°, senza perturbazioni iniziali. Con Overture si era trovato che questa configurazione dopo un certo tempo diveniva instabile a $Re = 10000$, si verificherà se con OpenFOAM si giungerà a un risultato simile a $Re = 1000$, svolgendo però prima l'analisi di grid dependency.

7.2.1 Grid dependency

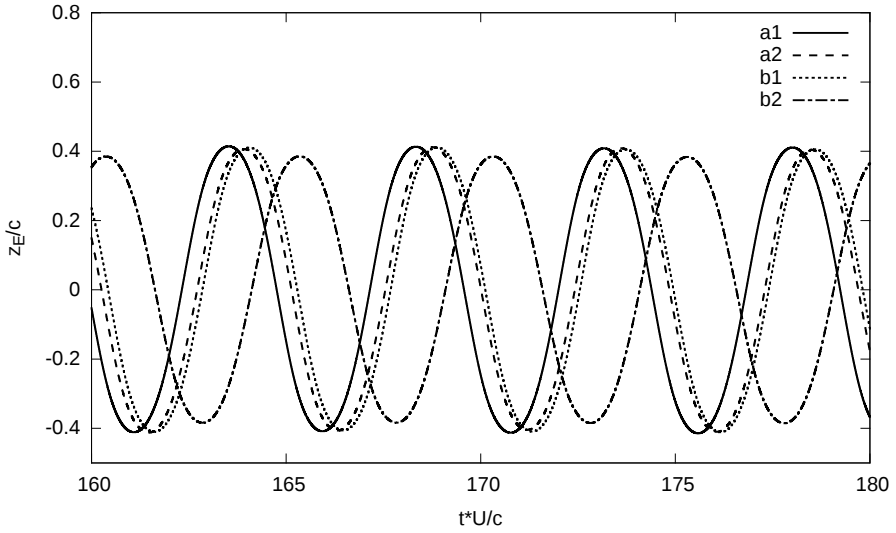
L'analisi di grid dependency prevede di monitorare lo spostamento orizzontale e verticale del punto dove è fissata la molla e l'angolo di attacco dell'ala. Si mostreranno i grafici riferiti solo al comportamento a regime del dispositivo, senza analizzare il transitorio.

Tutti i grafici sono presentati in forma adimensionale. Per la precisione le lunghezze sono adimensionalizzate con la corda c, mentre il tempo con il termine $\frac{U}{c}$, dove U è la velocità indisturbata del flusso.

Dal grafico dell'angolo in funzione del tempo si può osservare che le quattro mesh forniscono sostanzialmente gli stessi risultati. Lo scostamento massimo si ha tra il picco della griglia b_2 ed a_1, il quale è circa il 5% del picco di a_1. Lo sfasamento invece è dovuto principalmente dal tempo in cui è stato colto l'inizio del movimento di *flapping*, che dipende dall'istante in cui si è verificato lo squilibrio dei momenti, difficilmente valutabili nel loro insieme. L'unico fatto che si può dire è che le diverse mesh colgono comunque la stessa soluzione ma in momenti diversi. Questo effetto comunque non arreca alcun problema alla validazione dei risultati delle mesh considerate, poichè non in-



(a) Spostamento della coordinata orizzontale del punto di attacco della molla.



(b) Spostamento della coordinata verticale del punto di attacco della molla.

Figura 7.1: Andamento della coordinata orizzontale e verticale del punto di attacco della molla in funzione del tempo di simulazione.

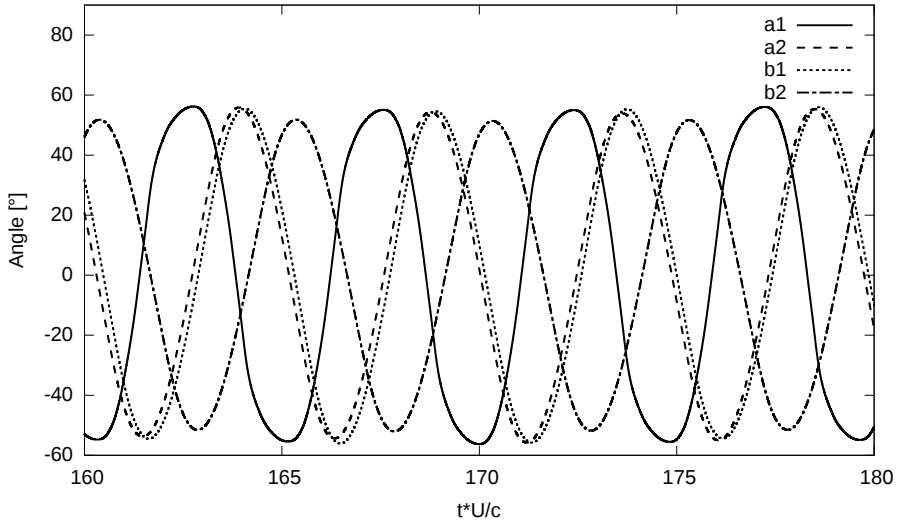


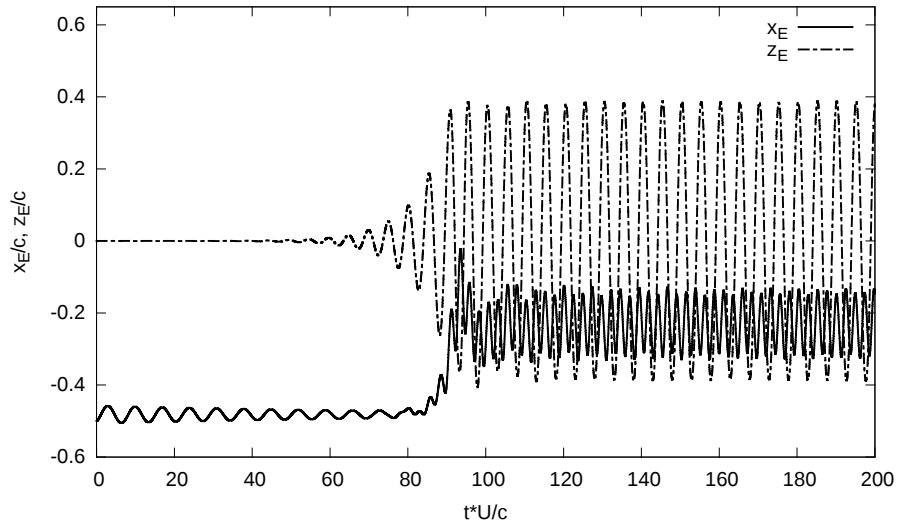
Figura 7.2: Confronto angolo di angolo di attacco per le quattro griglie.

cide sul risultato di interesse, che è appunto la variazione effettiva dell'angolo di attacco.
Si trova un risultato analogo per l'andamento del moto lungo la direzione verticale, con curve che danno quasi gli stessi valori tra loro, con scostamento massimo sempre tra la mesh a_1 e b_2 pari a circa il 4% del massimo valore valutato da a_1. Anche in questo caso si evidenzia lo stesso sfasamento citato prima, legato agli stessi effetti.

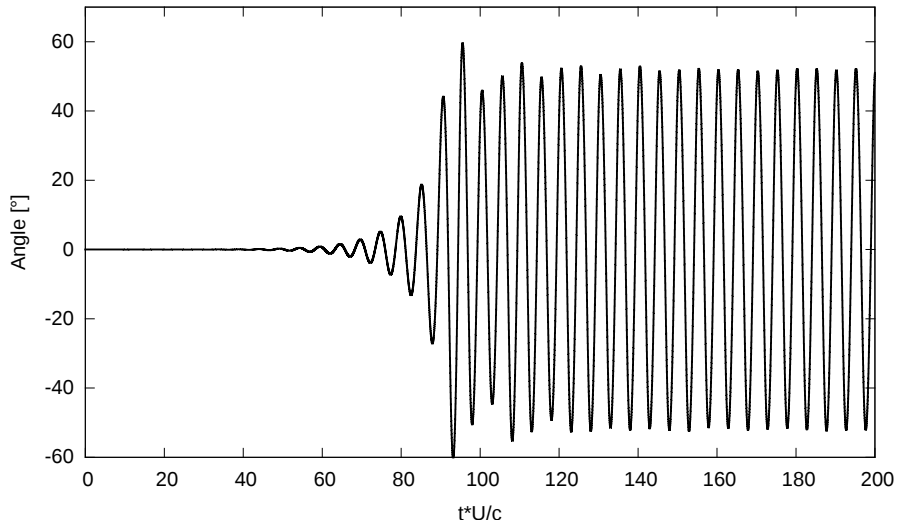
L'analisi dello spostamento del punto di attacco lungo la direzione orizzontale risulta invece un po' più complicato, in quanto si osserva che le mesh a_1, a_2 e b_1 rilevano spostamenti di entità differente senza mostrare segni di periodicità. Solo la griglia b_2 mostra una soluzione periodica.
Con questo andamento del moto lungo la direzione parallela al flusso indisturbato, è difficile definire se ci sia indipendenza dei risultati in funzione della risoluzione della mesh, poichè non si hanno valori costanti da confrontare a differenza dell'angolo di attacco e lo spostamento della coordinata verticale, i quali mostrano andamenti periodici, permettendo di calcolare agevolmente lo scostamento tra i risultati delle varie mesh. L'unico aspetto che si può trarre è che la griglia più fine, ovvero b_2, valuta una oscillazione orizzontale meno marcata rispetto alle altre tre griglie e con un carattere periodico. Se si fosse avuto ulteriore tempo si sarebbero fatte altre prove con mesh ancora più fini, per valutare se la risoluzione di b_2 fosse stata sufficiente per cogliere a dovere la soluzione. La conclusione positiva di questa analisi è che è possibile assumere la griglia b_2 affidabile, dato che le differenze dei risultati di interesse con le altre mesh è risultata accettabile.

7.2.2 Risultati con $\rho_w^* = 3.5$, $1/k^* = 0.35$

Valutata affidabile la mesh b_2, si descrive brevemente come reagisce l'ala all'azione del vento con il set di parametri $\rho_w^* = 3.5$, $1/k^* = 0.35$ e si confrontano i risultati trovati da OpenFOAM con i risultati di Overture.



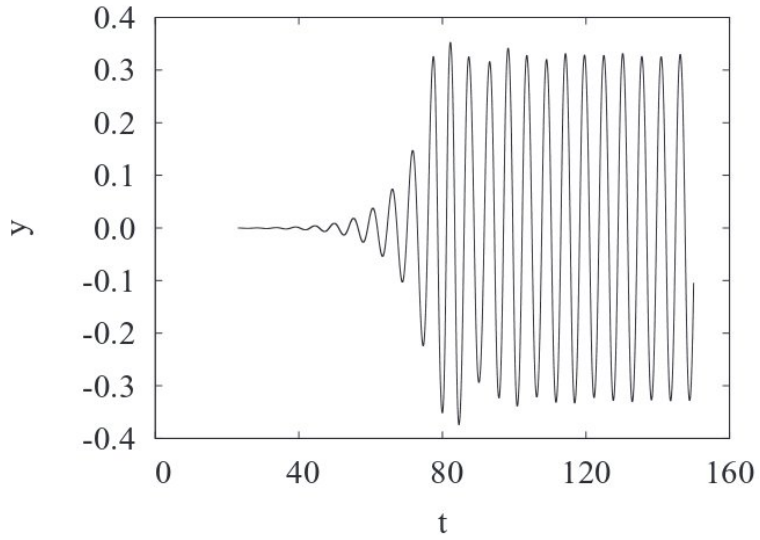
(a) Spostamento della coordinata orizzontale e verticale del punto di attacco della molla.



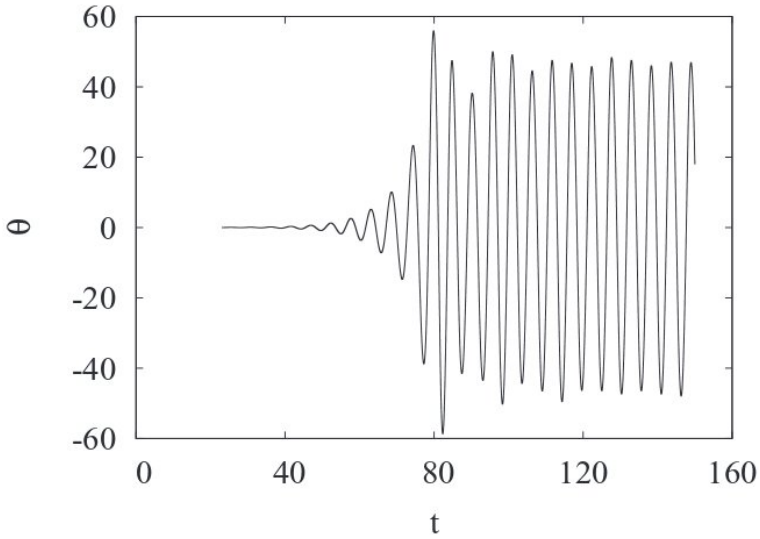
(b) Variazione dell'angolo di attacco dell'energy harvester.

Figura 7.3: Grafici in funzione del tempo del punto di attacco della molla e dell'angolo di attacco valutati per la griglia b_2.

Nella figura 7.3 sono rappresentati gli andamenti in funzione del tempo della coordinata orizzontale e verticale del punto di attacco della molla



(a) Spostamento della coordinata verticale del punto di attacco della molla valutato da *Overture*.



(b) Variazione dell'angolo di attacco dell'energy harvester valutato da *Overture*.

Figura 7.4: Grafici in funzione del tempo della coordinata verticale del punto di attacco della molla e dell'angolo di attacco valutati da *Overture*.

e dell'angolo di attacco, in particolare dal grafico (a) si può notare come i moti lungo le due direzioni raggiungano un andamento periodico stabile dopo un certo periodo. Interessante osservare come il moto orizzontale nella fase transitoria sia oscillatorio di piccola entità, in quanto la sua ampiezza pari a circa il 10% della corda, mentre lungo la direzione verticale non si può osservare alcun movimento. Questo è dovuto al fatto che il dispositivo inizialmente è disposto parallelamente al flusso d'aria, in analogia al caso statico con ala fissa a 0°, il coefficiente di portanza è praticamente nullo. Il moto nella direzione orizzontale invece è dovuto alla corrente fluida che lambisce la superficie superiore ed inferiore esercitando uno sforzo di trascinamento. Il movimento alternato osservato si sviluppa grazie all'interazione tra forze aerodinamiche e forze elastiche della molla equivalente che tende a riportarsi nella posizione di riposo.

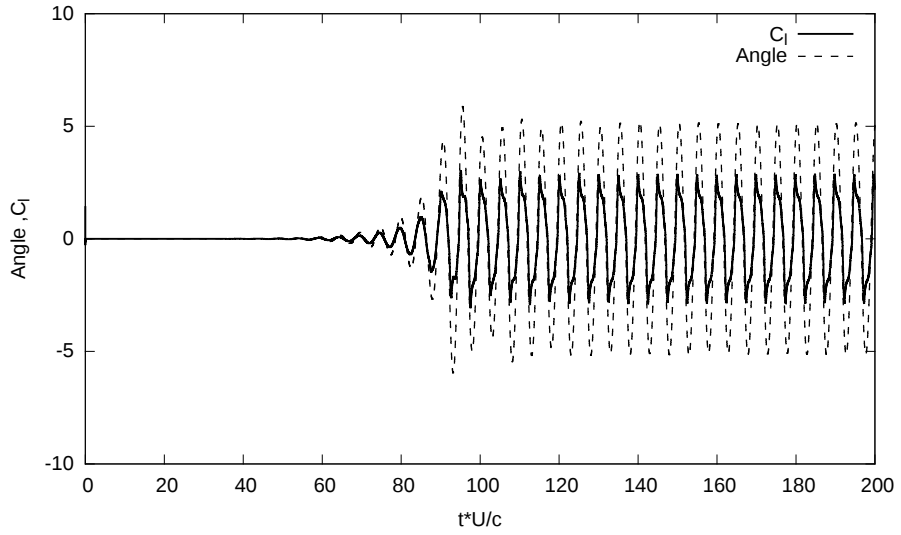
Dopo circa 60 unità di tempo adimensionali, si cominciano a vedere i primi spostamenti lungo la direzione verticale. Questo accade perchè si crea uno squilibrio dei momenti, dovuti alle forze elastiche e aerodinamiche, che tendono a far ruotare il corpo. Una volta che l'ala non è più in posizione parallela al flusso, si creano delle forze di portanza e deportanza in base al suo angolo di attacco, determinando il movimento di *flapping*, che dopo un centinaio di unità di tempo adimensionali tende a stabilizzarsi grazie all'interazione con la molla. Da questo momento si instaura un fenomeno di *fluttering* controllato, in quanto le oscillazioni non tendono a divergere.

L'andamento dell'angolo di attacco del profilo (Fig. 7.5) è fortemente correlato all'andamento dei coefficienti aerodinamici. Infatti dai seguenti grafici, in cui l'angolo è opportunamente ridotto di un fattore cento e i coefficienti sono amplificati di un fattore dieci per renderli confrontabili tra loro, si può apprezzare come l'andamento dell'angolo e dei coefficienti sia in fase, poichè a valori del C_l corrispondono angoli di attacco massimi, stesso fatto per il C_d a cui corrispondono valori massimi, per l'effetto della resistenza di forma, ad angoli di attacco massimi.

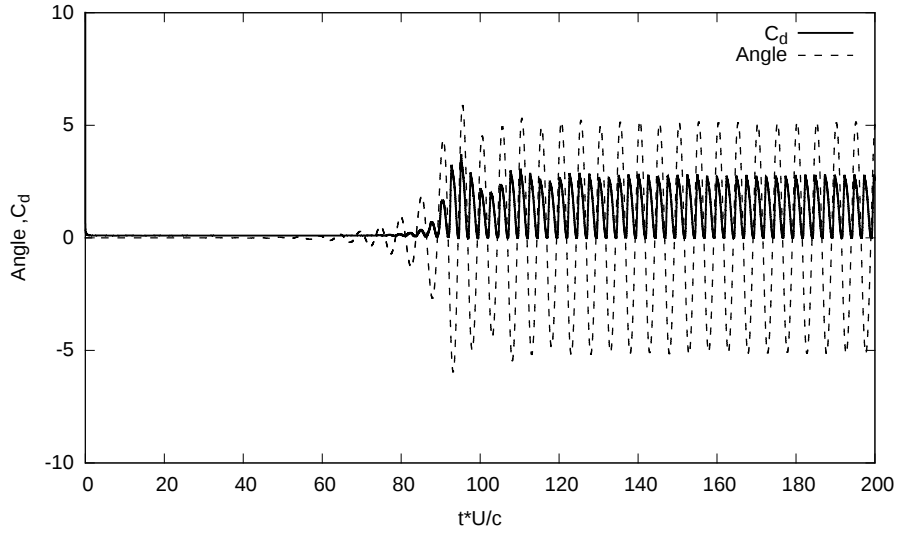
7.3 Instabilità sotto-critica ($1/k^* = 0.1$, $\rho_w^* = 20$)

Si sono svolte sette ulteriori simulazioni con un diverso valore di $1/k^*$ pari a 0.1 e ρ_w^* pari a 20. I risultati presentati nell'articolo [21] mostrano che se l'ala parte con angolo di attacco pari a 0° non si verifica un'instabilità autosostenuta, invece se la si dispone con un angolo di attacco pari a 90° il moto dell'ala diviene instabile autosostenuto. Si valuterà se con OpenFOAM si giungerà allo stesso risultato, utilizzando la griglia b_2.

Ciò che bisogna sottolineare nuovamente è il regime di Reynolds differente in alcuni casi, infatti i casi svolti con Overture sono stati condotti a $Re = 10000$, mentre con OpenFOAM le prime quattro prove effettuate sono state compiute a $Re = 1000$ e le altre tre a $Re = 10000$. Un altro elemento

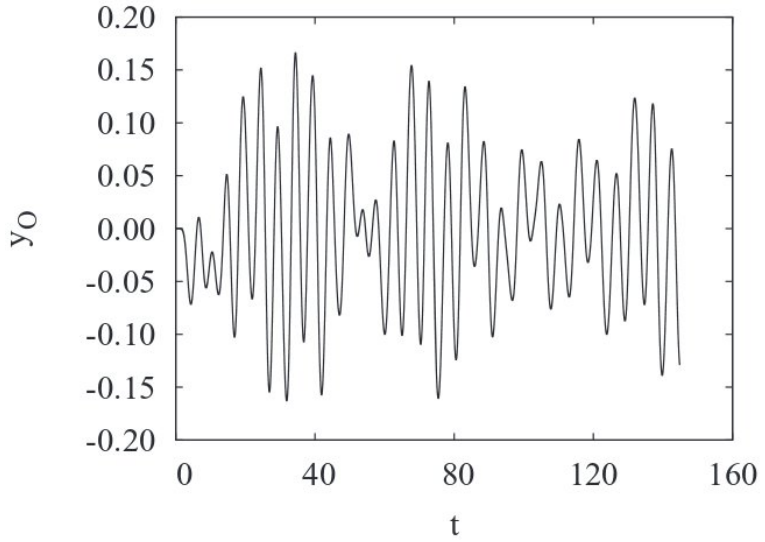


(a) Andamento dell'angolo di attacco e del C_l in funzione del tempo.

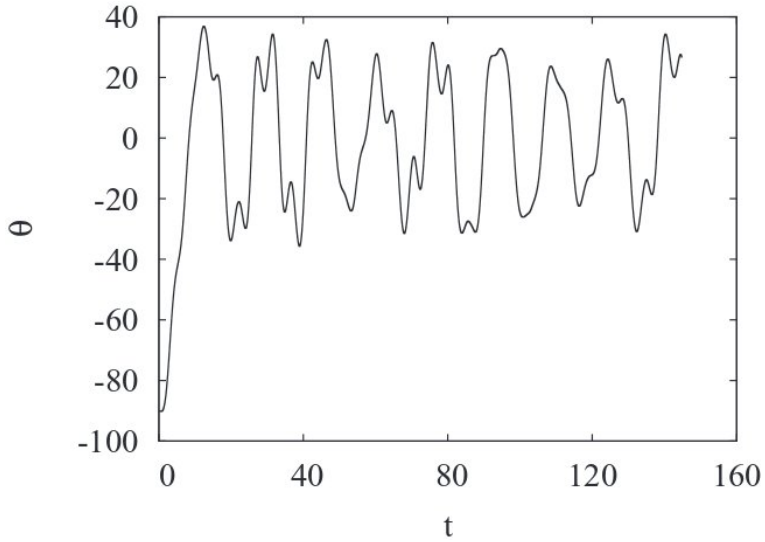


(b) Andamento dell'angolo di attacco e del C_d in funzione del tempo.

Figura 7.5: Andamento dei coefficienti aerodinamici e dell'angolo di attacco in funzione del tempo.



(a) Spostamento della coordinata verticale del punto di attacco della molla valutato da *Overture* con ala posizionata inizialmente con angolo di attacco 90°.



(b) Variazione dell'angolo di attacco dell'energy harvester valutato da *Overture* con ala posizionata inizialmente con angolo di attacco 90°.

Figura 7.6: Grafici in funzione del tempo della coordinata verticale del punto di attacco della molla e dell'angolo di attacco valutati da *Overture* con ala posizionata inizialmente con angolo di attacco 90°.

di discordanza tra le due simulazioni riguarda l'impostazione della perturbazione iniziale a causa del diverso tipo di mesh utilizzato. Il caso simulato con Overture prevedeva che l'ala fosse posizionata inizialmente con un angolo di attacco di 90° rispetto al flusso non creando alcun tipo di problema sui parametri di qualità della mesh, essendo questa una overlapping grid. Nei casi svolti OpenFOAM non si è potuto fornire tale condizione iniziale per il fatto che il mesh morphing non avrebbe garantito una soluzione precisa con una deformazione della griglia così spinta. Per emulare la condizione dell'articolo si è fornito all' ala una velocità angolare iniziale. Si passa adesso a descrivere come sono state condotte le simulazioni e risultati che si sono ottenuti.

Le prime quattro simulazioni, condotte a $Re = 1000$, si differenziano per le condizioni iniziali. La prima prevedeva di non alterare lo stato iniziale dell'ala in alcun modo e come accaduto per Overture non si sono rilevate oscillazioni lungo la direzione verticale. Nella seconda e nella terza si applicava una velocità angolare all'ala di 1 rad/s e 5 rad/s ma non si è rivelata sufficiente in quanto lo smorzamento della struttura ha riportato in posizione orizzontale l'ala dopo piccole oscillazioni. Nella quarta simulazione si è fornita all'ala una velocità verticale sia una velocità angolare con $\omega = 30$ rad/s ma anche in questo caso il *flapping* iniziale è andato a smorzarsi nel tempo.

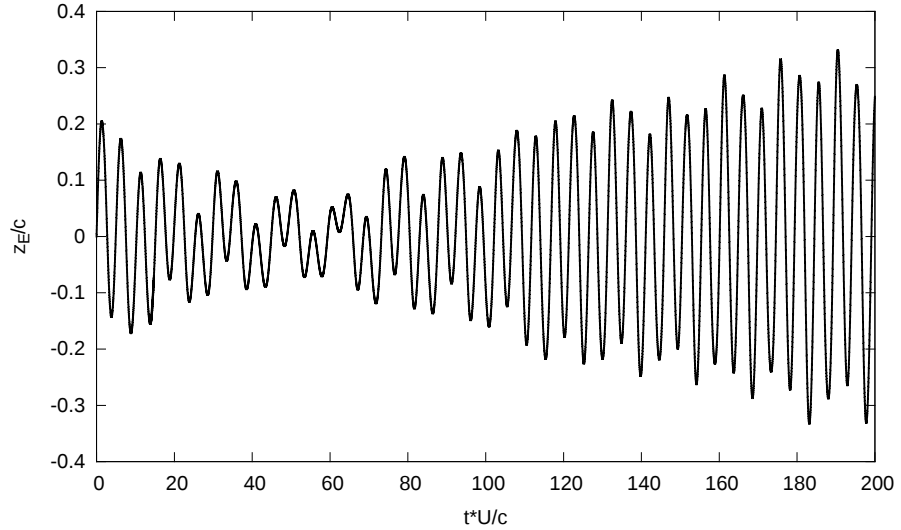
Si è dunque scelto di condurre le successive prove a $Re = 10000$ dato che le condizioni iniziali assegnate in precedenza non hanno fornito i risultati sperati. Nella quinta simulazione si è verificato che in assenza di perturbazioni l'ala non cominciasse ad oscillare. Nella sesta si è fornita una ω pari 1 rad/s, senza però dare segni di instabilità nel tempo. Nella settima si è incrementato il valore della velocità angolare a 5 rad/s dove si è osservato il fenomeno di *fluttering* autosostenuto controllato.

Dal grafico 7.7 si possono osservare i risultati ottenuti con OpenFOAM i quali si avvicinano alle soluzioni valutate da Overture (Fig. 7.6) ma non si presentano praticamente uguali come nel caso con $\rho_w^* = 3.5$, $1/k^* = 0.35$. Questo è dovuto al fatto che il problema con $1/k^* = 0.1$, $\rho_w^* = 20$ ha un carattere instabile sottocritico, il quale presenta il fenomeno del *fluttering* autosostenuto controllato solo per alcune eccitazioni iniziali, non presentando lo stesso moto per ogni disturbo che permette il *flapping* dell'ala.

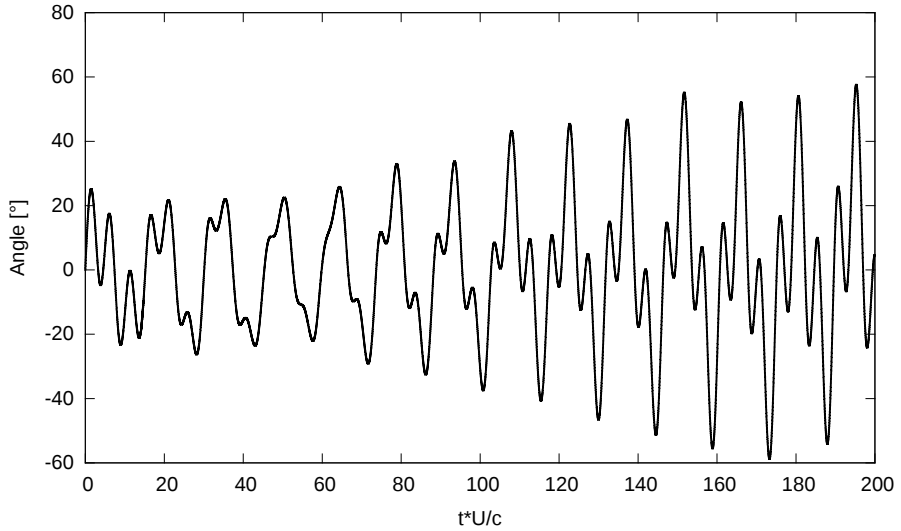
Queste simulazioni confermano i risultati dell'articolo il quale afferma che il numero di Reynolds è un parametro fondamentale ai fini dell'innesco di fenomeni instabili.

7.4 Conclusioni casi dinamici

Dai confronti che si sono svolti con i risultati dell'articolo, OpenFOAM può ritenersi un codice affidabile per questo tipo di problema, in quanto, sia per il caso con $\rho_w^* = 3.5$ e $1/k^* = 0.35$ sia per $\rho_w^* = 20$ e $1/k^* = 0.1$, si sono



(a) Spostamento della coordinata verticale del punto di attacco della molla valutato da OpenFOAM con ala dotata di velocità angolare iniziale pari a 5 rad/s.



(b) Andamento dell'angolo di attacco valutato da OpenFOAM con ala dotata di velocità angolare iniziale pari a 5 rad/s.

Figura 7.7: Grafici in funzione del tempo della coordinata verticale del punto di attacco della molla e dell'angolo di attacco valutati da OpenFOAM con ala dotata di velocità angolare iniziale di 5 rad/s.

ottenuti risultati molto simili con il software Overture. Un altro aspetto molto importante da sottolineare è che, nel caso con $\rho_w^* = 3.5$ e $1/k^* = 0.35$ in assenza di perturbazioni iniziali, OpenFOAM rileva un moto oscillatorio a $Re = 1000$ anzichè a $Re = 10000$ come ottenuto con le prove svolte con Overture. Questo fatto è molto incoraggiante ai fini del funzionamento reale del dispositivo poichè mostra che, anche in caso di deboli forze agenti sul dispositivo, è possibile ottenere un moto oscillatorio, fondamentale per estrarre energia.

Capitolo 8

Conclusioni

Lo scopo di questa tesi era quello di verificare se il codice di calcolo OpenFOAM potesse essere un software idoneo per contribuire al proseguimento degli studi del progetto FLEHAP. Il lavoro che è stato svolto mostra come i risultati trovati siano in linea con la realtà fisica per le simulazioni con mesh statica e in accordo con i dati sperimentali e le simulazioni svolte col software Overture per i casi con griglia dinamica.

Per i casi statici si è evidenziato che la risoluzione della mesh, per i casi con flusso stazionario (0° e 5°), si è mostrata sufficiente, in quanto le differenze tra i risultati monitorati si sono rivelate limitate. Per i casi che hanno presentato un flusso instazionario (20°, 40°, 60°) invece i risultati ottenuti sono dipendenti dal tipo di mesh utilizzata, rendendo necessario l'utilizzo di griglie più fini per svolgere un'analisi più accurata.

L'utilizzo di *cfMesh* si è rivelato molto funzionale per il problema che si è studiato, in quanto è stato possibile generare mesh direttamente *2D*, inoltre *cfMesh* si è mostrato un *meshatore* facilmente controllabile e preciso poichè ha costruito griglie che hanno portato a calcolare una soluzione accettabile.

Anche per il caso dinamico avente come parametri l'inverso della rigidezza della molla pari a 0.35 e densità adimensionale dell'ala pari a 3.5 si è concluso dall'analisi di grid dependency che sussiste una sufficiente indipendenza dei risultati al variare della mesh utilizzate, permettendo di utilizzare la mesh b_2 per validare i risultati trovati. A livello di risultati è importante sottolineare che il caso che utilizza i parametri $\rho_w^* = 3.5$ e $1/k^* = 0.35$ coglie praticamente lo stesso moto oscillante verticalmente valutato da Overture. La configurazione con $\rho_w^* = 20$ e $1/k^* = 0.1$ mostra, come rilevato da Overture, un comportamento sotto-critico; solo con un opportuno disturbo iniziale si è innescato un fenomeno instabile autosostenuto controllato simile a quello rilevato da Overture.

I futuri sviluppi, sempre nell'ambito CFD, possono includere due differenti strade. La prima consiste nell'approfondire lo studio di alcuni aspetti del sistema idealizzato trattato in questa tesi, quali ad esempio il carattere

delle instabilità. La seconda strada consiste nell'avvicinare la simulazione alla realtà sperimentale, adottando per esempio la legge di deformazione completa non lineare degli elastomeri oppure studiando il dispositivo in tre dimensioni, fino ad arrivare alla riproduzione dell'esperimento in galleria del vento.

Appendice A

Schemi numerici

Quest'appendice riporta il contenuto dei file di OpenFOAM in cui si specificano gli schemi numerici ed i metodi di soluzione utilizzati nelle simulazioni.

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.x                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      fvSchemes;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

ddtSchemes
{
    default Euler;
}

gradSchemes
{
    default         cellLimited Gauss linear 1;
    grad(p)         cellLimited Gauss linear 1;
    grad(U)         cellLimited Gauss linear 1;
}

divSchemes
{
    default         none;
    div(phi,U)      Gauss linearUpwind grad(U);
    div((nuEff*dev(T(grad(U))))) Gauss linear;
}

laplacianSchemes
{
    default         Gauss linear limited corrected 0.5;
```

```
}

interpolationSchemes
{
    default         linear;
}

snGradSchemes
{
    default         corrected;
}

fluxRequired
{
    default         no;
    pcorr           ;
    p;
}

// ************************************************************************* //


/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.x                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

solvers
{
    pcorr
    {
        solver           GAMG;
        tolerance        1e-05;
        relTol           0;
        smoother         GaussSeidel;
        nPreSweeps       0;
        nPostSweeps      2;
        cacheAgglomeration true;
        agglomerator     faceAreaPair;
        nCellsInCoarsestLevel 100;
        mergeLevels      1;
    }

    p
    {
        $pcorr
        tolerance        1e-6;
        relTol           0;
    }

    pFinal
```

```
    {
        $p;
        tolerance        1e-7;
        relTol           0;
    }

    "(U|k|omega)"
    {
        solver          smoothSolver;
        smoother        symGaussSeidel;
        tolerance       1e-06;
        relTol          0;
    }

    "(U|k|omega)Final"
    {
        $U;
        tolerance       1e-06;
        relTol          0;
    }

    cellDisplacement
    {
        solver          GAMG;
        tolerance       1e-5;
        relTol          0;
        smoother        GaussSeidel;
        cacheAgglomeration true;
        nCellsInCoarsestLevel 10;
        agglomerator    faceAreaPair;
        mergeLevels     1;
    }
}

PIMPLE
{
    correctPhi          yes;
    nOuterCorrectors    2;
    nCorrectors         1;
    nNonOrthogonalCorrectors 2;
}

relaxationFactors
{
    fields
    {
        p               0.3;
    }
    equations
    {
        "(U|k|omega)"   0.7;
        "(U|k|omega)Final" 1.0;
    }
}

cache
{
    grad(U);
}

// ************************************************************************* //
```

Bibliografia

[1] *SALOME Tutorial, User's Guide*, 2013.

[2] *OpenFOAM. The Open Source CFD Toolbox. User Guide*, 2014.

[3] Edoardo Alinovi. Un metodo originale ed innovativo per l'energy harvesting. Tesi di laurea in ingegneria meccanica,Università di Genova., 2011.

[4] J. D. Anderson Jr. *Fundamentals of aerodynamics.* McGraw-Hill Education, 1985.

[5] A. Bartolazzi. *Le energie rinnovabili.* Hoepli editore, 2005.

[6] C. Boragno, R. Festa, and A. Mazzino. Elastically bounded flapping wing for energy harvesting. *Applied Physics Letters*, 100(25):253906, 2012.

[7] Y. A. Cengel and J. M. Cimbala. *Fluid Mechanics*, volume 1. McGraw-Hill Education, 2006.

[8] A. R. Collar. The first fifty years of aeroelasticity. *Aerospace*, 5(2):12–20, 1978.

[9] P. Douglas. Thermoelectric energy harvesting. *Article, School of Engineering, University of Glasgow*, 2014.

[10] J. H. Ferziger and M. Perić. *Computational methods for fluid dynamics*, volume 3. Springer Berlin, 2002.

[11] D. C. Gelvin, L. D. Girod, W. J Kaiser, W. M. Merrill, F. Newberg, G. J. Pottie, A. I. Sipos, and Sandeep Vardhan. Method for collecting data using compact internetworked wireless integrated network sensors (wins), May 11 2004. US Patent 6,735,630.

[12] M. Genova, F. Bregani, and P. Motta. Conversione diretta di energia termica in energia elettrica: stato dell'arte dei generatori termoelettrici ad effetto seebeck. Technical report. 2001, Rapporto Cesi.

[13] J. Guerrero. Course notes from a openfoam course. www.dicat.unige.it/guerrero/.

[14] A. Jackson. A comprehensive tour of snappyhexmesh. In *7th OpenFOAM Workshop*, 2012.

[15] F. Juretic. *cfMesh v1.0. User Guide*, 2014.

[16] M. Kuron. Three criteria for assessing cfd convergence. *www.engineering.com/DesignSoftware/DesignSoftwareArticles/ArticleID/9296/3-Criteria-for-Assessing-CFD-Convergence.aspx*.

[17] L. Lecce. Materiali piezoelettrici e loro applicazioni. 2002, Università di Lecce, Appunti del corso di materiali.

[18] F Lionetto, A Licciulli, F Montagna, and A Maffezzoli. Piezoceramici: guida introduttiva al loro utilizzo. *Journal Stress*, 1:2.

[19] L. Marrè Brunenghi. A flapping wing harvesting mechanism. Tesi di laurea in ingegneria meccanica,Università di Genova, 2014.

[20] P. D. Mitcheson, T. C. Green, E. M. Yeatman, and A. S. Holmes. Architectures for vibration-driven micropower generators. *Journal of Microelectromechanical Systems,*, 13(3):429–440, 2004.

[21] A Orchini, A Mazzino, J Guerrero, R Festa, and C Boragno. Flapping states of an elastically anchored plate in a uniform flow with applications to energy harvesting by fluid-structure interaction. *Physics of Fluids (1994-present)*, 25(9):097105, 2013.

[22] S. Patankar. *Numerical heat transfer and fluid flow*. CRC Press, 1980.

[23] S. Salvadori and F. Martelli. Fluidodinamica delle macchine. 2013, Università di Firenze, appunti del corso di macchine.

[24] H. K. Versteeg and W. Malalasekera. *An introduction to computational fluid dynamics: the finite volume method*. Pearson Education, 2007.

[25] F. Yildiz. Architectures for vibration-driven micropower generators. *Digital Library and Archives of the Virginia Tech University Libraries*, 2009.