НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Теплоенергетичний факультет

Кафедра автоматизації проектування енергетичних процесів і систем

"На правах рукопису"
УДК_534.2.231____

«До захисту допущено»
Завідувач кафедри
_____ О.В. Коваль
(підпис) (ініціали, прізвище)
“___”_____2018р.

Магістерська дисертація

зі спеціальності - 121 Інженерія програмного забезпечення
за спеціалізацією - Програмне забезпечення розподілених систем
на тему: «Комп'ютерне моделювання процесу випромінювання звуку при русі вісесиметричних тіл в морському середовищі»

Виконав: студент _6__курсу, групи_ТВ-71мп__
___Карпенко Дмитро Ігорович_____ _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник _д. ф.-м. н, проф. Гуржій О. А._____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Рецензент___пров.н.с., д.т.н. Воскобійник В. А._____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ - 2018

Національний технічний університет України
“ Київський політехнічний інститут ім. Ігоря Сікорського”

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти другий, магістерський

зі спеціальності - 121 Інженерія програмного забезпечення
за спеціалізацією - Програмне забезпечення розподілених систем

ЗАТВЕРДЖУЮ
Завідувач кафедри
Коваль О.В. _____
(прізвище, ініціали) (підпис)
«____» _____ 2018р.

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ ДИСЕРТАЦІЮ СТУДЕНТУ

_____ Карпенку Дмитру Ігоровичу
(прізвище, ім’я, по батькові)

1. Тема дисертації __ Комп'ютерне моделювання процесу випромінювання звуку при русі вісесиметричних тіл в морському середовищі_____
Науковий керівник ___Гуржій Олександр Андрійович д. ф.-м. н., проф._____
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету №___ від “___”_____20___ року
2. Строк подання студентом дисертації _____
3. Об’єкт дослідження _інформаційні технології моделювання процесу випромінювання звуку_____
4. Предмет дослідження __інформаційні технології дослідження поведінки вісесиметричних тіл в морському середовищі_____
5. Перелік питань, які потрібно розробити_____
1) проаналізувати особливості процесів випромінювання звуку в необмежених середовищах;_____
2) проаналізувати використання інженерних та програмних рішень для підвищення точності досліджень;_____
3) розробити алгоритмічну модель для розв’язання задачі розповсюдження звуку у необмеженому середовищі;_____

4) розробити програмне забезпечення для побудови траєкторії руху вихорових кілець, моделюванні тиску, спектру звукового поля, та побудови кореляційної функції.

6. Орієнтований перелік ілюстративного матеріалу

1) Математична модель

2) Етапи роботи з програмою

3) Функції програмного забезпечення

4) Структура програмного забезпечення

5) Інтерфейс

7. Орієнтований перелік публікацій

1) Карпенко Д.І., Гуржій О.А. **"** Комп'ютерне моделювання процесу випромінювання звуку при русі вісесиметричних тіл в морському середовищі**"**

2) Карпенко Д. І., Гуржій О.А. "Система моделювання випромінювання звуку вісесиметричними вихоровими структурами."

8. Дата видачі завдання «_29_»__вересня____ 2017 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання магістерської дисертації	Строки виконання етапів магістерської дисертації	Примітка
1	Отримання завдання	29.09.17р.	
2	Збір інформації	09.10.17р. – 25.01.18р.	
3	Аналіз вимог завдання, вибір методів і засобів розв'язання поставленої задачі	26.01.18р. – 05.09.18р.	
4	Підготовка публікацій	19.07.18р., 02.09.18р.	
5	Підготовка доповідей на конференції	15.08.18р., 20.10.18р.	
6	Підготовка дисертації	03.07.18р. – 07.12.18р.	
7	Розробка програмного продукту	10.01.18р. – 09.10.18р.	
8	Захист програмного продукту	24.10.18р.	
9	Передзахист	28.11.18р.	
10	Захист	12.18р.	

Студент _____ __Карпенко Д.І._____
(підпис) (прізвище та ініціали)

Науковий керівник _____ ___Гуржій О. А._____
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Структура й обсяг дипломної роботи

Магістерська дисертація складається зі вступу, 5 розділів, висновку, переліку посилань з 37 найменувань, 2 додатків і містить 20 рисунків, 23 таблиці. Повний обсяг магістерської дисертації складає 88 сторінок, з яких перелік посилань займає 4 сторінок, додатки – 9 сторінок.

Актуальність теми. Дану технологію можна використовувати у різних сферах промисловості. Наприклад, у риболовстві, за допомогою дослідження траєкторії розповсюдження звукових хвиль у воді, можна дослідити в яких районах багато риби. Також технологію можна використати у військовій справі. За допомогою подальших досліджень можна ідентифікувати підводне судно, яке випромінює звукові хвилі внаслідок свого переміщення, і дізнатись його місцезнаходження. За допомогою побудови кореляційної функції можна буде визначити характер розповсюдження звуку, що може допомогти зробити висновок про тіл, що випромінює звук.

Мета дослідження полягає в створенні програмного продукту, направленого на моделювання процесу випромінювання звуку, який генерується взаємодією вісесиметричних вихорових кілець у морському середовищі.

За допомогою програмного продукту можна досліджувати траєкторію руху тіла у морському середовищі, тиск та спектр звукового поля, а також аналіз розповсюджуваного руху за допомогою кореляційної функції.

Призначення програмного забезпечення полягає в наданні можливості користувачу побудувати траєкторію руху кілець протягом певного часу в ідеальній необмеженій в часі рідині. Іншим призначенням програмного застосунку являється моделювання генерованого звукового тиску, а також моделювання модулю його спектра і побудова кореляційної фукнції.

Для досягнення поставленої задачі були сформульовані наступні **завдання дослідження**, що визначили логіку дослідження та його структуру:

- проаналізувати особливості розповсюдження звуку від локалізованих вихорових структур;

- проаналізувати використання інженерних та програмних рішень для підвищення точності досліджень;
- провести аналіз розповсюдженого звуку за допомогою кореляційної функції;
- розробити програмне забезпечення для дослідження розповсюдження звуку в морському середовищі.

Об'єктом дослідження є комп'ютерні інформаційні технології моделювання процесу випромінювання звуку.

Предметом дослідження є комп'ютерні інформаційні технології дослідження поведінки вісесиметричних тіл в морському середовищі.

Методи дослідження. Розв'язання поставлених задач виконувались з використанням наступних методів:

- метод Рунге-Кутта для дослідження траєкторії руху вісесиметричних тіл;
- кореляційна функція для аналізу розповсюджування звуку.

Наукова новизна одержаних результатів. Найбільш суттєвими науковими результатами магістерської дисертації є:

- удосконалено спосіб розрахунку траєкторії руху локалізованих вихорових структур, зведення до задачі Коші з використанням еліптичних інтегралів першого та другого порядку. Обчислення коефіцієнту кореляції для побудови кореляційної функції і аналізу розповсюдження звуку;
- моделювання траєкторії руху вихорових кілець, аналіз тиску та спектру звукового поля, а також дослідження процесів у яких беруть участь від двох до п'яти кілець.

Практичне значення одержаних результатів роботи полягає в розробленому пакеті прикладних програм, направлених на дослідження розповсюдження звуку локалізованими вихоровими структурами в морському середовищі.

Ключові слова. ВИХОРОВІ СТРУКТУРИ, ТРАЄКТОРІЯ РУХУ, КОРЕЛЯЦІЙНА ФУНКЦІЯ, ЗВУКОВЕ ПОЛЕ.

ABSTRACT

Structure and volume of the dissertation

Master's dissertation consists of introduction, 5 sections, conclusion, list of sources of 37 items, 3 appendixes and has 20 images, 23 tables. Whole dissertation volume is 88 pages, out of which list of sources takes 4 pages, appendixes - 9 pages.

Significance of the topic. This technology can be used in various industries. For example, in fishing, by studying the trajectory of propagation of sound waves in water, one can investigate in which areas a lot of fish. Also, technology can be used in military affairs. Further research can identify a submarine that emits sound waves due to its displacement and find out its location. By constructing a correlation function, one can determine the nature of the propagation of the sound, which can help to draw conclusions about the bodies that emit sound.

The **objective** of the topic is to create a software product aimed at simulating the sound emission process, which is generated by the interaction of axisymmetric vortex rings in the marine environment.

With the help of the software product, you can study the trajectory of body motion in the marine environment, the pressure and spectrum of the sound field, as well as the analysis of distributed motion with the help of correlation function.

The purpose of the software is to enable the user to build a trajectory of motion of the rings for a certain time in a perfect, unlimited fluid in time. Another purpose of the software application is simulation of generated sound pressure, as well as modeling its spectrum module and building a correlation function.

In order to achieve the objective next **goals** were defined:

- to analyze the peculiarities of the propagation of sound from localized vortex structures;
- to analyze the use of engineering and software solutions to improve the accuracy of research;
- to analyze the distributed sound using the correlation function;
- to develop software for the study of sound propagation in the marine environment.

The **object** of the exploration is computer information technology simulation process of radiation sound.

The **subject** of the exploration is computer information technologies for the study of the behavior of axisymmetric bodies in the marine environment.

Exploration methods. Solution of defined goals was made with usage of next methods:

- the Runge-Kutta method for studying the trajectory of motion of axesymmetric bodies;
- correlation function for analysis of sound distribution.

Scientific novelty of the results. The most valuable scientific results of masters dissertation are:

- the method of calculation of the trajectory of motion of localized vortex structures, reduction to the Cauchy problem using elliptic integrals of the first and second order is improved. Calculation of the correlation coefficient for constructing a correlation function and analysis of sound propagation;
- simulation of the trajectory of the motion of vortex rings, analysis of pressure and spectrum of the sound field, as well as research of processes involving from two to five rings.

Practical significance of the results of work in the developed package of applied programs aimed at studying the propagation of sound by localized vortex structures in the marine environment.

Key words. EXCHANGE STRUCTURES, TRACKER MOVEMENT, CORRELATION FUNCTION, SOUND FIELD

ЗМІСТ

ВСТУП

Стрімкий розвиток інноваційних технологій, розвиток науки та техніки сьогодні супроводжується постійним впровадженням нових технологічних процесів у промисловості та побуті. До числа таких задач відноситься задача про випромінювання звуку великомасштабними вихоровими структурами у морському середовищі. Важливим аспектом цієї проблеми являється дослідження траєкторії руку кілець, також дослідження генерованого звукового тиску, дослідження модулю його спектра і коефіцієнту кореляційної функції [1].

На сьогодні не існує повноцінних засобів призначених для моделювання траєкторії руху різної кількості кілець, моделювання звукового тиску і спектру, а також за допомогою кореляції підвищення точністі досліджень та висновків щодо розповсюдження звуку у морському середовищі. Це зумовлює актуальність розробки нових програмних засобів для моніторингу звукового поля та дослідження поведінки різної кількості вихорових структур.

Розроблено програмний продукт, який моделює траєкторію руху різної кількості кільц. Програма також моделює генерований звуковий тиск, а також моделювати модуль його спектра. Ці дослідження покажуть момент, коли взаємна енергія кілець максимальна. Це пояснюється тим, що при взаємодії переднє вихорове кільце, збільшуючи свою енергію, розширюється і призупиняється, в той час як заднє кільце, зменшуючи свою енергію, звужується і розганяється, проскакуючи через переднє кільце. Така “чехарда” в рамках моделі ідеальної рідини продовжується нескінченно довго без участі відводу енергії на випромінювання. Одним з аспектів дослідження був аналіз випромінюваного звуку. Програмний продукт дає можливість користувачу побудувати кореляційну функцію, яка дасть відповідь на питання який саме звук випромінюється, чи це шум, чи конкретний звук від джерела [2].

Метою дослідження магістерської дисертації є розробка засобів моделювання звукувого поля, генерація траєкторії руху вихорових кілець, моделювання спектру, моделювання кореляційної функції та спектру.

Дану технологію можна використовувати у різних сферах промисловості. Наприклад, у риболовстві, за допомогою дослідження траєкторії розповсюдження звукових хвиль у воді, можна дослідити в яких районах багато риби. Також технологію можна використати у військовій справі. За допомогою досліджень можна ідентифікувати підводне судно, яке випромінює звукові хвилі внаслідок свого переміщення, і дізнатись його місцезнаходження.

Магістерська дисертація складається з чотирьох розділів. У першому розділі наведено огляд існуючих рішень, який може бути врахована при розробці програмного забезпечення.

Другий розділ присвячено докладному опису алгоритмічної моделі запропонованого рішення задачі розповсюдження звуку локалізованими вихоровими структурами у морському середовищі.

У третьому розділі описано структуру та функціональність розробленої програмної системи моделювання розповсюдження звуку, куди входить моделювання траєкторії польоту вихорових кілець, моделювання тиску та спектру звукового поля і побудова кореляційної функції.

Четвертий розділ присвячено аналізу можливостей просування розробленого продукту у якості стартап-проекту [3].

В програмній реалізації для графічного відображення було вирішено використовувати програмну бібліотеку Zed Graph, яка призначена для побудови графіків та візуалізації графічних компонентів.

Середовищем реалізації програмного продукту було обрано Microsoft Visual Studio, а мовою програмування — C#.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

ЕОМ	- електронно обчислювальна машина
БД	- база даних
ФРЕЙМВОРК	- інфраструктура програмних рішень, що полегшує розробку складних систем
ООП	- об’ектно – орієнтоване програмування
ОС	- операційна система
НТУУ “КПІ”	- Національний технічний університет України "Київський політехнічний інститут"
ІС	- інформаційна система

1. ПОСТАНОВКА ЗАВДАННЯ МАГІСТЕРСЬКОЇ РОБОТИ

Задача про випромінювання звуку великомасштабними вихровими структурами належить до найбільш цікавих проблем гідроакустики. Дослідження Лайтхілла (2.1) є основоположною роботою присвяченою проблемі випромінювання звуку нестаціонарними джерелами. У ході дослідження була сформована аналогія між задачею про генерацію широкополосного шуму і класичною задачею про випромінюванні звуку [4].

1.1. Геометрія задачі

Роздивимось в циліндричній системі координат поодиноке ізольоване вихорове кільце, що володіє осьовою симетрією по окружній координаті (рисунок 1.1). В цьому випадку поле завихрення має одну компоненту вектора $\omega = (0, \omega, 0)$, а поле швидкості — дві: $U = (U_r, 0, U_z)$.

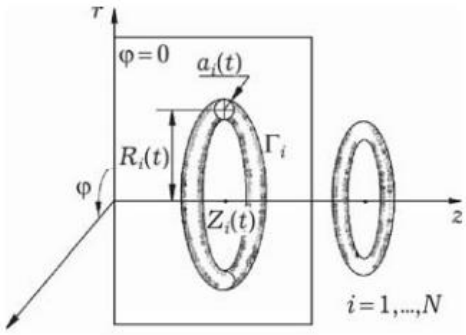


Рисунок 1.1 — Геометрія задачі

Тоді векторне рівняння Гельмгольця, яке отримано для завихрення $\frac{\partial \omega}{\partial t} + (U \cdot V)\omega = (\omega \cdot V)U$ зводиться до одного скалярного рівняння, пов'язуючи їх вуглову

компоненту вектора завихрення з осьовою і радикальною компонентами шкидкості:

$$\frac{\partial}{\partial t}(\frac{\omega}{r})+U_r\frac{\partial}{\partial r}(\frac{\omega}{r})+U_r\frac{\partial}{\partial z}(\frac{\omega}{r})=0. \quad (2.1)$$

Рівняння 2.1 має часткове рішення

$$\frac{\omega}{r}=const, \quad (2.2)$$

що відповідає так званому випадку рівномірного завихрення в середині вихорового кільця. Відзначимо, що рішення 2.2 не залежить від часу, тому воно справедливе як для стаціонарного, так і для нестаціонарного руху вихорів [5].

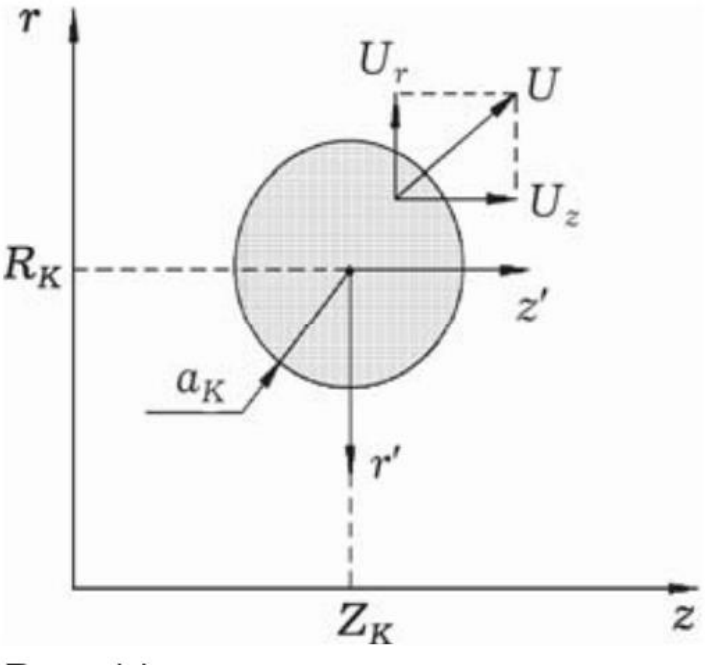


Рис 1.2 — Поперечне січення вихорового кільця

Оскільки рух рідини поза кільцем є потенційним, необхідно, щоб рішення 2.2 рівнялось нулю поза областю, зайнятою вихорем [6].

Поперечне січення вихорового кільця зображено на рисунку 1.2. Тут R це радіус вихорового кільця, Z осьове положення, а товщина кільця, Г інтенсивність. Надалі в роботі будуть проаналізовані різні типи взаємодії з різною кількістю кілець (від двох до шести кілець прі різних початкових умовах. Надалі будемо використовувати ці обозначення, коли будемо розглядати характерні випадки взаємодії двох кілець з різними початковими умовами.

1.2. Випромінювання звуку

Розглянемо задачу про випромінювання звуку системою тонких вісесиметричних вихрових кілець в ідеальній безмежній рідині. Досліджено, що дальнє звукове поле, яке випромінюється вихоровими структурами, визначається виразом

$$p(\mathbf{x},t)=\frac{\rho_0}{c^2}\frac{n_i n_j}{|\mathbf{x}|}\frac{d^3}{dt^3}Q_{ij}\left(t-\frac{|\mathbf{x}|}{c}\right),$$

де

$$Q_{ij}=\frac{1}{12\pi}\int_{V_w} y_i(\mathbf{y}*\mathbf{w}(t))_j\,d\mathbf{y}.$$

Відомо, що процес звуковипромінювання вихоровими структурами визначається динамікою їх взаємодії. Основним співвідношенням, що описує цей процес є система диференційних рівнянь, яка складається з наступних рівнянь [7]:

$$\dot{R}_i=\sum_{j=1}^{N}\frac{\chi_i(Z_j-Z_i)}{2\pi R_i R\max_{ij}}[K(k_{ij})-E(k_{ij})-\frac{2R_iR_j}{R^2\min_{ij}}E(k_{ij})],$$

$$\dot{Z}_i=\frac{\chi_i}{4\pi R_i}(\ln\frac{8R_i}{a_i}-\frac{1}{4})+\sum_{j=1}^{N}\frac{\chi_i}{2\pi R\max_{ij}}[K(k_{ij})-E(k_{ij})-\frac{2R_j(R_j-R_i)}{R^2\min_{ij}}E(k_{ij})].$$

Взаємодія системи N тонких вісесиметричних кілець інтенсивності χ_i, розташовані в точках (R_i, Z_i) циліндричної системи координат, яка співпадає з віссю симетрії, описується гамільтоновою системою рівнянь:

$$\dot{Z}=\frac{\chi_i}{4\pi R_i}(\ln\frac{8R_i}{a_i}-\frac{1}{4})+\frac{1}{\chi_i R_i}\frac{\partial U}{\partial R_i},\qquad a_i^2 R_i=const,\ i=1,\ldots N$$

де

$$U=\sum_{i=1}^{N}\sum_{j>1}^{N}\frac{\chi_i\chi_j}{2\pi}\sqrt{R_iR_j}\left[(\frac{2}{k_{ij}}-k_{ij})\mathbf{K}(k_{ij})-\frac{2}{k_{ij}}\mathbf{E}(k_{ij})\right],$$

$$k_{ij}^{\ 2}=\frac{4R_iR_j}{(Z_i-Z_j)^2+(R_i+R_j)^2},\quad n_i^0<<1.$$

Тут **K**(k) и **E**(k) — повні еліптичні інтеграли першого і другого роду [3].

Для задачі Коші було сформовано наступні початкові умови: $R_i(t) = R_i^0$, $Z_i(t) = Z_i^0$, $a_i(0) = n_i^0 R_i^0$.

Для аналізу особливості звукового поля будується спектр звукового поля. Для цього застосовується алгоритм прямого перетворення Фур'є

$$\left|Sp(w)\right| = \frac{1}{T}\left|\int_0^\infty Q(t)e^{jwt}dt\right|.$$

Розглянемо характерний випадок періодичної взаємодії двох однакових вихрових кілець ($R_1^0 = R_1^0 = 1.0$, $Z_1^0 = 0.0$, $Z_2^0 = 1.0$, $n_2^0 = n_1^0 = 0.01$). Два вихори рухаються у напрямку позитивних значень координат на осі симетрії. Плином часу перший вихор прискорюється і зменшується, а другий вихор сповільнюється і розширюється. В деякий момент часу перший вихор проскакує крізь другий. Далі процес проскакування періодично повторюється. Показано, що вихори генерують максимальне звукове поле в момент проходження одного з кілець скрізь іншого, у момент, коли їх взаємний вплив максимальний. На рисунку 2.3 зображені траєкторії руху вихорів для цього випадку.

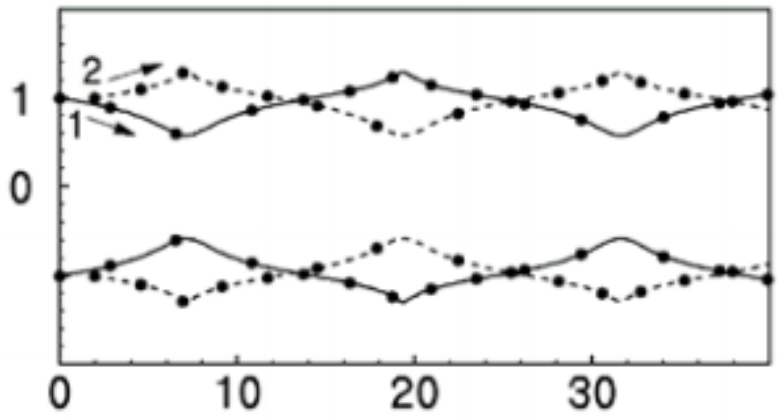


Рисунок 2.2 — Періодична взаємодія двух однакових вихорових кілець

Досліджено, що генерований звуковий тиск набуває максимального значення під час проскакування кілець. Якщо в випромінюванні беруть участь три кільця, то можемо спостерігати, що друге і третє будуть рухатись періодично, а перше незалежно від них.

Подібне явище спостерігається і у випадку, коли участь беруть чотири і більше кілець. Але якщо підібрати початкові значення, то можна добитись проскакування усіх кілець і максимального значення звукового тиску [8].

Ззовні кільця состовляючі швидкості мають бути неперервними. Разом з цим виникає класична задача про склейку потенціального і вихорового рухів. Існує точне співвідношення поперечного січення вихорового кільця, яка залежить від закону руху області завихреності і створює складну задачу.

Надалі за допомогою обчислення кореляційної функції можна буде проаналізувати звук, який розповсюджується від джерела. Цей результат можна отримати використовуючи значення, які були отримані при дослідженні звукового поля, а саме обчислюванні звукового тиску. Отримаючи графік кореляційної функції можна проаналізувати чи звук надходить від локалізованого джерела, чи це просто шум. Якщо помітна періодичність, то можемо судити що джерело існує і воно локалізоване, а значить можна продовжувати дослідження. Якщо ж графік стрімко і хаотично наблизиться до нуля, то можемо констатувати, що це шум, а значить не доцільно продовжити дослідження.

Висновки до розділу 1

У першому розділі було:

- проаналізовано сучасні методи моделювання звукового поля, яке створюється системою тонких вихорових структур. Дослідження показали, що дотепер не проводились дослідження стосовно моделювання траєкторії руху вихорових кілець, обчислення звукового тиску та спектру;
- аналіз досліджень показав, що можливо дослідити характерний випадок для двох однакових вихорових структур.

2. МОДЕЛЮВАННЯ ЗВУКОВОГО ПОЛЯ, УТВОРЕНОГО СИСТЕМОЮ ТОНКИХ ВИХОРОВИХ КІЛЕЦЬ

Вихорові кільця у рідині можна легко зімітувати як у домашніх умовах, так і у лабораторних умовах. Прикладом є рисунку 2.1, на якому видно, що дельфін під водою створює вихорове кільце [9].

Рисунок 2.1 — Вихорове кільце, створене дельфіном

Цей процес утворюється за допомогою крапель підфарбованої рідини, які падають на звичайну вільну поверхню рідини. Згадаємо першу експериментальну роботу, яка була направлена на детальне вивчення виникнення завихреності в спокійній рідині. Автором цієї роботи є основник Массачусетського технологічного інституту США Роредсу. Він використовував круглу піпетку, закріпивши на кронштейні і великий циліндричний сосуд, який був наповнений чистою водою. Зазначимо, що його робота вийшла одночасно з мемуарами Гельмгольца. Слід за ними свої роботи опублікував Томілсон. Вони були присвячені процесам формування вихорових кілець в залежності від фізико-хімічних параметрів рідин каплі і резервуара. Більш детальні експерименти по формуванню вихорових кілець шляхом

виштовхуванням фіксованого об’єму рідини з отворами були проведені Томсоном та Невілом.

Виникнення вихорових кілець за допомогою струй описано в роботі Реуша, в якій більше уваги приділено фізичному поясненню утворення кілець при імпульсній інжекції фіксованого об’єму рідині. Трохи пізніше Тейт з Едінбургського університету (Шотландія) продемонстрував простий спосіб генерації димових вихорових кілець у повітрі. Він використав коробку з кроговим отвором з одної сторони і резинові діафрагми з іншої сторони. Процес горіння сульфата магнія всередині коробки утворює білий дим, який після удару по резиновій діафрагмі утворює кільце , яке віддалюється від пристрою. Томсон, зустрівшись з Тейтов в середині січня в 1867 році в Едінбурзі побачив власним оком експеримент з димовими кільцями і був враженій експериментом.

Трохи пізніше Обербек детально описав різні ситуації виникнення вихорових кілець при витіканні імпульсних струй і їх еволюцій в залежності від початкових і граничних умов. Було показано, що на витіканні струй суттєво впливають сили тертя, завдяки яким видна чітка границя між спокійною рідиною і струєю, яка формує вихорове кільце [10].

2.1. Аналіз проблеми та опис задачі

Як було показано, основною роботою в динаміці вихорів є робота Гельмгольца, у якій в перший раз було поставлено закони та теореми для вихорового руху в рамках моделі ідеальної необмеженої рідини. Розглянуто умови при потенціальності зовнішніх сил і неможливості стискання однорідної рідини. Якщо разом взяти вісесиметричку задачу про рух системи вихорових кілець в ідеальній необмеженій рідині теореми Гельмгольца, то вона дозволить зробити деякі висновки про закономірності і важливі примітки про рух тіл. Спершу, вихорові кільця беруть участь у русі в рамках прийнятої моделі рідини стільки, скільки позволяє час. Потім, по другій теоремі Гельмгольца, спостерігається, що в процесі руху об’єм торообразної

вихорової сруктури залишається сталим. Третя теорема говорить нам, що значення інтернсивності при розразунку слід розраховувати по формулі [11]:

$$\Gamma = \int S(\omega * n)dS = const,$$

де n – це вектор, який перпендикулярний до площини поперечного січення кільця.

Якщо роздивлятис в циліндричній системі координат одиничне ізольоване вихорове кілце, яке в свою чергу має осьову симетрію по кругу. У цьому випадку поле завихренності має одну співставляючу вектора $\omega = (0, \omega, 0)$, а за цим поле видкості – дві: $U = (U_r, 0, U_z)$. Тоді векторне співвідношення Гельмгольца для завихрені зводиться до одного рівняння, скалярного типу. І воно зводиться з угловим компонентом вектора завихреності з осьовою і радикальною компонентами швидкості.

$$\frac{d}{dt}(\frac{\omega}{r}) + U_r \frac{d}{dr}(\frac{\omega}{r}) + Uz\frac{d}{dz}(\frac{\omega}{r}) = 0.$$

Рівняння має часткове рішення:

$$\frac{\omega}{r} = const.$$

Воно відповідає так званому випадку рівномірної завихреності, яка відбувається в середині вихорової структури, а саме вихоровому кільці. Слід сказати, що рішення не залежить ніяким чином від часу, тому це твердження є справедливим як для устоявшегось, так і для нестаціонарного руху вихорів.

Так як рух рідини поза межами вихорового кільця є більш потенціальни, необхідно що б рішення рівнялось нуля поза межами кільця.

Ззовні кільця состовляючі швидкості мають бути неперервними. Разом з цим виникає класична задача про склейку потенціального і вихорового рухів. Існує точне співвідношення поперечного січення вихорового кільця, яка залежить від закону руху області завихреності і створює складну задачу. Спрощенням вважається рішення, яке утворюється при вивченні тонких тороїдальних вихорових структурах. Форма поперечного січення у таких кільцях є колоподібною. Отже, рух вихора супроводжується перетворенням границі вихора при незмінній круговій формі

поперечного січення і того, як розділяються між собою завихреності у середині кільця.

2.2. Взаємодія двух вихорових кілець

Одинична вихорова структура в безграничній середі рухається, як правило, стаціонарно. Але, чи зможе брати участь в подібному процесі два вісесиметричних вихорових кільця, якийсь час було незрозуміло. Чимало робіт було зроблено, що б внести розуміння у це питання. У них були показані умови, за яких два вихорових кільця рухаються стаціонарно з однаковими осьовими швидкостями.

Якщо роздивлятись інтенсивності заданих кілець а також відносну товщину, у випадку руху в будь який момент часу повинні бути наступні умови

$$\dot{R}_1 = \dot{R}_2 = 0, Z_1 - Z_2 = 0.$$

Провівши аналіз, то отримані результати радіальних компонентів швидкості кілець, можемо зробити настпуні висновки, що радіальні швидкості дорівнюють нуля. Мова йде про випадки, коли вихорові кільця знаходяться на максимальному віддалені один від одного ($|Z_1 - Z_2| = \infty$). Даний випадок відноситься до ізольованих вихорових структур. Але радіальні швидкості будуть рівнятись нулю, коли структури будуть лежати в одній площині, ($Z_1 = Z_2$). Тоді умова дасть інший зв'язок між радіусами і інтенсивностями вихорових кілець.

Дивлячись на залежність інтенсивності другого кільця від радіусу для Γ_1=1.0 і R_1 = 1.0 при $n_1 = n_2$ = 0.01 показано на рисунку 2.2. Для любого R_2 є наступне значення Γ_2, за яким вихорові структури перебувають у стаціонарному русі. Для не тонких кільце значення Γ_2 є від'ємним завжди, а для тонких існує область значень більше нуля Γ_2 для малих значень радіусу другого вихорового кільця, R_2 < 0.6.

Для здійснення аналізу взаємодії вихорових структур слід застосувати для задачі значення інтернсивності Γ_1 і початкого радіусу R_1^0, для першого кільця. При чому Z_1^0 = 0.0, $n_1^0 = n_2^0$ = 0.01.

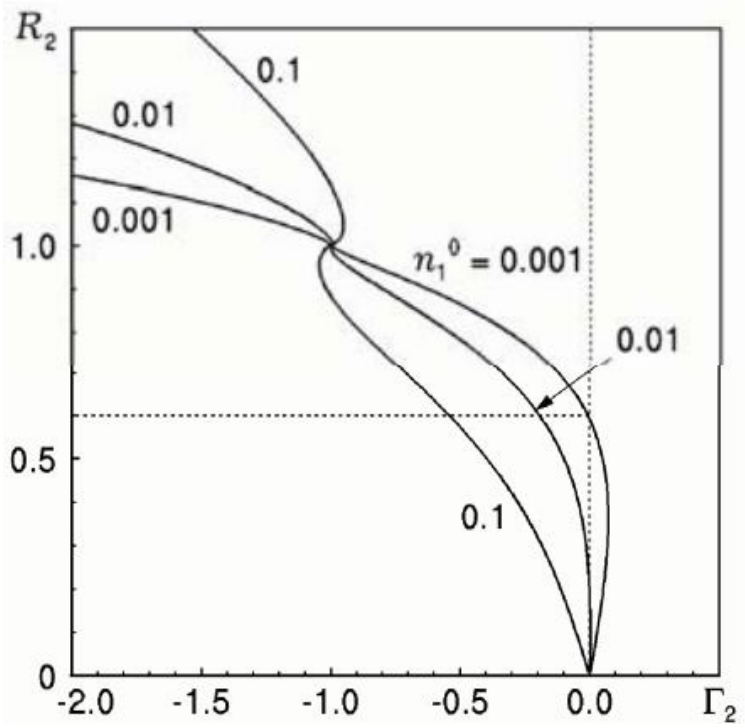


Рисунок 2.2 — Параметри стаціонарної взаємодії двох кілець

Відомо, що при участі двох ідентичних вихорових кілець, вони будуть брати участь в періодичній взаємодії. Ця взаємодія пояснюється проскауванням одного кільця скрізь інше при відсутності зміщення вздовж позитивної осі симетрії за рахунок самоіндукційної щвидкості. В літературі такий процес називається «чехардою» вихорових кілець.

В роботі зроблено припущення, що для визначення області допустимих початкових значень структур для періодичного процесу слід застосувати умову самоіндукційних швидкостей кілець на максимальному віддаленні один від іншого. Виявляється, що в цьому випадку кільця можуть удалитись в нескінченність, або ж почнуть рухатись один до одного, так би кажучи, зближатись. Після цього виникає взаємодія вихорових структур. Кільця можуть ставати одне на місце іншого, і в силу симетрії цієї задачі відносно $Z_1 - Z_2$, кільця будуть віддалятись знов на нескінченно взаємний шлях і після цього не зустрінуться ніколи.

Отже, для обчислення області можливих початкових значень для періодичної взаємодії кілець слід до умови саміндукційних швидкостей вихорів на нескінченній віддаленості одне від одного, додати два варіанти інваріанта руху.

На рисунку 2.3 показані області початкових значень іншого кільця для періодичної взаємодії при Γ_2 = 0.5, 1.0, 1.5 і $n_1^0 = n_2^0$ = 0.01. Як бачимо на рисунку, площі цих областей при рості значенні інтенсивності збільшуються і взаємна «чехарда» можлива при будь якому значенні інтенсивності Γ_2.

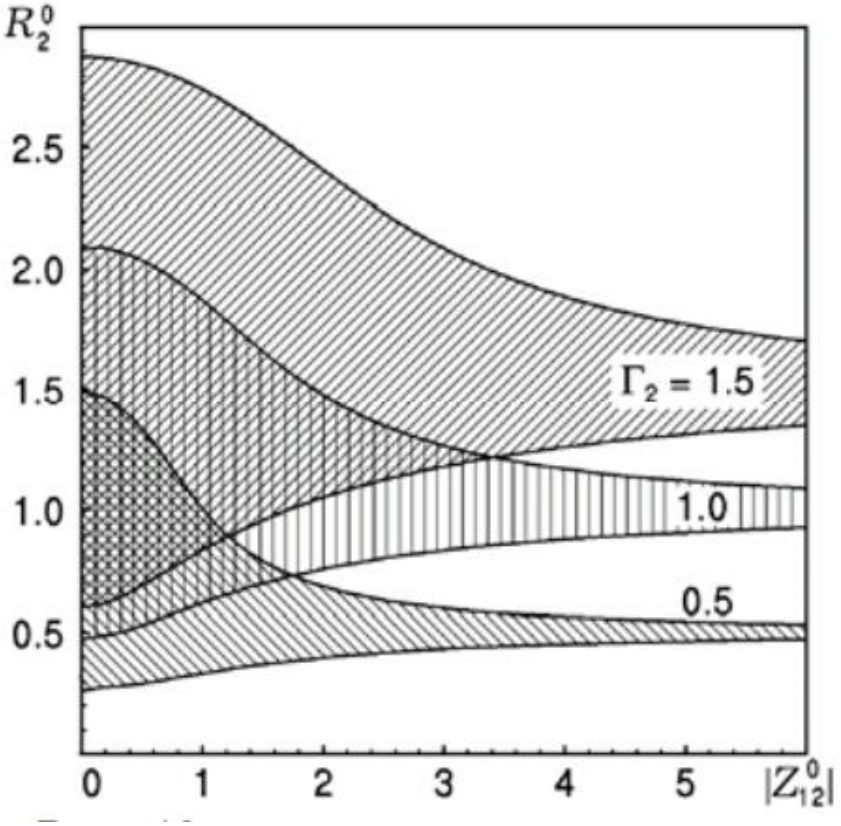


Рисунок 2.3 — Початкові параметри вихорів при періодичній взаємодії

Періодична взаємодія характеризується почерговим процесом проскакування одного вихорового кільця в середні іншого. На рисунку 2.4 представлені траєкторії руху вихорів для випадку Γ_2 = 1.0, R_2^0 = 1.0 і Z_2^0 = 2.0. Траєкторія першого $R_1(Z_1)$ на рисунку показана сплошною лінією, а траєкторія другого кільця $R_2(Z_2)$ намальована штрих пунктиром. Положення вихорів через рівні інтервали часу 5.0 секунд показані кругами. Стрілками а рисунку позначені напрямки руху при взаємодії вихорів. При взаємодії кільце, що іде попереду під впливом кільця, що поруч зменшується, цей вихор зменшує свій радіус і набирає швидкість. Після проскакування меншого вихора в середині того, що більше, кільце, яке вже зменшилось, знаходиться попереду більшого вихора. Процес прискорення заднього кільця і торможення переднього

періодично повторюється. Така «чехарда» вихорових кілець в рамках ідеальної рідини буде повторюватись необмежено довго [12].

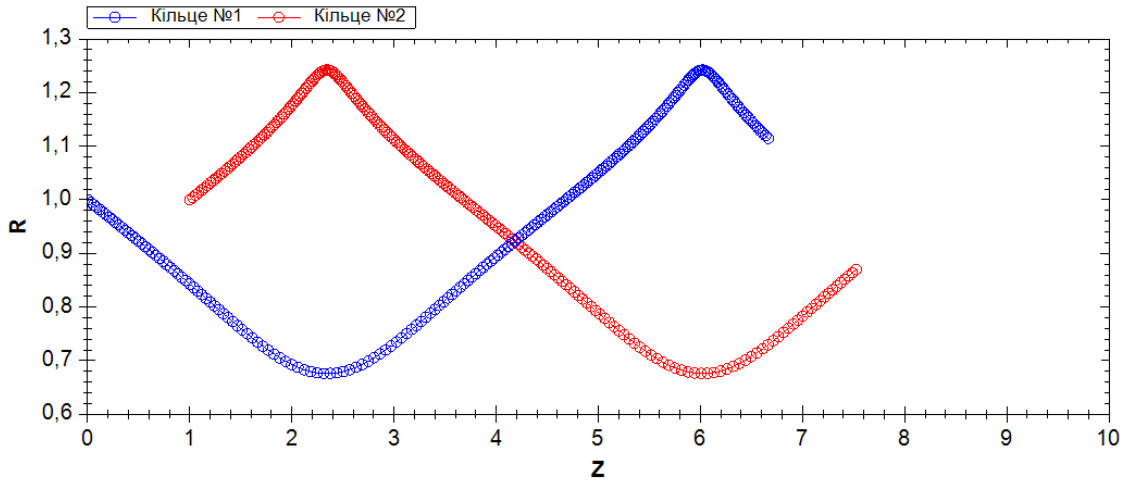


Рисунок 2.4 — Періодична взаємодія двох однакових кілець

У випадку, коли початкові значення не входять в область параметрів для періодичної взаємодії, то вихорові структури або взагалі не взаємодіють, або взаємодіють лиш раз, в залежності від значення осьових швидкостей вихорів. Якщо значення швидкості для вихорового кільця, що рухається попереду більше по відношенню до кільця, що йде позаду, то відносна дистанція між кільцями збільшиться з часом. Відзначимо, що взаємодія буде відсутня. Якщо переднє кільце має меншу швидкість, то кільце, що йде позаду наздожене переднє. Внаслідок цього виникне проскакування одного кільця скрізь інше, тобто заднє кільце проскоче в середині переднього. Потім же , в силу симетрії задачі тепер вже переднє кільце, яку буде на той чай мати більше осьову швидкість, віддалиться на нескінченність. На рисунку 2.5 зображений випадок, коли і відбувається така взаємодія для $\Gamma_2 = 0.5$, $Z_2^0 = 2.0$ та $R_2^0 = 1.0$ в позначеннях, які були прийняті раніше.

Надалі у роботі будуть розглядатись випадки, коли у процесі будуть приймати учать більша кількість кілець. При взаємодії трьох чи більше можна дослідити випадки, коли кільця будуть рухатись з якоюсь визначеною закономірністю. Але в більшості випадків ми будемо спостерігати хаотичне недосліджуєме просування кільця уздовж позитивної осі. З рисунків буде видно, що взаємодія не є такою періодичною.

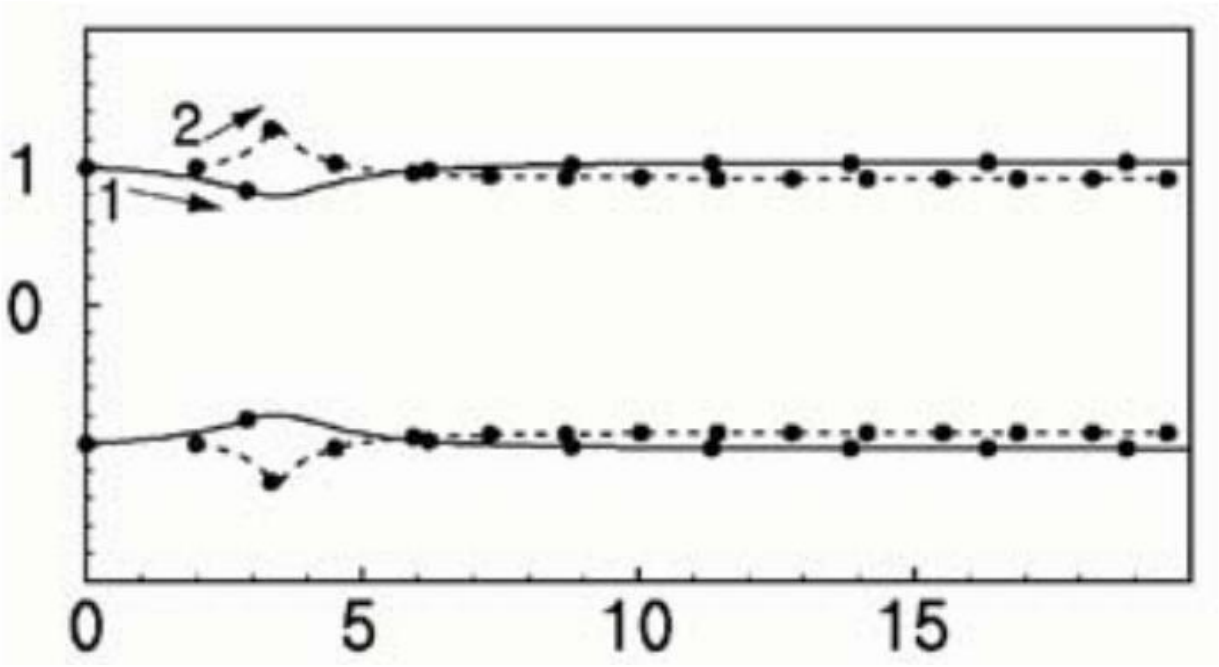


Рисунок 2.5 — Разова взаємодія двох кілець з інтенсивностями одного знаку

У випадку для систему, яка складається з двох вихорових структур з інтенсивністю протилежних значень можливі три типи взаємодії. Розглянемо перші два, які описуються зустрічним рухом двох вихорових кілець, і вони проскакують один крізь інше по ходу свого руху. Зазначимо, що кільця, які маються меншу початкову енергію є більш чуттєвими до зміненням у порівнянні з кільцями, які мають більшу енергію.

Характерним випадком проскакування меншого кільця у середині того, що більше зображено на рис 16 для Γ_2 = - 0.6, та R_2^0 = 1.0 і Z_2^0 = 2.0. Траєкторія першого вихора на рисунку показана суцільною лінією, а траєкторія другого – штрих пунктиром. Положення вихорів через рівні інтервали часу в 0.5 секунд нанесені на рисунку кружечками, а стрілки указуються напрям руху вихорів вздовж осі. На початку руху вихори рухаються один на зустріч другому. Вони рухаються з осьовими швидкостями, які є рівними само індукційної швидкості вихорових кілець. По мірі наближення починається взаємний вплив одного кільця на інше. Цей вплив призводить до того що швидкість руху падає, а радіуси кілець збільшується. У випадку, який ми розглядаємо, друге кільце міняє знак осьової швидкості і деякий час просувається в зворотному напрямку. Після того, як перше кільце проскакую крізь

друге радіальні швидкості обох вихорів міняють свій знак на протилежний. Це призводить до того, що взаємний вплив вихорів зменшується і обидва вихори віддаляються один від одного з само індукованими швидкостями, які направлені в протилежні сторони.

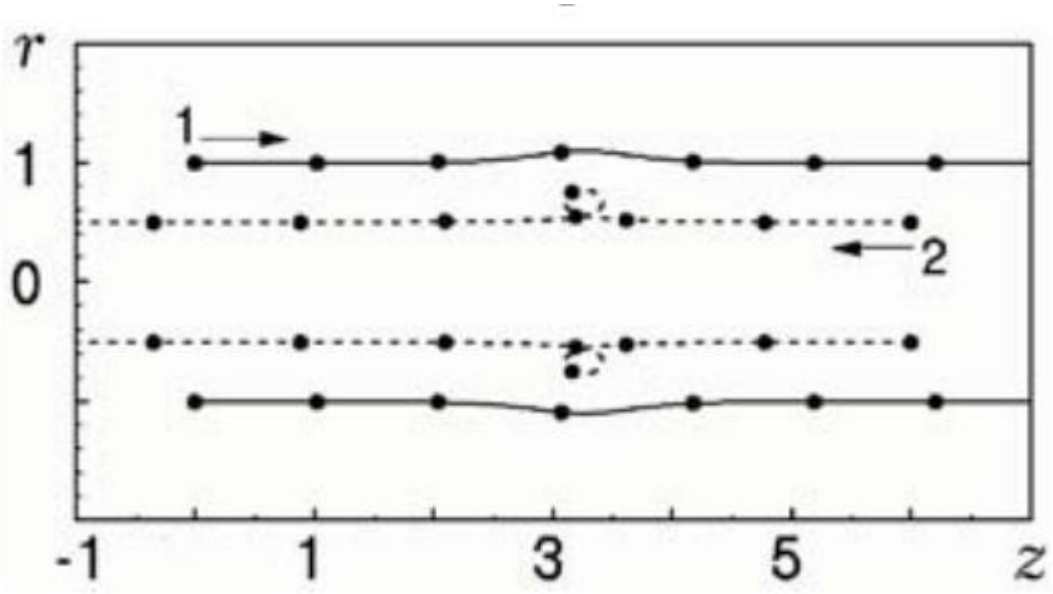


Рисунок 2.6 — Разове проскакування другого вихора всередині першого кільця

Зазначимо, що у випадку, коли другий вихор буде мати меншу інтенсивність, то можливий випадок взаємодії, при якому меньший вихор проскакує над більшим кільцем. На рисунку 2.6 зображено випадок для Γ_2 = - 0.6, та R_2^0 = 0.6 і Z_2^0 = 6.0. При зближенні вихорів другий виор, маючи меншу інтенсивність, підвержен більшим змінам в порівнянні з випадком, який розглядався до того. Другий вихор в результаті набуває більшу радіальну швидкість і збільшує свій радіус швидше, ніж перше кільце. Радіус меншого кільця стає більше ніж радіус першого. Виходячи з цього спочатку менше кільце проскакую над більшим. Потім, в силу симетрії задачі, друге кільце швидше змінює значення радіальної швидкості і асимптотично виходить на початкове значення радіусу. Взаємний вплив вихорів поступово зменшується і обидва вихори поступово відходять один від одного з самоіндукційними швидкостями. Направленими в різні сторони.

В роботі зроблено припущення, що для визначення області допустимих початкових значень структур для періодичного процесу слід застосувати умову

самоіндукційних швидкостей кілець на максимальному віддаленні один від іншого. Виявляється, що в цьому випадку кільця можуть удалитись в нескінченність, або ж почнуть рухатись один до одного, так би кажучи, зближатись.

Можна показати, що вид взаємодії вихорових кілець з різнмим значеннями інтенсивностей визначається значенням осьових швидкостей в момент, коли кільця знаходяться в одній площині ($Z_1 = Z_2$). Якщо кільця будуть мати радіуси меньше ніж їх радіуси відповідно для стаціонарного руху, різницею осьових швидкостей кілець $\dot{Z}_1 = \dot{Z}_2$ змінює свій знак. В результаті відбувається проскакування другого вихора скрізь перше кільце. Якщо в площині $Z_1 = Z_2$ кільця будуть мати великі радіуси, ніж при стаціонарній взаємодії, то відносні траєкторії повинні бути замкнутими. Даний випадок руху повинен супроводжуватись періодичним рухов в силу симетрії задачі.

Отже, якщо вихори асиметрично достигають площину $Z_1 = Z_2$ в окресності стаціонарного режиму, то радіальні швидкості вихорів не міняють свій знак. В цьому випадку має місце проскакування другого кільця над першим.

Висновки до розділу 2

У другому розділі було:

- визначено новітні методи моделювання поведінки вихорових кілець в морському середовищі;
- аналізовано спосіб моделювання траєкторії руху вихорових кілець у необмеженому середовищі;
- удосконалено спосіб обчислення тиску та спектру звукового поля;
- досліджено характер випромінюваного звуку, який розповсюджується на основі кореляційної функції.

3. ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

Основним середовищем розробки було середовище Microsoft Visual Studio. Для створення певних деталей інтерфейсу використовувався Adobe Photoshop CS6, що є програмою для створення та редагування графічних елементів. Окрім цього, попередній шаблон порталу також було спроектовано засобами Photoshop.

3.1 Середовище розробки Microsoft Visual Studio

Microsoft Visual Studio — це інтегроване середовище розробки (IDE) від Microsoft. Він використовується для розробки комп'ютерних програм, а також веб-сайтів, веб-програм, веб-сервісів та мобільних додатків. Visual Studio використовує платформи для розробки програмного забезпечення Microsoft, такі як Windows API, Windows Forms, Windows Presentation Foundation, Windows Store і Microsoft Silverlight. Він може створювати як власний код, так і керований код. Приклад середи зображений на рисунку 3.1 [13].

Продукт, присвячений розробці настільних і серверних додатків Windows, інтегроване середовище розробки (IDE) Microsoft Visual Studio все більше нагадує швейцарський армійський ніж, здатний підтримувати безліч обчислювальних платформ, мов та середовища виконання середовища.

Це перша версія IDE для включення компілятора нового покоління Roslyn, — сказав С. Сома Сомасегар, віце-президент корпорації Microsoft у відділі розробників. Розроблений корпорацією Майкрософт, Roslyn пропонує безліч розширених можливостей налагодження для мов програмування компанії C # та Visual Basic [14].

Розроблені на Roslyn поліпшення "є однією з найважливіших нових функцій для професійного розробника", — написав аналітик програмного забезпечення IDC Al Hilwa в електронному листі. Наприклад, функція "лампочки" на основі Rosyln

може активно пропонувати виправлення для зламаного коду або шляхи вдосконалення робочого коду.

Розроблені на Roslyn поліпшення "є однією з найважливіших нових функцій для професійного розробника", — написав аналітик програмного забезпечення IDC Al Hilwa в електронному листі. Наприклад, функція "лампочки" на основі Rosyln може активно пропонувати виправлення для зламаного коду або шляхи вдосконалення робочого коду

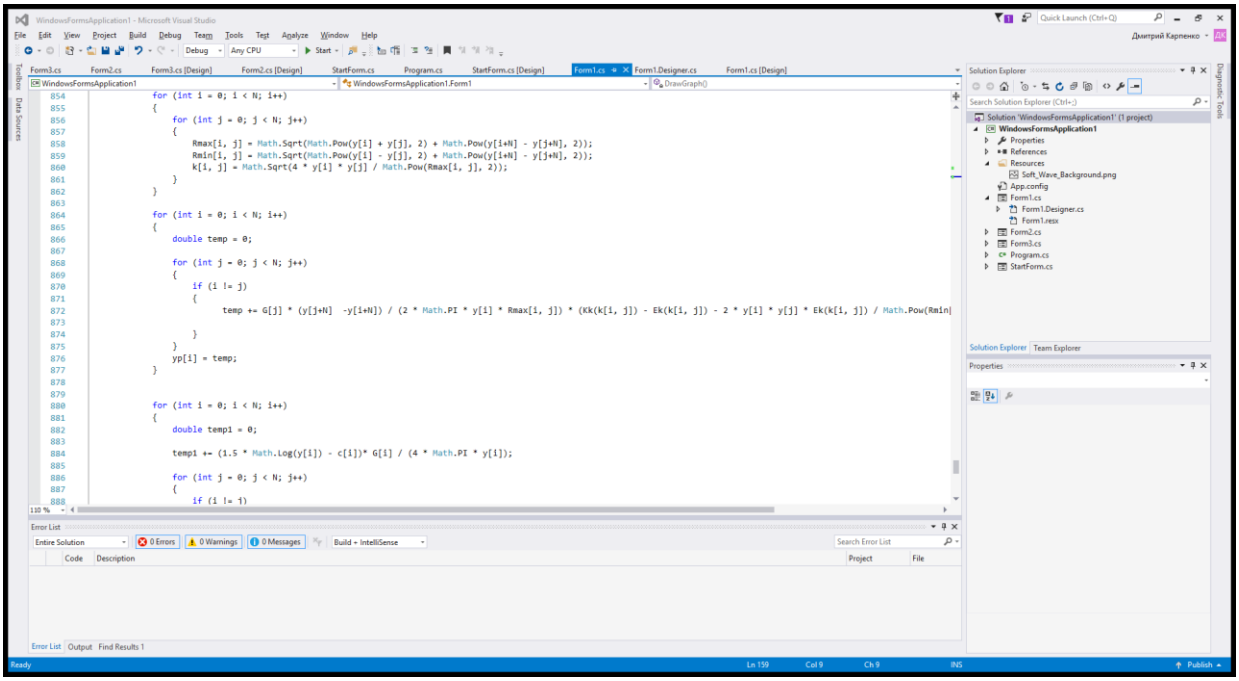


Рисунок 3.1 — Середовище розробки Microsoft Visual Studio

Крім того, завдяки Rosyln, Visual Studio також дає змогу розробникам налаштовувати попередження та пропозиції. Командний проект може використовувати цю функцію для створення набору попереджень, які можуть запобігти розробникам загальних помилок або порушення організаційних інструкцій [15].

Visual Studio дає змогу розробнику написати єдину програму для роботи на кількох платформах Windows, таких як мобільний, настільний і навіть експериментальний середовище Microsoft HoloLens. Він також забезпечує спосіб створення додатків, які взагалі не працюють на комп'ютерах Windows, але замість цього вони працюють на пристроях iOS або у веб-додатках у хмарі.

Visual Studio включає в себе редактор коду, що підтримує IntelliSense (компонент завершення коду), а також рефакторинг коду. Вбудований відладчик працює як відладчик на рівні вихідного рівня, так і відладчик на рівні машини. Інші вбудовані інструменти включають програму для кодування, конструктор форм для побудови графічних інтерфейсів, веб-дизайнер, дизайнер класів та дизайнер схеми баз даних. Він приймає плагіни, які покращують функціональність практично на всіх рівнях, включаючи підтримку систем керування джерельними ресурсами (наприклад, Subversion та Git) та додавання нових наборів інструментів, таких як редактори та візуальні розробники для мов або наборів інструментів для інших аспектів розробки програмного забезпечення життєвий цикл (наприклад, клієнт Team Foundation Server: Team Explorer).

Visual Studio включає в себе редактор коду, що підтримує IntelliSense (компонент завершення коду), а також рефакторинг коду. Вбудований відладчик працює як відладчик на рівні вихідного рівня, так і відладчик на рівні машини. Інші вбудовані інструменти включають програму для кодування, конструктор форм для побудови графічних інтерфейсів, веб-дизайнер, дизайнер класів та дизайнер схеми баз даних. Він приймає плагіни, які покращують функціональність практично на всіх рівнях, включаючи підтримку систем керування джерельними ресурсами (наприклад, Subversion та Git) та додавання нових наборів інструментів, таких як редактори та візуальні розробники для мов або наборів інструментів для інших аспектів розробки програмного забезпечення життєвий цикл (наприклад, клієнт Team Foundation Server: Team Explorer).

Visual Studio підтримує 36 різних мов програмування і дозволяє редакторові коду та відладчику підтримувати (в тій чи іншій мірі) майже будь-яку мову програмування, якщо існує певна мова-служба. Вбудовані мови включають C, C ++, C ++ / CLI, Visual Basic. NET, C #, F #, JavaScript, TypeScript, XML, XSLT, HTML та CSS. Підтримка інших мов, таких як Python, Ruby, Node.js та M, серед інших, доступна через плагіни. Java (і J #) були підтримані в минулому. А також містить в собі сильний інструмент NuGet (рис 3.2) для завантаження різного роду плагінів та додатків.

Visual Studio дозволяє підключити функціональність, кодовану як VSPackage. Після встановлення функціональність доступна як Сервіс. IDE надає три сервіси: SVsSolution, що забезпечує можливість переліку проектів та рішень; SVsUIShell, що забезпечує вікна та функціональність інтерфейсу користувача (включаючи вкладки, панелі інструментів та вікна інструментів); і SVsShell, що займається реєстрацією VSPackages. Крім того, IDE також відповідає за координування та забезпечення зв'язку між службами. Всі редактори, дизайнери, типи проектів та інші інструменти реалізовані як VSPackages. Visual Studio використовує COM для доступу до VSPackages. SDK Visual Studio також включає керовану пакувальну структуру (MPF), яка являє собою набір керованих обгорток навколо COM-інтерфейсів, які дозволяють писати пакунки на будь-якій сумісній мові CLI [10]. Проте MPF не забезпечує всі функціональні можливості, що висуваються через інтерфейси Visual Studio COM [11]. Ці послуги можуть бути використані для створення інших пакетів, які додають до Visual Studio IDE функціональність [16].

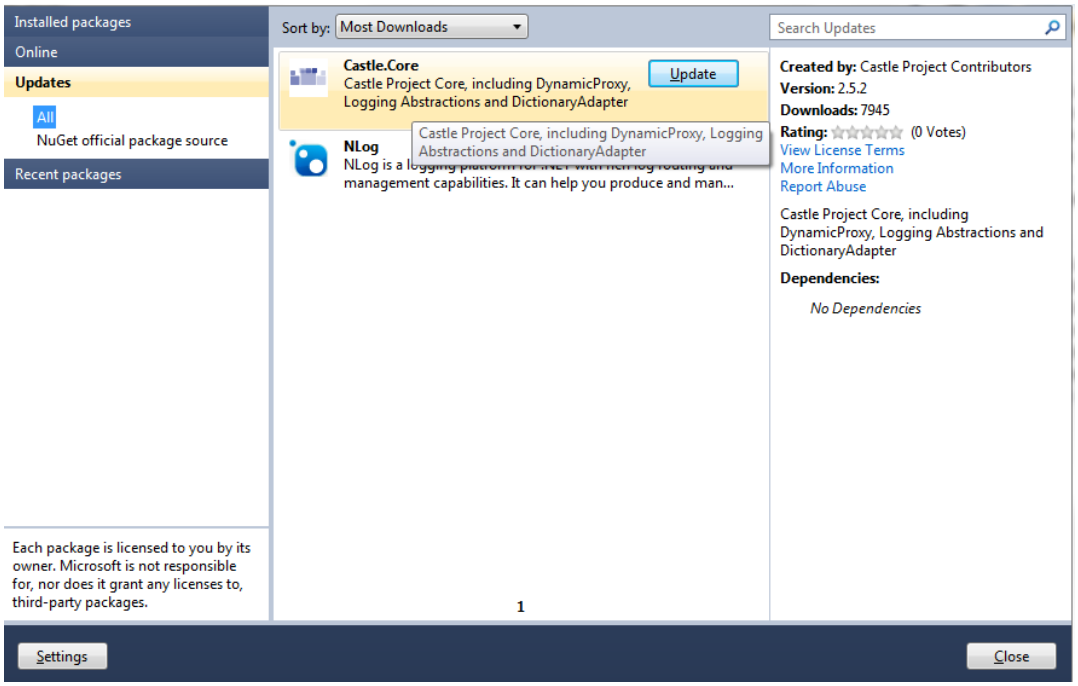


Рисунок 3.2 — Інструмент NuGet

Підтримка мов програмування додана за допомогою певного VSPackage, який називається службою мови. Мовна служба визначає різні інтерфейси, які реалізація VSPackage може здійснювати, щоб додати підтримку для різних функцій.

Функціональні можливості, які можна додати цим способом, включають синтаксичну окраску, завершення звітів, узгодження фігурних підказок, підказки про параметри інформації, списки учасників та маркери помилок для фонової компіляції. Якщо інтерфейс буде реалізовано, функціональність буде доступною для мови. Мовні служби реалізуються на кожній мові. Реалізації можуть повторно використовувати код з аналізатора або компілятора для мови [17]. Мовні служби можуть бути реалізовані як у власному коді, так і в керованому коді. Для власного коду можуть бути використані як рідні COM-інтерфейси, так і Babel Framework (частина Visual Studio SDK). Для керованого коду MPF включає обгортки для написання керованих мовних служб .

Visual Studio не включає в себе будь-яку підтримку управління вихідним кодом, але вона визначає два альтернативні шляхи для інтеграції систем керування вихідними кодами з IDE. Контроль джерела VSPackage може забезпечити власний користувальницький інтерфейс. На відміну від цього плагін керування джерелом, який використовує MSSCCI (інтерфейс керування вихідним кодом Microsoft), надає набір функцій, які використовуються для реалізації різних функцій керування вихідним кодом, з використанням стандартного користувальницького інтерфейсу Visual Studio. MSSCCI вперше був використаний для інтеграції Visual SourceSafe з Visual Studio 6.0, але пізніше був відкритий через Visual Studio SDK. Visual Studio .NET 2002 використовував MSSCCI 1.1, а Visual Studio .NET 2003 використовував MSSCCI 1.2. Visual Studio 2005, 2008 та 2010 використовують MSSCCI версії 1.3, яка додає підтримку для перейменування та видалення розповсюдження, а також асинхронного відкриття .

Visual Studio підтримує запуск декількох екземплярів середовища (кожен зі своїм набором VSPackages). У прикладах використовуються різні вулики реєстру (див. Визначення MSDN терміна «вулик реєстру» у тому значенні, яке використовується тут) для зберігання їх конфігураційного стану і диференціюються за допомогою їх AppId (ідентифікатор програми). Екземпляри запускаються за допомогою специфічного для AppId .exe, який вибирає AppId, встановлює корінь вулика та запускає IDE. VSPackages, зареєстровані для одного AppId, інтегровані з

іншими VSPackages для цього AppId. Різні випуски продукту Visual Studio створені за допомогою різних AppIds. Випуски Visual Studio Express встановлюються за допомогою власних AppIds, але продукти Standard, Professional та Team Suite мають однаковий AppId. Отже, ви можете встановити видання Express одночасно з іншими виданнями, на відміну від інших випусків, які оновлюють одну і ту ж установку. Професійне видання містить у собі унікальний набір VSPackages у стандартному випуску, а комплектний комплект включає в себе надшипну версію VSPackages в обох інших виданнях. Система AppId використовується Visual Studio Shell у Visual Studio 2008 [18].

Нова версія програмного забезпечення організовує свою колекцію інструментів для налагодження та профілювання за допомогою єдиного інтерфейсу інструментів діагностики. Тепер розробник може отримати інформацію про продуктивність коду безпосередньо з вікна редагування.

Цей випуск також має значний вплив на те, що він повністю підтримує бачення Microsoft для створення універсальних програм Windows, — сказав Хілва. За допомогою Visual Studio розробник може одночасно написати комп'ютерну програму та працювати на кількох пристроях під керуванням Windows, включаючи настільні та мобільні клієнти, а також навіть експериментальну HoloLens голографічну обчислювальну систему компанії.

Microsoft працює над розширенням Visual Studio за рамки початкової бази розробників програмного забезпечення Windows. Раніше цього року він випустив Visual Studio Code, основний редактор коду для Apple Mac, Linux та Windows клієнтів. За три місяці з моменту його випуску, код Visual Studio було завантажено понад 500 000 разів, при цьому більше половини для Linux і Mac. Microsoft продовжує розширювати Visual Studio для підтримки нових мов за межами власного Microsoft. Розробники тепер можуть створювати веб-додатки в середовищі IDE, використовуючи вузол сервера Java-середовища виконання середовища.

У новому випуску представлений набір інструментів мобільного розроблення, який дозволяє програмістам створювати додатки для телефонів iOS, Android та Windows, використовуючи свої власні мови.

IDE полегшує підключення до хмарних середовищ. Розробники можуть надавати свої заявки в хмару Microsoft Azure одним кліком. Або вони можуть упакувати свої програми в контейнери Docker, щоб вони могли працювати в будь-якому хмарному сервісі.

Програмне забезпечення також може легко зачепитись до програмного забезпечення Microsoft для керування командними проектами, Team Foundation Server 2015 та Visual Studio Online, обидва з яких служать базою для швидкого середовища розробки, що розробляються за допомогою Devops.

Як і будь-який інший IDE, він включає редактор коду, який підтримує виділення синтаксису та завершення коду за допомогою IntelliSense для змінних, функцій, методів, циклів та запитів LINQ. IntelliSense підтримується для включених мов, а також для XML, каскадних таблиць стилів та JavaScript при розробці веб-сайтів та веб-програм. Автозаповнення пропозицій з'являються у вікні списку без моделей над вікном редактора коду в безпосередній близькості від курсору редагування. У Visual Studio 2008 він може бути тимчасово напівпрозорий, щоб побачити код, який його заблокував. Редактор коду використовується для всіх підтримуваних мов.

Редактор коду Visual Studio також підтримує налаштування закладок у коді для швидкої навігації. Інші навігаційні засоби включають згортання блоків коду та послідовного пошуку, крім звичайного текстового пошуку та пошуку регулярного пошуку. Редактор коду також включає в себе багатопозиційний буфер обміну та список завдань. Редактор коду підтримує фрагменти коду, які є збереженими шаблонами для повторюваного коду, і можуть бути вставлені в код і налаштовані для роботи над проектом. Вбудований інструмент управління фрагментами коду. Ці інструменти з'являються у вигляді плаваючих вікон, які можуть бути встановлені для автоматичного приховування, коли вони не використовуються або стикуються збоку екрана. Редактор коду Visual Studio також підтримує рефакторинг коду, включаючи реорганізацію параметрів, зміну і перейменування методів, вилучення інтерфейсу та інкапсуляцію членів класу всередині властивостей, серед іншого.

Visual Studio має фонову компіляцію (також називається додатковою компіляцією). У міру написання коду Visual Studio компілює його у фоновому

режимі, щоб надати відгук про помилки синтаксису та компіляції, позначені червоним хвилястим підкресленням. Попередження позначені зеленим підкресленням. Тональна компіляція не генерує виконуваний код, оскільки для цього потрібен інший компілятор, ніж той, який використовується для створення виконуваного коду. [19] Спочатку була введена довідкова компіляція з Microsoft Visual Basic, але тепер її було розширено для всіх включених мов.

3.2 Програмна технологія .NET Framework

.NET Framework — це програмне забезпечення, розроблене корпорацією Майкрософт, яке працює переважно на Microsoft Windows. Вона включає в себе бібліотеку великого класу з ім'ям Framework Class Library (FCL) і забезпечує сумісність мов (кожен мову може використовувати код, написаний іншими мовами) на кількох мовах програмування. Програми, написані для .NET Framework, виконуються в програмному середовищі (на відміну від апаратного середовища) називається Common Language Runtime (CLR) — віртуальна машина додатків, яка надає послуги, такі як безпека, керування пам'яттю та обробка виключень. Таким чином, комп'ютерний код, написаний з використанням .NET Framework, називається "керованим кодом". FCL і CLR разом складають .NET Framework.

FCL забезпечує користувальницький інтерфейс, доступ до даних, підключення до бази даних, криптографію, розробку веб-додатків, цифрові алгоритми та мережеві зв'язки. Програмісти створюють програмне забезпечення, об'єднуючи їх вихідний код з .NET Framework та іншими бібліотеками. Рамка призначена для використання більшістю нових додатків, створених для платформи Windows. Корпорація Майкрософт також випускає інтегроване середовище розробки, головним чином для .NET програмного забезпечення, яке називається Visual Studio [20].

.NET Framework розпочалася як фірмове програмне забезпечення, хоча фірма прагнула стандартизувати пакет програмного забезпечення практично відразу, навіть до її першого випуску. Незважаючи на зусилля щодо стандартизації, розробники, головним чином, у вільних та відкритих програмних спільнотах, висловлювали

занепокоєння вибраними умовами та перспективами здійснення будь-якої вільної та відкритої версії, особливо стосовно патентів на програмне забезпечення.

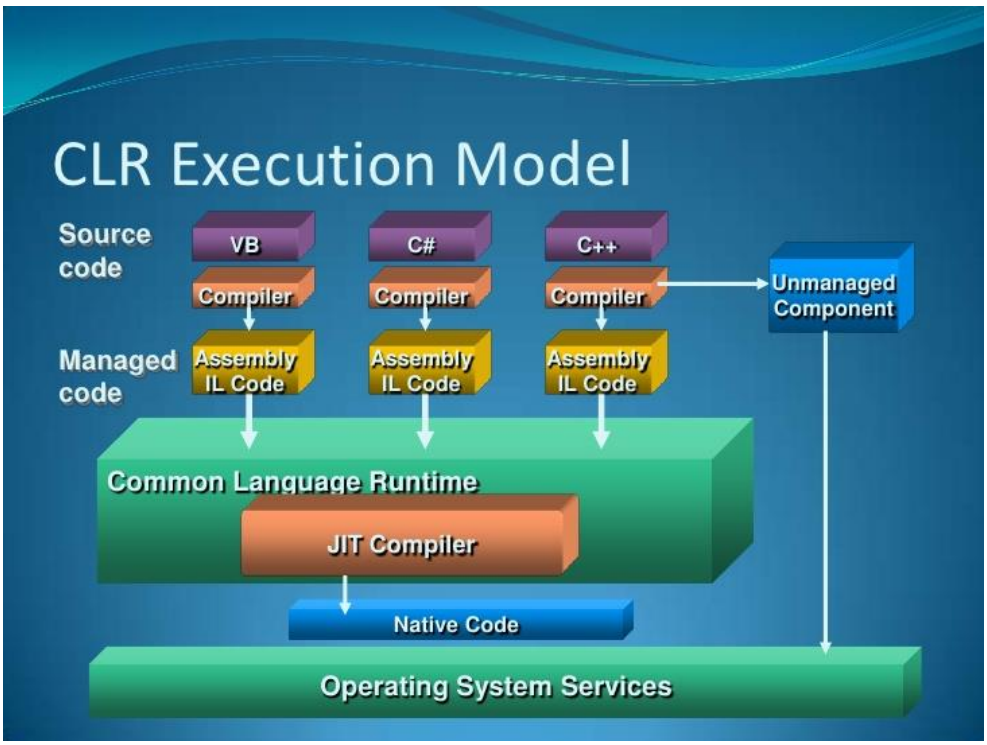


Рисунок 3.3 — Технологія .NET

.NET Framework привів до створення сімейства платформ .NET, орієнтованих на мобільні обчислення, вбудовані пристрої, альтернативні операційні системи та плагіни веб-браузера. Скорочена версія системи, .NET Compact Framework, доступна на платформах Windows CE, включаючи пристрої Windows Mobile, такі як смартфони. .NET Micro Framework призначений для вбудованих пристроїв з обмеженими ресурсами. Silverlight доступний як плагін веб-браузера. Mono доступний для багатьох операційних систем і налаштований на популярні операційні системи для смартфонів (Android і iOS) та ігрові движки. .NET Core орієнтована на універсальні платформи Windows (UWP), а також крос-платформні та обласні обчислювальні навантаження.

Microsoft почала розробку .NET Framework наприкінці 1990-х років, спочатку під назвою служб Windows Next Generation (NGWS), як частину стратегії .NET. Наприкінці 2000 року були випущені перші бета-версії .NET 1.0.

Хоча Microsoft та їх партнери володіють патентами на CLI та C #, ECMA та ISO

вимагають, щоб всі патенти, необхідні для реалізації, були доступними за "розумних та недискримінаційних умов". Фірми погодилися дотримуватися цих умов і зробити патенти доступними безоплатно. Проте це не стосується частини платформи .NET Framework, які не охоплені стандартами ECMA-ISO, включаючи Windows Forms, ADO.NET та ASP.NET. Патенти, які Microsoft володіє в цих сферах, можуть стримувати реалізацію не Майкрософт у повній структурі.

3 жовтня 2007 року корпорація Майкрософт оголосила, що вихідні коди бібліотек .NET Framework 3.5 повинні бути доступні за ліцензією Microsoft Reference Source (Ms-RSL). Репозиторій вихідного коду стає доступним в Інтернеті 16 січня 2008 р. І включає BCL, ASP.NET, ADO.NET, Windows Forms, WPF та XML. Скотт Гатрі з Microsoft пообіцяв, що бібліотеки LINQ, WCF та WF були додані.

У листопаді 2014 року корпорація Майкрософт також оновила свої патентні грантові рішення, що додатково розширює сферу застосування, ніж попередні зобов'язання. Попередні проекти, такі як Mono, існували в законній сірій зоні, оскільки попередні грантові програми Microsoft застосовуються лише до технології в "критичних специфікаціях", включаючи чітко четверте видання кожного ECMA-334 та ECMA-335. Однак нова патентна обіцянка не встановлює верхньої межі для версії специфікації та навіть поширюється на будь-які технології .NET, що виконуються, задокументованих на MSDN, які формально не зазначено групою ECMA, якщо проект вирішить їх реалізувати. Це дозволяє Mono та іншим проектам підтримувати співвідношення між характеристиками сучасних .NET, які були введені з моменту публікації 4-го видання без ризику патентного судочинства щодо реалізації цих функцій. Новий грант підтримує обмеження, що будь-яка реалізація повинна забезпечувати мінімальне дотримання обов'язкових частин специфікації CLI.

31 березня 2016 р. Корпорація Майкрософт оголосила в Microsoft Build, що вони повністю перейдуть до ліцензії Mono за ліцензією MIT навіть у сценаріях, де раніше потрібна комерційна ліцензія. Microsoft також доповнила свої попередні патентні обіцянки щодо Mono, заявивши, що вони не будуть заявляти про "застосовні патенти" проти партій, які "використовують, продають, пропонують для продажу, імпортують або поширюють Mono". Це було що проект Mono був внесений до .NET

Foundation. Ці події спричинили придбання Xamarin, який почався в лютому 2016 року і був завершений 18 березня 2016 року.

У прес-релізі Microsoft висвітлено, що тепер крос-платформне зобов'язання забезпечує повний набір серверів з повністю відкритим вихідним кодом .NET. Проте, Microsoft не планує випускати джерело для WPF або Windows Forms.

C # — об'єктно-орієнтована мова програмування. Розроблено в 1998-2001 роках групою інженерів компанії Microsoft під керівництвом Андерса Хейлсберг і Скотта Вільтаумота як мову розробки додатків для платформи Microsoft .NET Framework. Згодом був стандартизований як ECMA-334 і ISO / IEC 23270.

C # відноситься до сім'ї мов з C-подібним синтаксисом, з них його синтаксис найбільш близький до C ++ і Java. Мова має статичну типізацію, підтримує поліморфізм, перевантаження операторів (в тому числі операторів явного і неявного приведення типу), делегати, атрибути, події, властивості, узагальнені типи і методи, ітератори, анонімні функції з підтримкою замикань, LINQ, виключення, коментарі у форматі XML.

Перейнявши багато від своїх попередників - мов C ++, Pascal, Модула, Smalltalk і, особливо, Java - C #, спираючись на практику їх використання, виключає деякі моделі, що зарекомендували себе як проблематичні при розробці програмних систем, наприклад, C # на відміну від C ++ і деяких інших мов, не підтримує множинне успадкування класів (між тим допускається множинне спадкування інтерфейсів).

C # розроблявся як мова програмування прикладного рівня для CLR і, як такий, залежить, перш за все, від можливостей самої CLR. Це стосується, перш за все, системи типів C #, яка відображає BCL. Присутність або відсутність тих чи інших виразних особливостей мови диктується тим, чи може конкретна мовна особливість бути трансльований в відповідні конструкції CLR. Так, з розвитком CLR від версії 1.1 до 2.0 значно збагатився і сам C #; подібної взаємодії слід очікувати і в подальшому (проте, ця закономірність була порушена з виходом C # 3.0, що представляє собою розширення мови, що не спираються на розширення платформи .NET). CLR надає C #, як і всім іншим .NET-орієнтованим мовам, багато можливостей, яких позбавлені «класичні» мови програмування. Наприклад, прибирання сміття не реалізована в

самому C #, а проводиться CLR для програм, написаних на C # точно так же, як це робиться для програм на VB.NET, J # і ін.

Проект C # був початий в грудні 1998 і отримав кодову назву COOL (C-style Object Oriented Language). Версія 1.0 була анонсована разом з платформою .NET в червні 2000 року, тоді ж з'явилася і перша загальнодоступна бета-версія; C # 1.0 остаточно вийшов разом з Microsoft Visual Studio .NET в лютому 2002 року.

Перша версія C # нагадувала за своїми можливостями Java 1.4, кілька їх розширюючи: так, в C # були властивості (виглядають в коді як поля об'єкта, але на ділі викликають при зверненні до них методи класу), індексатори (подібні до властивостей, але приймають параметр як індекс масиву), події, делегати, цикли foreach, структури, що передаються за значенням, автоматичне перетворення вбудованих типів в об'єкти при необхідності (boxing), атрибути, вбудовані засоби взаємодії з некерованим кодом (DLL, COM) та інше.

Крім того, в C # вирішено було перенести деякі можливості C ++, відсутні в Java: беззнакові типи, перевантаження операторів (з деякими обмеженнями, на відміну від C ++), передача параметрів в метод по посиланню, методи зі змінним числом параметрів, оператор goto (з обмеженнями). Також в C # залишили обмежену можливість роботи з покажчиками - в місцях коду, спеціально позначених словом unsafe і при вказівці спеціальній опції компілятору.

C# підтримує строго типізовані неявні оголошення змінних з ключовим словом var і неявно типізовані масиви з ключовим словом new [], за яким слідує ініціалізатор колекції.

C# підтримує суворий тип даних Boolean, bool. Вирази, які приймають умови, такі як while та if, вимагають висловлювання, що реалізує оператор true або false. Хоча C++ також має тип Boolean, він може бути вільно перетворений в цілі числа та з них, а вирази, такі як if(a), вимагають тільки того, щоб a був конвертований в bool, що дозволяє бути a int-типу або вказівником. C# забороняє «ціле значення означає справжній або помилковий підхід» на тій підставі, що примус програмістів використовувати вирази, які повертають точно bool, можуть створювати деякі типи помилок програмування, наприклад if (a = b) (використання присвоювання = замість

рівності == , які, хоча і не є помилкою на C або C ++, все одно будуть спіймані компілятором).

C# безпечніший в порівнянні з C++. Єдиними неявними перетвореннями за замовчуванням є ті, які вважаються безпечними, наприклад, розширення цілих чисел. Це застосовується під час компіляції, під час JIT і, в деяких випадках, під час виконання. Не відбувається неявних перетворень між булевими і цілими числами, а також між членами перерахування і цілими числами (крім літерала 0, який може бути неявно перетворений в будь-який нумерований тип). Будь-яке призначене для користувача перетворення повинно бути явно позначене як явне або неявне, на відміну від конструкторів копіювання C++ і операторів перетворення, які за умовчанням є неявними.

C# має явну підтримку ковариації та контраваріантності в родових типах, на відміну від C++, яка має певний рівень підтримки контраваріантності просто через семантику типів,що повертаються, на віртуальні методи.

Мова C # не допускає глобальних змінних або функцій. Всі методи і члени повинні бути оголошені всередині класів. Статичні члени відкритих класів можуть замінювати глобальні змінні та функції.

Методи в мові програмування є членами класу в проекті, деякі методи мають підписи, а деякі не мають підпису. Методи можуть бути недійсними або можуть повертати щось на зразок рядка, цілого, подвійного, десяткового, float і bool. Якщо метод недійсний, це означає, що метод не повертає жодного типу даних.

Подібно C++, і на відміну від Java, програмісти на C # повинні використовувати ключове слово virtual, щоб дозволити перевизначати методи підкласами[1].

Методи розширення в C# дозволяють програмістам використовувати статичні методи, як якщо б вони були методами з таблиці методів класу, дозволяючи програмістам додавати методи до об'єкта, який, на їхню думку, повинен існувати на цьому об'єкті і його похідних.

Динамічний тип dynamic допускає прив'язку методу під час виконання, що дозволяє використовувати JavaScript-подібні виклики методів і склад часу виконання.

У C# є підтримка строго типізованих покажчиків функцій через delegate ключового слова. Подібно псевдо-C ++ - signal і slot фрейма Ct, C# має семантику, спеціально пов'язану з подіями стилю публікації-підписки, хоча C# використовує делегати для цього.

C # пропонує Java-подібні синхронізовані synchronized виклики методів через атрибут [MethodImpl (MethodImplOptions.Synchronized)] і підтримує взаємовиключні блокування за допомогою блокування ключових слів.

Хоча визначення мови C# і CLI стандартизовані ISO та Ecma, що забезпечує розумний і недискримінаційний ліцензійний захист (RAND) від патентних позовів, Microsoft використовує C# і CLI у своїй бібліотеці Base Class Library (BCL), яка є фундаментом їхньої власницької платформи .NET framework, і яка забезпечує безліч нестандартизованих класів (розширений I/O, GUI Windows Forms, веб-служби тощо). У деяких випадках, де патенти Microsoft відносяться до стандартів, використаних у .NET framework, документовані Microsoft, і застосовані патенти доступні через інші RAND умови або через Обітницю Відкритої Специфікації Microsoft (Microsoft's Open Specification Promise, OSP), які випускають патентні права публічно[7]. Але є деякі застереження і обговорення про те, що існують додаткові аспекти, патентовані Microsoft, що не покриті, які можуть утримувати незалежних реалізаторів повного фреймворку. Microsoft також погодився не позиватися проти розробників відкритого програмного забезпечення щодо порушення прав у неприбуткових проектах для частини свого фреймворку, покритого OSP[8]. Microsoft погодився не порушувати патентних вимог щодо продуктів Novell проти платних клієнтів Novell за винятком переліку продуктів, що явно не згадують C#, .NET чи реалізацію .NET від Novell (проект Mono). Проте Novell дотримується точки зору, що Mono не порушує жодного патенту Microsoft. Microsoft також уклав спеціальну угоду не позиватися проти браузерного плагіну Moonlight, який спирається на Mono, отриманного від Novell.

C# має явну підтримку коваріації та контраваріантності в родових типах, на відміну від C++, яка має певний рівень підтримки контраваріантності просто через семантику типів,що повертаються, на віртуальні методи.

3.3 Застосування методології об’єктно-орієнтованого програмування

Виклик програмування розглядався як написання логіки, а не як визначення даних. Об'єктно-орієнтоване програмування вважає, що те, що нам треба це об'єкти, якими ми хочемо маніпулювати, а не логіка, необхідна для маніпулювання ними. Приклади об'єктів від людини (описуються за назвою, адресою тощо) до будівель та поверхів (властивості яких можна описати та керувати) до маленьких віджетів на робочому столі комп'ютера (наприклад, кнопки та смуги прокрутки) [23].

Першим кроком в ООР є визначення всіх об'єктів, якими програміст хоче маніпулювати, і те, як вони стосуються один одного. Як тільки об'єкт ідентифіковано, він узагальнюється як клас об'єктів, який визначає тип даних, які він містить, та будь-які логічні послідовності, які можуть керувати ним. Кожна чітка логічна послідовність відома як метод. Об'єкти спілкуються з чітко визначеними інтерфейсами. На рисунку 3.3 зображено основні принципи ООП.

Першим кроком в ООР є визначення всіх об'єктів, якими програміст хоче маніпулювати, і те, як вони стосуються один одного. Як тільки об'єкт ідентифіковано, він узагальнюється як клас об'єктів, який визначає тип даних, які він містить, та будь-які логічні послідовності, які можуть керувати ним. Кожна чітка логічна послідовність відома як метод. Об'єкти спілкуються з чітко визначеними інтерфейсами. На рисунку 3.4 зображено основні принципи ООП.

Поняття та правила, що використовуються в об'єктно-орієнтованому програмуванні, надають переваги, які розглянуто далі.Поняття класу даних дає змогу визначити підкласи об'єктів даних, які поділяють деякі або всі основні характеристики класу. Наслідування, це властивість ООП, яка змушує проводити більш детальний аналіз даних, зменшує час розробки та забезпечує більш точне кодування [24].

Оскільки клас визначає лише дані, які потребують до нього, коли екземпляр цього класу (об'єкта) запускається, цей код не зможе випадково отримати доступ до

інших даних програми. Ця характеристика приховування (інкапсуляції) даних забезпечує більшу безпеку системи та дозволяє уникнути непередбачених пошкоджень даних.

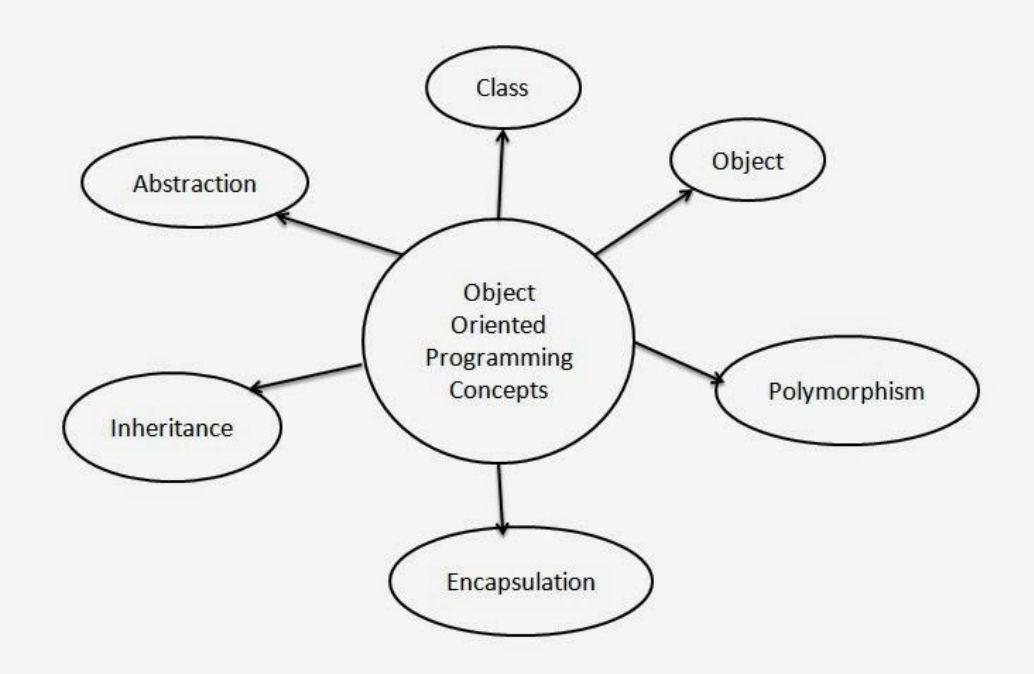


Рисунок 3.4 — Принципи ООП

Визначення класу може бути багаторазово використане не тільки програмою, для якої вона спочатку була створена, але також і іншими об'єктно-орієнтованими програмами (і, з цієї причини, може бути більш легко розподілена для використання в мережах). Поняття класів дає змогу програмісту створити будь-який новий тип даних, який ще не визначено самою мовою.

Simula була першою об'єктно-орієнтованою мовою програмування. Java, Python, C ++, Visual Basic .NET та Ruby — це найпопулярніші мови ООР сьогодні. Мова програмування Java розроблений спеціально для використання в розподілених програмах корпоративних мереж та Інтернету. Ruby використовується у багатьох веб-додатках. Curl, Smalltalk, Delphi та Eiffel — також приклади об'єктно-орієнтованих мов програмування [25].

У нашому випадку ООП застосоване в ПЗ для розділення сущностей, які відповідають за розрахунок траєкторії, визначення спектру, визначенню тиску і розразунку кореляційної функції.

3.4 Технологія ReSharper

ReSharper це плагін для Visual Studio, який додає безліч чудових кодів навігації та редагування, які кожен користувач хотів би мати в Visual Studio. Приклад роботи наведений на рисунку 3.5.

Visual Studio, незважаючи на те, що вона існує протягом більше десяти років і має підтримку для створення всіх видів додатків, все ще не вистачає основних функцій навігації та редагування коду.

Плагін допомагає також функцією доповнення тексту. Можна ввести лише декілька символів і плагін запропонує потрібну змінну, метод.

Visual Studio набагато відстає від того, що забезпечує ReSharper, і ця відстань з часом збільшується. Нова версія ReSharper випускається кожні кілька місяців, тоді як Visual Studio оновлюється раз на 2 — 3 роки.

Деякі функції рефакторингу, які пропонує ReSharper, дозволять заощадити години та години розладу. У проекті ASP.NET MVC ви перейменовуєте дію, але не забудьте перейменувати посилання в представленні. Ваша програма порушена, і ви не знатимете, якщо ви не запустите програму і не знайдете її на цій сторінці.

Плагін встановлюється без особливих проблем, але має недолік, тим що є платним.

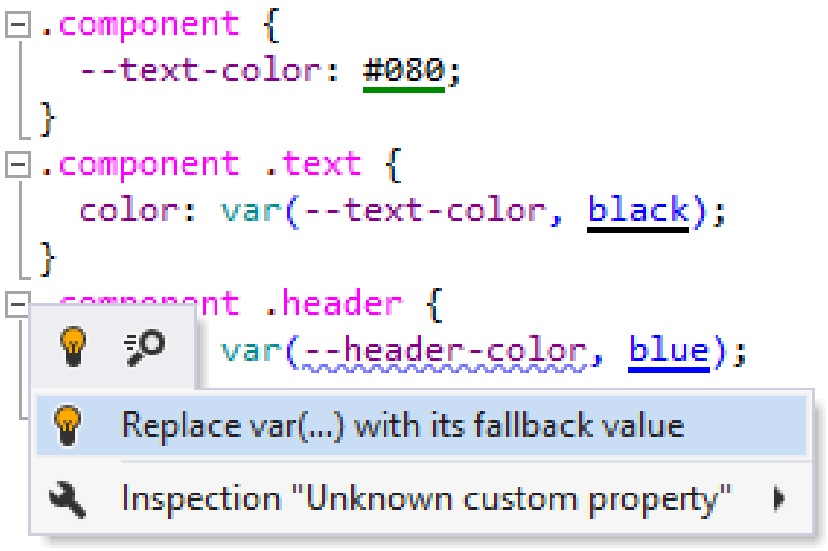


Рисунок 3.5 — Принципи роботи плагіну ReSharper

ReSharper спасла мене від цього конкретного питання протягом останніх двох-трьох років. Якщо я перейменую дію, ReSharper автоматично оновить свої посилання в переглядах. Два роки тому Visual Studio не зробив цього, і я не впевнений, чи зможе вона зараз.

3.5 Графічний плагін

ZedGraph — це бібліотека класів, Windows Forms UserControl та ASP веб-доступне керування для створення 2D ліній, барів та графіків з пікових аркушів довільних наборів даних. Класи забезпечують високий ступінь гнучкості — майже кожен аспект графа може бути змінений користувачем. У той же час, використання класів залишається простим, якщо встановлювати значення за замовчуванням для всіх атрибутів графіка [26]. Класи включають в себе код для вибору відповідних діапазонів масштабів та розмірів кроків на основі діапазону значень даних, які будуються на графіку. Крім того, ZedGraph сумісний з .NET 2.0 та VS .NET 2005.

ZedGraph підтримується як проект із відкритим вихідним кодом на SourceForge. Сайт включає в себе проект Wiki, документацію, проміжні (CVS) оновлення та всі версії випуску.

ZedGraph включає в себе можливість обробляти осередки дат-часу, наприклад, мітки осі можуть бути засновані на кодуваному значенні дат-часу, яке відображається у будь-якому вигляді від секунд до років. В основі осі дат-часу лежить структура XDate, яка фіксує значення часу і обробляє перетворення в / з широкого діапазону форматів часу дати. Ви, мабуть, цікавитесь, чому я заново створив колесо на цьому, ніж просто використовуючи вбудований клас DateTime (або еквівалент). Основна причина, через яку я створив свій клас, полягає в тому, що XDate зберігає інформацію про час дату як значення System.Double. Тому значення XDate можна просто зберегти у звичайному масиві подвійних, як і будь-який інший масив даних, який переданий ZedGraph. Структура та формат XDate докладно описані в Wiki. Якщо ви використовуєте структури DateTime для зберігання даних, ви можете перетворити їх

безпосередньо в подвійні, сумісні з ZedGraph, за допомогою методу DateTime.ToOADate ().

ZedGraph включає можливості діаграми для стрічок для вертикальних та горизонтальних стрічкових діаграм, стекових стрічкових діаграм, процентних стовпчикових діаграм, накладені гістограми, графіки помилок, високочастотні діаграми, відкриті високочастотні діаграми та японські смужкові діаграми . Структурна діаграма створюється схоже на лінійний графік, за винятком того, що ви використовуєте GraphPane.AddBar (), GraphPane.AddErrorBar () або GraphPane.AddHiLowBar () для створення екземпляра панелі. Можна змішувати бари, лінії та символи на одному графіку, просто додаючи різні типи, враховуючи складність задачі.

Набір зразкових графіків також доступний у Вікі, у комплекті з вихідним кодом (багато зразків включають код C # та VB).

Там є багато графічних бібліотек, але ніхто, здавалося, не підходив до того, що мені потрібно. Я виявив, що MSChart теж химерний, і багато інших варіантів не мали гнучкості, необхідної для полірованого вигляду. Звичайно, більшість комерційних пакетів зробить трюк, але мені потрібно було щось, що було безкоштовним. Отже, народився ZedGraph [27].

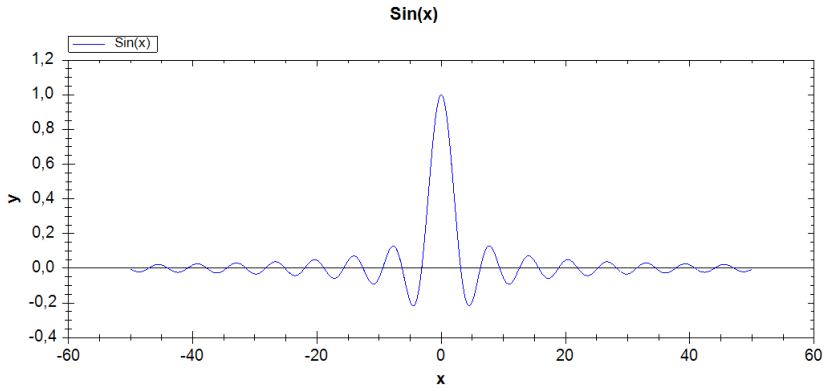


Рисунок 3.6 — Графік синусоїди у Zed Graph

Ці класи генерують різноманітні рядки, бари або кругові діаграми на формі або веб-сторінці, з урахуванням прямокутника місця розташування та деяких точок даних.

ZedGraph обробляє 2D лінії / розкиду графіків, горизонтальні / вертикальні гістограми, стовпчастий діаграми, складених відсотків гістограми, гістограми помилок, відкриту високий-низький-закрити діаграми, японський свічкових графіків і кругові діаграми — це ще не обробляти 2.5 D або 3D поверхні або діаграми. Діаграми можуть бути одягнені з мітками та назвами осі, легендами, текстовими мітками та стрілками (як показано у наведеному вище прикладі), зображення тощо.

Зразки ZedGraphWiki та документація онлайн-класу містять багато корисних порад та описів. Зверніться до них для більш детальної інформації — у ZedGraph є величезна кількість опцій, які не задокументовані в цьому навчальному посібнику. Графічні лінії графіків не є великою справою, але є деякі аспекти класів малювання, які виявилися цікавими. Гнучкість, надана матрицею перетворення бібліотеки GDI +, дуже класна. Це дозволило використовувати той срисамий код для нанесення всіх трьох осей. Система координат трансформується для розміщення розташування та орієнтації осі. У кожному випадку система координат перекладається і повертається так, щоб ось була орієнтована вздовж напрямку X, а початок розташовувався на лівому краю осі при облицюванні з боку позначки. Кілька "якщо" винятків вимагали пояснення тим, що "ліва" сторона осей X і Y2 є мінімальним значенням, а "ліва" сторона осі Y — максимальне значення.

Інший аспект вибору масштабу полягає в тому, чи слід масштаб збільшувати, щоб включити нульове значення. Наприклад, якщо діапазон масштабу становить від 1 до 10, ви зазвичай хочете йти вперед і запустити його з нуля. Це виконується за допомогою параметра за замовчуванням ZeroLever. ZeroLever — це допустима частка, яку діапазон масштабу можна збільшити, включивши нульове значення. Це застосовується нижче діапазону масштабу для позитивних ваг або вище діапазону масштабу для значень негативної шкали. Як приклад, якщо ZeroLever — 0,25, а діапазон даних — від 2,0 до 12,0, тоді масштаб буде збільшено до діапазону від 0,0 до 12,0, оскільки нульове значення лежить на 20% поза межами фактичного діапазону даних (що перевищує 25 % дозволено).

ZedGraph включає в себе можливість обробляти осередки дат-часу, наприклад, мітки осі можуть бути засновані на кодуваному значенні дат-часу, яке відображається

у будь-якому вигляді від секунд до років. В основі осі дат-часу лежить структура XDate, яка фіксує значення часу і обробляє перетворення в / з широкого діапазону форматів часу дати. Ви, мабуть, цікавитесь, чому я заново створив колесо на цьому, ніж просто використовуючи вбудований клас DateTime (або еквівалент). Основна причина, через яку я створив свій клас, полягає в тому, що XDate зберігає інформацію про час дату як значення System.Double. Тому значення XDate можна просто зберегти у звичайному масиві подвійних, як і будь-який інший масив даних, який переданий ZedGraph. Структура та формат XDate докладно описані в Wiki. Якщо ви використовуєте структури DateTime для зберігання даних, ви можете перетворити їх безпосередньо в подвійні, сумісні з ZedGraph, за допомогою методу DateTime.ToOADate ().

ZedGraph включає можливості діаграми для стрічок для вертикальних та горизонтальних стрічкових діаграм, стекових стрічкових діаграм, процентних стовпчикових діаграм, накладені гістограми, графіки помилок, високочастотні діаграми, відкриті високочастотні діаграми та японські смужкові діаграми . Структурна діаграма створюється схоже на лінійний графік, за винятком того, що ви використовуєте GraphPane.AddBar (), GraphPane.AddErrorBar () або GraphPane.AddHiLowBar () для створення екземпляра панелі. Можна змішувати бари, лінії та символи на одному графіку, просто додаючи різні типи.

стрічкових діаграм тик знаки, як правило, знаходяться між кластери барів, які можуть бути виконані за допомогою властивості Axis.MajorTic.IsBetweenLabels. Однак ця властивість застосовується тільки для осей AxisType.Text.

Орієнтація (горизонтальна або вертикальна) та розмір панелей BarItem визначаються глобально за допомогою параметрів GraphPane.BarSettings.Base та інших параметрів GraphPane.BarSettings. Тому всі панелі BarItem матимуть аналогічні властивості, а розмір смуг буде автоматично масштабований, щоб заповнити доступний простір. На відміну від цього, розміри балів ErrorBarItem та бари HiLowBarItem контролюються окремими властивостями для кожного елемента панелі, наприклад, ErrorBarItem.Bar.Size. Ці типи смуг насправді схожі на символи,

оскільки ширина смужки вказана в точках (1/72 дюйма). Один зразок може мати різні ErrorBarItems та HiLowBarItems з різними розмірами.

Два властивості включені в клас GraphPane для управління прогалинами між панелями BarItem; GraphPane.BarSettings.MinBarGap (за замовчуванням = 0.2) — це мінімальний розмір розриву між кожним рядком у кластері барів (кілька стрічок, які мають однакове значення X), а GraphPane.BarSettings.MinClusterGap (за замовчуванням = 1.0) — це мінімальний розмір від розриву між смужками кластерів. Обидва ці параметри виражаються як частка від окремих розмірів стрічки, тобто значення 1,0 призведе до того, що проміжок буде таким же, як у смужках. Зверніть увагу, що ці властивості застосовуються тільки до барів BarItem (не ErrorBarItem або HiLowBarItem барів). Новий клас Бар був доданий до класу BarItem для контролю властивостей панелей. Цей клас Бар має властивості для заповнення.

Висновки до розділу 3

У третьому розділі було:

- проаналізовано та обрано засоби реалізації для створенні програмного забезпечення;
- розроблено діаграму прецедентів, яка представляє алгоритм взаємодії користувача з програмою;
- проведено аналіз сутностей в програмі і розбиття основних з них на класи;
- розроблено систему моделювання звукового поля, яке генерується системою тонких вісесиметричних кілець у морському середовищі;
- обрано та використано описуєму технологію для запису та зчитування даних з текстового файлу.
- застосовано графічний плагін для візуального зображення досліджень.

4 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ СИСТЕМИ

Основна розробка програмного продукту поділялася на 2 частини: побудова інтерфейсу та організація коректної роботи обчислювальних методів.

Візуальна частина була спроектована за допомогою графічного редактора Adobe Photoshop SC6.

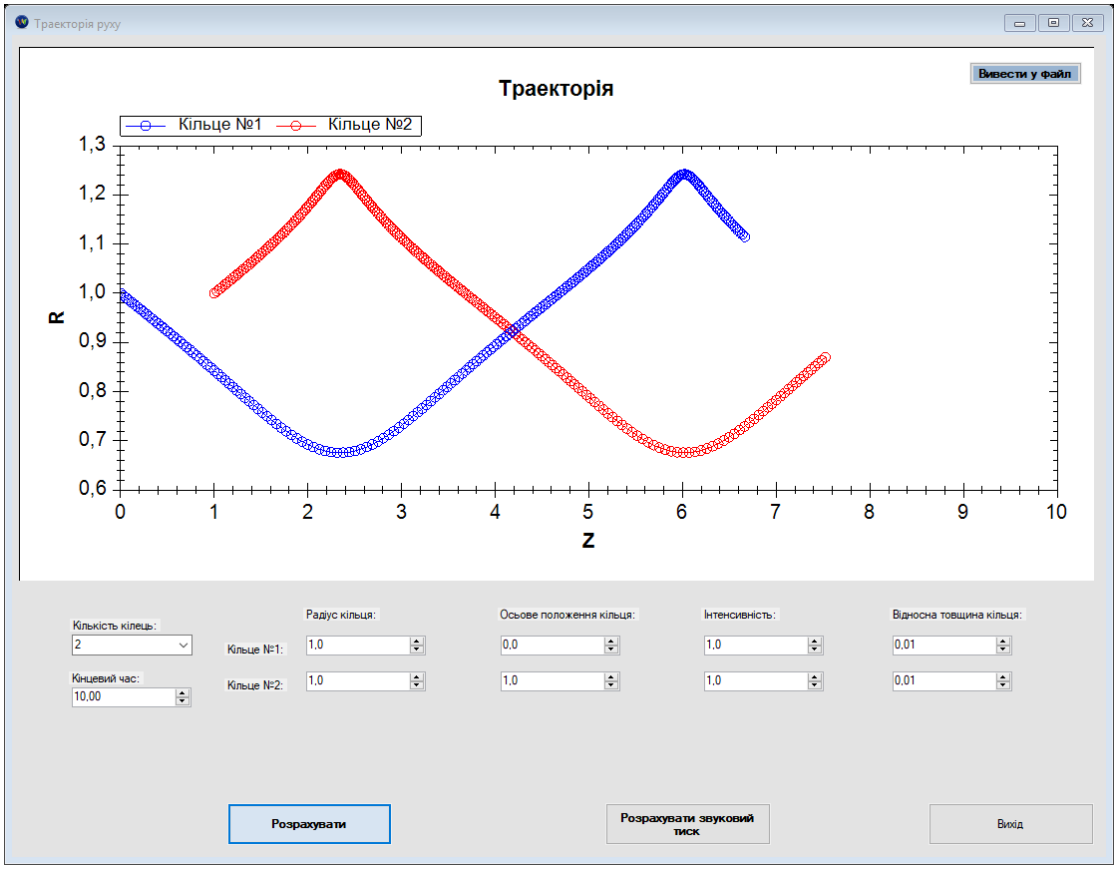


Рисунок 4.1 — Приклад розрахування траєкторії

Користувач має змогу вручну вибрати кількість кілець. В залежності від кількості кілець буде відображатись різна кількість полів для введення параметрів. Для кожного кільця користувач зможе ввести такі параметри як радіус кільця, його осьове положення, інтенсивність випромінювання, а також відносну товщину кільця. Також користувач може ввести кінцевий час випромінювання. Всі параметри вже задані по замовченню, але користувач може змінити будь яке значення. Параметри радіусу, осьового положення і інтенсивності змінюються з кроком в 0.1, а параметр

кінцевого часу і відносної товщини кільця — 0.01. Слід зазначити, що відносна товщина кільця може бути від 0.01 до 0.3.

Звуковий тиск, випромінюваний системою вихорів являє швидко осцилюючу функцію, тому слід проводити розрахунки з малий кроком по часу, а це призведе до втрати точності обчислення другої похідної, незалежно від порядку застосованого алгоритму.

На рисунку 4.1 зображено приклад розрахування траєкторії руху двох вихорових кілець. На даній формі присутні кнопки виходу в меню, кнопка розрахунку траєкторії руху вихорових кілець, кнопка переходу на форму розрахування звукового тиску, який моделюється, виходячи зі значень, які були отримані при моделюванні руху, а також кнопка виходу з програми. На формі є поля для введення значень. Їх кількість залежить від вибраної кількості кілець, які будуть брати участь в генеруванні звукового поля. В програмі передбачено від двох до чотирьох кілець. Всі форми, окрім початкової мають таку ж концепцію.

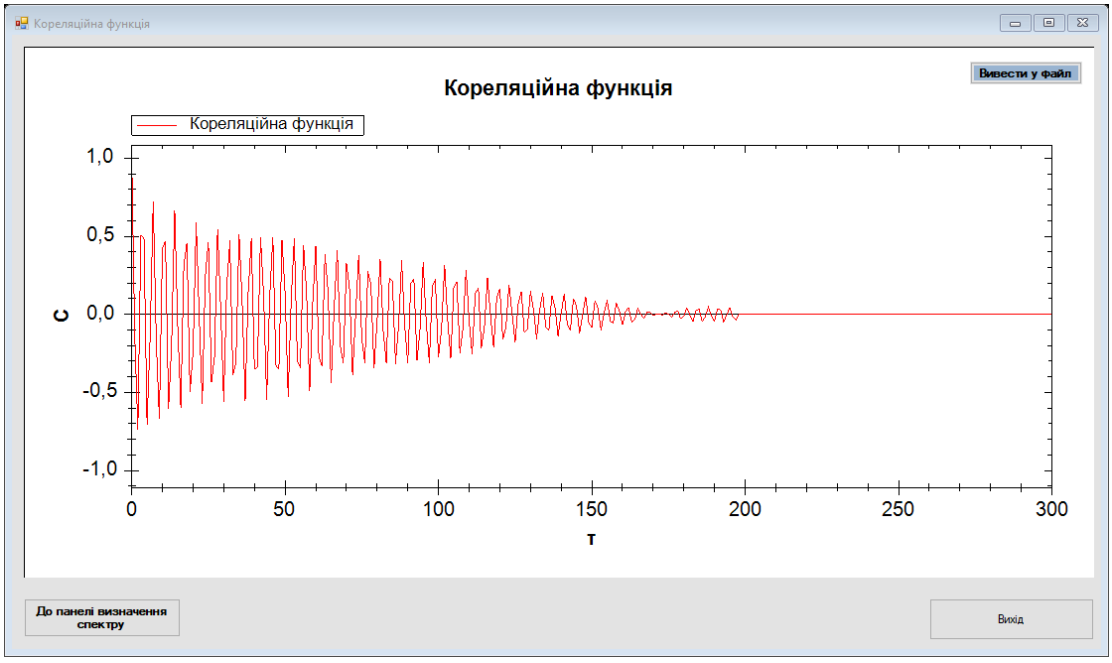


Рисунок 4.2 — Приклад розрахування кореляційної функції

Після того, як користувач дослідив і обчислив звукове поле, він отримав масив значень тиску звукового поля на заданому проміжку часу. Функція передбачає періодичну взаємодію у випадку розв’язання задачі, яка є характерним випадком.

Дослідивши і отримавши значення тиску, користувач може використати кореляційну функцію для того, що б зробити аналіз характеру звуку, який розповсюджується. Якщо графік стрімко йде до нуля, то значить це шум, і ми не можемо ідентифікувати джерело. А якщо ж видна якась періодичність, то значить можна судити, що звук іде від якогось джерела. Результат роботи показано на рисунку 4.2.

Присутня функція “Вивести у файл”. За допомогою її користувач має можливість збереження файлу, або завантаження існуючого файлу з координатами точок, таким чином користувач може використовувати існуючий набір точок для досліджень.

4.1 Діаграма прецедентів користувача

UML-діаграма — діаграма, заснована на UML (Unified Modeling Language — Мова уніфікованої моделювання), з метою візуально представляти систему разом з її головними дійовими особами, ролями, діями, артефактами або класами, з UML — абревіатура, що означає уніфіковану мову моделювання. Простіше кажучи, UML — це сучасний підхід до моделювання та документування програмного забезпечення. Фактично, це одна з найпопулярніших методів моделювання бізнес-процесів.

Вона заснована на схемних уявленнях програмних компонентів. Як свідчить стара прислів'я: "картина стоїть тисячі слів". Використовуючи візуальні уявлення, ми можемо краще зрозуміти можливі недоліки або помилки у програмному забезпеченні або бізнес-процесах.

UML було створено в результаті хаосу, що обертається навколо розробки програмного забезпечення та документації. У 1990-х роках існує декілька способів представлення та документування програмних систем. Виникло потреба у більш уніфікованому способі візуально представляти ці системи, і в результаті в 1994-1996 роках UML було розроблено трьома розробниками програмного забезпечення, що працюють в Rational Software. Пізніше він був прийнятий як стандарт у 1997 році і залишався стандартом з тих пір, коли було отримано лише кілька оновлень.

В основному, UML була використана як загальноприйнята мова моделювання в галузі програмного забезпечення. Проте, він вже знайшов свій шлях до документації декількох бізнес-процесів або робочих процесів. Наприклад, діаграми активності, тип UML-діаграми, можуть бути використані як заміна для блок-схем. Вони забезпечують як більш стандартизований спосіб моделювання робочих процесів, так і широкий діапазон функцій для підвищення читабельності та ефективності.

UML не є самостійною мовою програмування, як-от Java, C ++ чи Python, але, з правильними інструментами, вона може перетворитися на псевдо-мову програмування. Щоб це досягти, вся система повинна бути задокументована в різних діаграмах UML, і, використовуючи правильне програмне забезпечення, діаграми можуть бути безпосередньо переведені в код. Цей метод може бути корисним лише тоді, коли час, необхідний для нанесення діаграм, займе менше часу, ніж написання фактичного коду.

Незважаючи на те, що UML було створено для моделювання програмних систем, він знайшов декілька прийомів у сферах бізнесу або непрограмних системах.

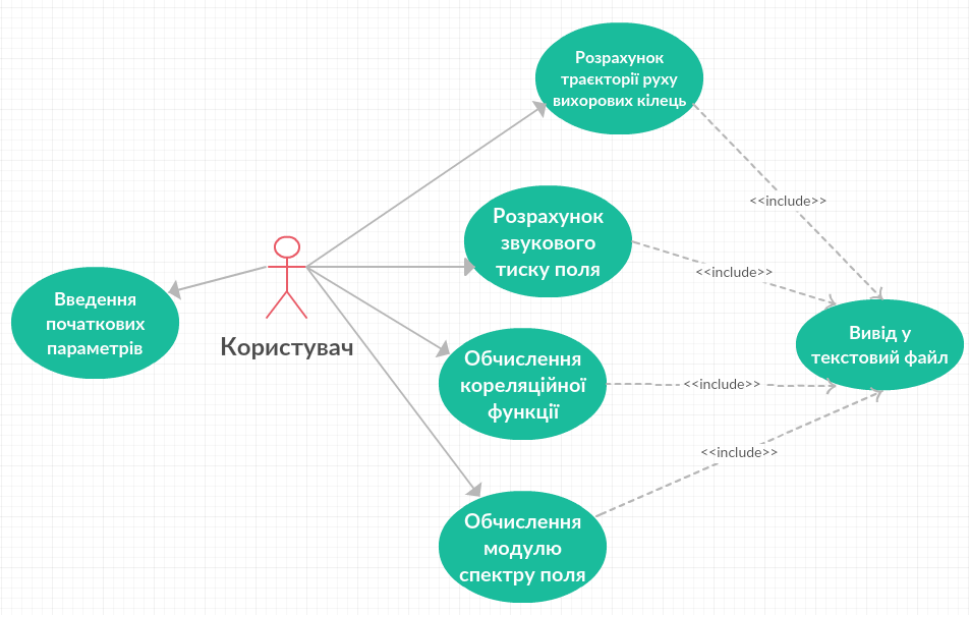


Рисунок 4.3 — Діаграма прецедентів роботи користувача з програмою

На рисунку 4.3 наведено приклад діаграми послідовності роботи користувача з програмою.

UML — діаграми для випадку є корисними для демонстрації динамічної поведінки між суб'єктами всередині системи шляхом спрощення вигляду системи та не відображаючи деталі реалізації.

На діаграмі знаходиться актор і сутності. Сутності відображаються кружечками, в яких вписано дію, яку виконує сутність. Зв’язок користувача з сутностями позначається стрілками.

4.2 Методи запису даних у текстовий файл

Простір імен System.IO містить типи, що дозволяють здійснювати читання і запис у файли і потоки даних, а також типи для базової підтримки файлів і папок. На рисунку 4.4 зображена структура простору імен System.IO.

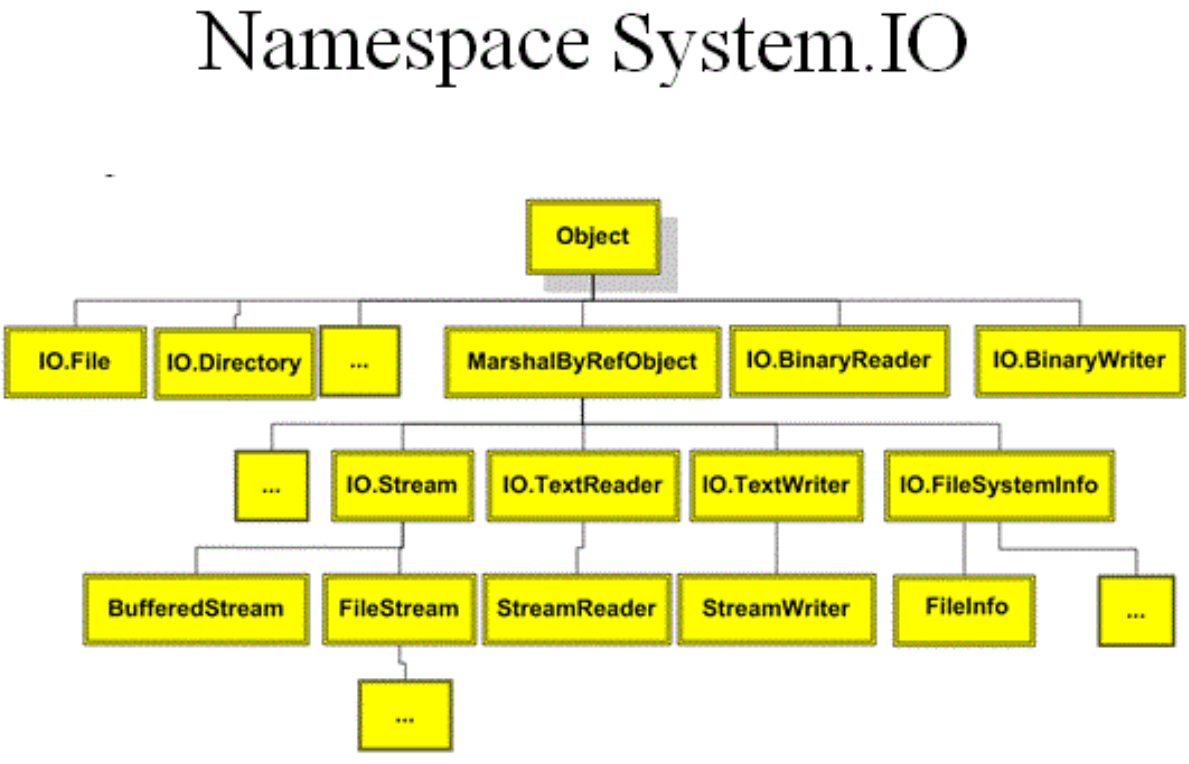


Рисунок 4.4 — Структура простору імен System.IO

Простір імен System.IO в .NET — це область бібліотек базових класів, присвячена службам файлового введення-виведення, а також введення-виведення з пам'яті. Подібно будь-якому простору імен, в System.IO визначено набір класів,

інтерфейсів, перерахувань, структур і делегатів, більшість з яких знаходяться в mscorlib.dll. На додаток до типів, що містяться всередині mscorlib.dll, в збірці System.dll визначені додаткові члени простору імен System.IO. Зверніть увагу, що всі проекти Visual Studio 2010 автоматично встановлюють посилання на обидві збірки.

Операції читання і запису даних в файли в принципі дуже прості, однак виконуються вони не через об'єкти DirectoryInfo і FileInfo. В .NET Framework 4 це робиться через об'єкт File.

До появи .NET Framework 2.0 навколо того, як слід проводити читання і запис даних в файли, велося багато суперечок. Класи з .NET Framework можна було використовувати, але такий підхід не був простим. У версії .NET Framework 2.0 клас File було розширено; з його допомогою стало можливим виконання операцій читання і запису даних в файли за допомогою всього одного рядка коду. Те ж саме є і в .NET Framework 4.

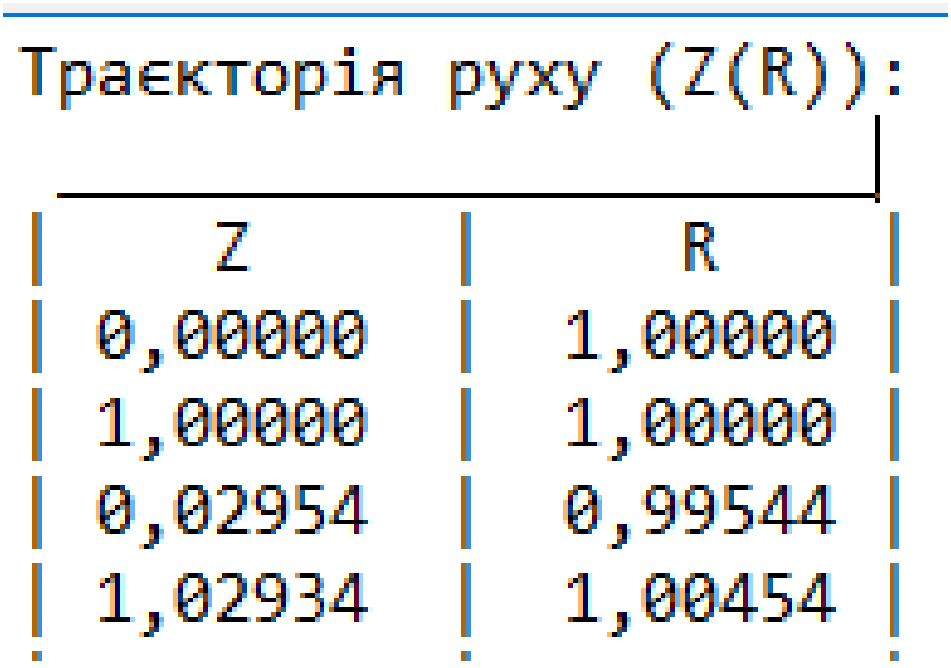
Використовуйте File класу для звичайних операцій, таких як копіювання, переміщення, перейменування, створення, відкриття, видалення і
додавання в один файл одночасно. Можна також використовувати File класу для отримання і установки атрибутів файлу або DateTime відомості, пов'язані зі створенням доступу і записи в файл.

На рисунку 4.4 показано приклад виведення у файл значень, для побудови траєкторії руху для характерного випадку періодичної взаємодії двох однакових вихрових кілець ($R_1^0 = R_1^0 = 1.0$, $Z_1^0 = 0.0$, $Z_2^0 = 1.0$, $n_2^0 = n_1^0 = 0.01$).

Як бачимо на рисунку 4.5, створений текстовий файл можна відкрити в звичайному текстовому редакторі.

Якщо ви збираєтеся використовувати об'єкт неодноразово, рекомендується використовувати відповідний метод примірника FileInfo так як перевірка безпеки не завжди буде використовуватися необхідні .

Статичні методи File класу виконують перевірку безпеки для всіх методів. Якщо ви збираєтеся використовувати об'єкт неодноразово, рекомендується використовувати відповідний метод примірника FileInfo.



Траєкторія руху (Z(R)):

Z	R
0,00000	1,00000
1,00000	1,00000
0,02954	0,99544
1,02934	1,00454

Рисунок 4.5 — Текстовий файл з шуканими значеннями для побудови траєкторії руху

Багато File методи повертають інші типи введення-виведення при створенні або відкритті файлу. Ці інші типи можна використовувати для подальших операцій з файлом. Додаткові відомості див. У розділі певних File такі елементи, як OpenText, CreateText, або Create.

Так як все File методи є статичними, може бути більш ефективно використовувати File методу замість відповідного FileInfo метод примірника, якщо потрібно виконати тільки одну дію. Всі File методів потрібно шлях до файлу, що використовуються для маніпуляції.

Статичні методи File класу виконують перевірку безпеки для всіх методів. Якщо ви збираєтеся використовувати об'єкт неодноразово, рекомендується використовувати відповідний метод примірника FileInfo так як перевірка безпеки не завжди буде використовуватися необхідні.

За замовчуванням читання і запис до нових файлів доступу до всіх користувачів.

Операції читання і запису даних в файли виконуються через об’єкт File.

4.3 Метод розв’язання диференційних рівнянь

Для обчислення і моделювання траєкторії руху вихорового кільця треба дізнатись значення його радіусу і осьового положення в кожний момент часу. Провівши дослідження було отримано систему диференційних рівнянь, за допомогою яких можна вираховувати ці значення. У роботі для знаходження розв`язків задачі Коші було реалізовано метод Рунге-Кутта четвертого порядку.

Метод Рунге-Кутта — великий клас чисельних методів розв'язання задачі Коші для звичайних диференціальних рівнянь і їх систем. Перші методи даного класу були запропоновані десь у 1900 році німецькими математиками К. Рунге і М. В. Куттом.

До класу методів Рунге-Кутта відносяться явний метод Ейлера і модифікований метод Ейлера з перерахунком, які представляють собою відповідно методи першого і другого порядку точності. Існують стандартні явні методи третього порядку точності, що не набули широкого поширення. Найбільш часто використовується і реалізований в різних математичних пакетах (Maple, MathCAD, Maxima) класичний метод Рунге-Кутта, що має четвертий порядок точності. При виконанні розрахунків з підвищеною точністю все частіше застосовуються методи п'ятого і шостого порядків точності. Побудова схем вищого порядку пов'язане з великими обчислювальними труднощами.

Метод Рунге-Кутта четвертого порядку при обчисленнях з постійним кроком інтегрування настільки широко поширений, що його часто називають просто методом Рунге-Кутта [29].

Метод Рунге-Кутта четвертого порядку вляється явним методам. На жаль, явні методи Рунге-Кутта, як правило, непридатні для вирішення жорстких рівнянь через малу області їх абсолютної стійкості. Нестійкість явних методів Рунге-Кутта створює досить серйозні проблеми і при вирішенні диференціальних рівнянь в часних похідних.

Рішення систем диференціальних рівнянь методом Рунге-Кутта є одним з найбільш поширених чисельних методів рішень в техніці [30].

Розв'язання систем диференціальних рівнянь методом Рунге-Кутта є одним з найбільш поширених числових методів розв'язання в техніці. В середовищі MATLAB/Octave (досить поширена і зручна мова програмування для технічних обчислень) реалізований один з його різновидів — метод Дорманда-Принса.

У чисельному аналізі методами Рунге-Кутти є сімейство неявних і явних ітераційних методів, до яких належать добре відома процедура під назвою Метод Ейлера, використовувана при тимчасовій дискретизації для наближених рішень звичайних диференціальних рівнянь.

Загалом, якщо в явному методі Рунге-Кутта явний s має порядок p, то можна довести, що кількість етапів повинна відповідати $s > p$, а якщо $p > 5$, то $s > p + 1$. Проте невідомо, чи ці межі різкі у всіх випадках; наприклад, всі відомі способи порядку 8 мають щонайменше 11 етапів, хоча це можливо, що існують методи з меншою кількістю стадій. Дійсно, відкрита проблема полягає в точності мінімальної кількості етапів s для явного методу Рунге-Кутти мати порядок p у тих випадках, коли ще не знайдено жодних методів, що задовольняють вищезазначеним равенствам.

Висновки до розділу 4

- Розроблено пакет програм, які досліджують звукове поле. Програма надає користувачу можливість побудувати траєкторію руху вихорових структур, обчислити звуковий тиск та спектр поля. Наведені приклади інтерфейсу та приклад взаємодії користувача з програмою;
- надано можливість проводити дослідження з різною кількістю вихорових кілець та вводити вручну початкові параметри кільця;
- обчислено кореляційну функцію, за допомогою якої можна робити аналіз про джерело, яке випромінює звук та характер самого звуку.

5. РОЗРОБЛЕННЯ СТАРТАП ПРОЕКТУ

Ідея проекту полягає у створенні системи моделювання процесу випромінювання звуку при русі вісесиметричних тіл в морському середовищі. Така система допоможе досліджувати звукове поле, яке генерується системою тонких вісесиметричних вихрових кілець. Програма допоможе ідентифікувати тіло, що випромінює звук, визначити траєкторію руху вихорових кілець, обчислити звковий тиск та спектр звукового поля. А також за допомогою систему можна аналізувати характер розповсюдженого звуку.

5.1. Опис ідеї стартап проекту

У підрозділі розглянуться наступні питання:

- ідея та її зміст;
- напрямки застосування;
- переваги для користувача;
- огляд існуючих аналогів.

Наступні пункти подаються у вигляді таблиці (таблиця 5.1) і відповідають на ряд поставлених питань.

Вперше термін «стартап» почав використовуватися Forbes у серпні 1976 року і Business Week у вересні 1977 року для позначення компаній з короткою історією діяльності. Поняття закріпилось в 1990-ті роки і широко розповсюдилось під час буму доткомів.

Творець методики розвитку клієнтів (англ. Customer development) американський підприємець Стів Бланк визначив стартапи як тимчасові структури, що існують для пошуку бізнес-моделі.

Автор книги «Ощадливий стартап» та ідеолог ітеративного підходу в підприємництві Ерік Ріс зазначає, що стартапом може бути названа організація, що створює новий продукт або послугу в умовах високої невизначеності.

Підприємець, венчурний капіталіст і есеїст, засновник бізнес-акселератора Y Combinator Пол Грем вважає швидке зростання головною характеристикою стартапів.

Однак, Пол Грем стверджує, що наявність технологічної інновації та венчурного фінансування не має значення, а малий вік не робить компанію стартапом.

Деякі стартапери розглядають стартапи як культурний феномен — спільні цінності всіх членів команди і відчуття значущості вкладу кожного співробітника. Вони стверджують, що збереження цієї культури дозволяє вважати команду стартапом незалежно від розміру та контролю засновників над компанією.

Таблиця 5.1. Опис ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Створенні системи моделювання процесу випромінювання звуку при русі вісесиметричних тіл в морському середовищі.	1. Аналіз звукового поля.	Можливість побудови графіків траєкторії руху вихорових кілець, графіку тиску звукового поля та спектру звукового поля.
	2. Аналіз характеру випромінюваного звуку	Можливість побудови кореляційної функції, за допомогою якої можно зробити аналіз.

Для порівняння з пропозиціями конкурентних продуктів зроблено аналіз, який передбачає:

- створення списку властивостей та функцій ідеї;
- пошук і аналіз конкурентних продуктів чи проектів, товарів-замінників чи товарів-аналогів, які вже існують у відкритому доступі на ринку;
- створення аналізу для порівняння властивостей та показників: для власного продукту аналізуються а) гірші значення; б) аналогічні значення; в) кращі значення [30].

Створено список потенційних переваг по відношенню до продуктів-конкурентів. Результат аналізу у таблиці 5.2.

Таблиця 5.2. Визначення характеристик ідеї проекту

Техніко-економічні характеристики ідеї	Продукція конкурентів			Слабкі (W), нейтральні (N) та сильні (S) сторони		
Назва продукту	Open Flow	Visual Flow	CFD	W	N	S
Операційна система та версії	Платформи Windows, MacOS	Платформи Windows, MacOS	Платформи Windows			+
Системні вимоги	Мінімальні	Від 2 Гб ОЗУ	Мінімальні			+
Мови програмування	JavaScript	JavaScript	Java		+	
Необхідність встановлення додаткового ПЗ	наявність АПК	наявність АПК	наявність АПК		+	
Ціна	$1,200.00	$500.00	безкоштовний			+

Аналог системи наразі функціонує та представлений як веб-додаток. Система підтримується на будь якій платформі. Вона не потребує особливих системних вимог. Також система не повинна мати вихід у мережу Інтернет

На вітчизняному ринку аналогів створеній системі взагалі не виявлено, тому це є значимим плюсом для ідеї створення стартап проекту на дану не досить популярну тему.

5.2. Технологічний аудит ідеї проекту

Проведено аудит технології для проведення технічного аудиту ідеї проекту. за допомогою якої можна реалізувати ідею проекту. З аудите можна визначити чи доступні ці технології, яким чином можна вдосконалити технології [31]. Результат представлений у таблиці 5.3.

Таблиця 5.3. Технологічна здійсненність ідеї проекту

Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
Створенні системи моделювання процесу випромінювання звуку при русі вісесиметричних тіл в морському середовищі.	Середовище розробки Microsoft Visual Studio	+	Доступна
	Графічний плагін Zed-Graph	+	Доступна

Було обрано Microsoft Visual Studio, як середовище для розробки ідеї, а також плагін Zed-Graph для візуального відображення досліджень, зокрема для побудови графіків.

Обрані технології є доступними, не потребують допрацювань, а також безкоштовні та надають усі необхідні можливості для реалізації поставленої задачі. Робим висновок, що можна сміливо проводити далі дослідження.

5.3. Аналіз ринкових можливостей запуску стартап-проекту

Наступним кроком є проведення аналізу, щодо попиту. У цьому аналізі беруться до уваги ринкові можливості, а саме такі, які використовуються під час впровадження проекту. Також мають місце ринкові загрози [32]. Такий аналіз зазвичай скаже про ризики та недоліки. Результат досліджень наведено в таблиці 5.4.

Таблиця 5.4. Попередня характеристика потенційного ринку стартап-проекту

Показники стану ринку	Характеристика
Загальна потреба в продукції	Необхідна
Можливі річні обсяги випуску в натуральних показниках	До 10000 копій
Річні обсяги випуску в вартісних показниках	1млн $
Динаміка ринку (якісна оцінка)	Стрімке зростання
Наявність обмежень для входу	Для роботи слід приєднуватись до сервера
Специфічні вимоги до стандартизації та сертифікації	Відсутні
Середня норма рентабельності в галузі (або по ринку)	60 відсотків.

Провівши дослідження можемо констатувати, що ринок є задовільним і привабливим. До його можна сміливо входити. Бачимо, що показники попиту зростають. Дана технологія стає більш потрібна та актуальна серед кваліфікованих спеціалістів. Тому робимо висновок, що доцільно продовжити роботу в даному напрямі.

Як наслідком зростання попиту серед програмних продуктів, слід зробити більш зручну та автоматизовану систему для розробки програмного продукту та збільшити швидкість обчислення [33].

Надалі визначаються потенційні групи клієнтів, їх характеристики, та формується орієнтовний перелік вимог до товару для кожної групи (таблиця 5.5).

Таблиця 5.5. Характеристика потенційних клієнтів стартап-проекту

Потреба, що формує ринок	Цільова аудиторія	Особливості поведінки споживачів	Вимоги споживачів до товару
Визначення характер звуку, що випромінюються	Морські воєнні	Розробники займаються написанням програм, які не завжди відповідають заявленим вимогам, не завжди достатньо оптимізовані, що впливає на	доступна ціна; зручність і простота використання; мобільність
		подальше життя створених програмних продуктів – виникають проблеми, недоліки та конфлікти.	
	Інженер-риболов	Основною метою створити програмний продукт та випустити його на ринок та мати більший прибуток з використання	- зручність і простота використання

Слідом за цим проводиться аналіз, який визначає клієнтів чи групу клієнтів, які є потенційними користувачами продукту. Слід провести аналіз моментів, які допоможуть чи перешкодять запровадити проект [34].

Результати представлені у таблицях 5.6 та 5.7 відповідно. Результатами досліджень можна скористатись у подальшій роботі та подальшому дослідженні ринку ти проблеми, а також врахувати всі фактори, для якої і буде безпосередньо створюватись стартап проект

Таблиця 5.6. Фактори загроз

Фактор	Зміст загрози	Можлива реакція компанії
Поява конкурентів	Виникнення конкурентних проектів, які будуть фінансуватись компаніями-гігантами.	Оптимізація роботи, пришвидшення обчислень та додаткові можливості для користувача.
Зміни тенденцій ринку	Виникнення спеціалізованих програм на подібну тему	Адаптація програмного продукту, який буду містити в собі максимальну кількість рішень, які є актуальними на ринку.
Економічний спад	У клієнтів буде поступово падати попит	Акції, скидки на певний термін.
Зниження репутації компанії	Ситуація при якій клієнти перейдуть до більш успішної компанії.	Додаткова рекламна компанія. Покращення продукту.

Таблиця 5.7. Фактори можливостей

Фактор	Зміст загрози	Можлива реакція компанії
Невелика кількість конкурентів	Подібних програмних продуктів в нашій країні майже нема.	Піар свого проекту, реклама, презентації.
Відповідні тенденцій ринку	Наразі зважаючи на ситуацію у країні попитом користуються технології, пов’язані з воєнною тематикою.	Проведення зустрічей з клієнтами з презентаціями переваг проекту.
Можливість побудови власної репутації	Чиста компанія без минулого може привернути потенційних клієнтів.	Активна реклама. Натяк на чисту репутацію.

Наступним кроком є визначення загальних рис та видів конкуренції, які зараз існують а ринку. Слід визначити яка конкуренція буде очікувати та рівень конкуренції у товарів. Результати наведені у таблиці 5.8.

Даний аналіз є дуже важливим і в якійсь мірі визначальним у процесі опису та підготовці для стартап проекту, тому слід дуже ретельно ознайомитись з проблемою і обрати рішення.

Таблиця 5.8. Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	У чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії)
Тип конкуренції	Чиста Залежить від кількості конкурентів та якості	Пониження ціни продукту одночасно з ростом якості продукту є запорукою успіху

	надання ними послуг у порівнянні з послугами компанії	у даному випадку конкуренції.
За рівнем конкурентної боротьби	Локальна Конкуренція на вітчизняному ринку	Якщо конкурентних проектів на ринку немає, значить можна гратись з ціною на продукт.
За галузевою ознакою	Внутрішньогалузева Продукт націлений лише на конкретну сферу діяльності	Немає можливостей та сенсу розширювати функціонал за межі комп'ютерного моделювання.
Конкуренція за видами товарів	Марки-конкуренти Створений товар може мати конкурентів, які пропонують аналогічний товар	Зниження ціни, розширення функціональних, реклама для популяризації програмного продукту
За характером конкурентних переваг	Цінова Важливо за скільки продається товар, та його прибуток	Можливе підвищення ціни на нові розробки, зниження на старі версії для заохочення покупців
За інтенсивністю	Марочна Можуть з'являтись конкуренти	Реклама позитивних сторін в продукті, у випадку виходу на зовнішній ринок.

Після аналізу конкуренції проводиться більш детальний аналіз умов конкуренції в галузі (таблиця 5.9) — за моделлю п'яти сил М. Портера, який вирізняє п'ять основних факторів, що впливають на привабливість вибору ринку з огляду на характер конкуренції:

– конкурент, що вже є у галузі;
– потенційні конкуренти;
– наявність товарів-замінників;
– постачальники, що конкурують за ринкову владу;
– споживачі, які конкурують за ринкову владу.

Таблиця 5.9. Аналіз конкуренції в галузі за М.Портером

Складові галузі	Прямі конкуренти в галузі	Потенційні конкуренти	Клієнти	Товари-замінники
	Розробники аналогічних систем	Кращі продукти, ширший функціонал	Мають найбільше значення. Більш важлива їх кількість, ніж постійна співпраця	Немає
Висновки	Інтенсивність конкурентної боротьби з боку прямих конкурентів незначна	Наявні усі можливості входу на ринок. Потенційні конкуренти не виявлені. Строки виходу на ринок – один день	Необхідність клієнтської-бази, тому важливо знаходити можливості приваблення споживачів до власного продукту	Без обмежень

На основі аналізу конкуренції за М. Портером, проведеного у таблиці 5.9, а також із враховуючи характеристики ідеї проекту (таблиця 5.2), вимог споживачів до товару (таблиця 5.5) та факторів маркетингового середовища (таблиці 5.6 і 5.7) визначимо та обґрунтуємо перелік факторів конкурентоспроможності (таблиця 5.10).

Таблиця 5.10. Обґрунтування факторів конкурентоспроможності

Фактор конкурентоспроможності	Обґрунтування
Невелика кількість конкурентів на ринку	На початок розробки проекту не виявлено конкурентів на ринку в нашій країні.
Доступність створеного продукту (програмно)	Немає якихось рамок. Для застосування слід бути підключеним до мережі.
Легкість і простота використання	Якомога зрозуміліший інтерфейс. Присутня інструкція.
Підключення до мережі Інтернет	Слід підключитись до мережі, що б викачати базу.
Потреба у постійному супроводі	Відсутня
Додаткові компоненти	Наявність принтеру для друку проміжних результатів.

Провівши попередні дослідження можна визначити сильні та, навпаки, слабкі риси продукту. Вони наведені у таблиці 5.11.

Таблиця 5.11. Порівняльний аналіз сильних та слабких сторін проекту

Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів						
		–3	–2	–1	0	+1	+2	+3
Мала кількість / відсутність конкурентів	10				+			
Системні вимоги	20			+				
Простота використання	19	+						
Не потрібен супровід	11					+		

Фінальним етапом ринкового аналізу можливостей впровадження проекту є складання SWOT-аналізу (матриці аналізу сильних (Strength) та слабких (Weak) сторін, загроз (Troubles) та можливостей (Opportunities) на основі виділених ринкових загроз та можливостей, та сильних і слабких сторін (таблиця 5.12).

Таблиця 5.12. SWOT-аналіз проекту

Сильні сторони (S):	Слабкі сторони (W):
відсутність конкурентів; ініціативна розробка; зручне керування; гнучка політика керівництва;	мале фінансування; затрати на датчики;
Можливості (O):	Загрози (T):
вихід на міжнародні ринки; дослідження подібних технологій; вдосконалення функціоналу; можливість експорту даних;	незрозуміла тенденція попиту; поява конкурентів;

На основі SWOT-аналізу розробимо альтернативу ринкової поведінки для виведення стартап-проекту на ринок та орієнтовний оптимальний час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути виведені на ринок. Дані дослідження доцільно використати в розробленні ринкової стратегії проекту.

Визначені альтернативи аналізуються з точки зору строків та ймовірності отримання ресурсів (таблиця 5.13).

Таблиця 5.13. Альтернативи ринкового впровадження стартап-проекту

Альтернатива ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації

Міжнародний ринок	Пошук клієнтів	Пів року
Розширення функціоналу	Пошук інвесторів	Залежить від складності. Від пів року.

Отже, програмний продукт чи проект слід випускати у світ лише після того, як буде проведений детальний аналіз ринку, конкурентів, можливих ризиків з якими можна зіштовхнутись.

5.4. Розроблення ринкової стратегії проекту

Розглянемо ринкову стратегію, розробка якої визначає стратегію охоплення ринку. Вона містить в собі опис цільових груп потенційних клієнтів та споживачів, вони наведені у таблиці 5.14.

Таблиця 5.14. Вибір цільових груп потенційних споживачів

Опис цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в сегменті	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
Морські воєнні	Наявні	Існує попит	Замала	Незначна
Інженери-риболови	Наявні	Існує попит, проте менший у порівнянні з морськими воєнними	Замала	Помірна

Оскільки різниця між цільовими групами зовсім незначна, а також враховуючи той факт, що компанія має бажання почати продажі (а відповідно і отримання

прибутку) як найшвидше, то доцільно враховувати обидві цільові групи, тобто використовувати масовий маркетинг, пропонуючи стандартизовану програму [35].

Провівши аналіз, та визначивши потенційних споживачів автори ідеї проекту починають вибирати цільові групи. Саме для них і буде пропонуватись товар. Виходячи з аналізу буде визначена і вибрана потрібна стратегія.

Для роботи в обраних сегментах ринку необхідно сформувати базову стратегію розвитку, яка визначається у таблиці 5.15.

Таблиця 5.15. Визначення базової стратегії розвитку

Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
Вихід на нові ринки	Стратегія диференціації	Надання програмному продукту відмінних якостей, які роблять систему особливою на фоні аналогічних розробок	Стратегія диференціації
Розширення виробничої лінії	Стратегія диференціації (допускається стратегія спеціалізації)	Надання товару кращих властивостей та розширення функціоналу	Стратегія диференціації (допускається стратегія спеціалізації)

Наступним кроком є вибір стратегії, яка подана у таблиці 5.16. Це дослідження займає особливу роль в розробці проекту. Тому слід детально вивчити предмет досліжень.

Таблиця 5.16. Визначення базової конкурентної поведінки

Чи є проект новий на ринку	Так
Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Так
Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Ні
Стратегія конкурентної поведінки	Стратегія виклику лідера

На основі вимог споживачів з обраних сегментів до постачальника (стартап-компанії) та до програмного продукту, а також в залежності від обраної базової стратегії розвитку та стратегії конкурентної поведінки необхідно розробити стратегію позиціонування (таблиця 5.17), що полягає у формуванні ринкової позиції, за яким споживачі мають ідентифікувати торгівельну марку або проект [36].

Перелік ринкових загроз та ринкових можливостей складається на основі аналізу факторів загроз та факторів можливостей маркетингового середовища.

Таблиця 5.17. Визначення стратегії позиціонування

Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту
Доступна ціна, простота і зручність використання	Стратегія диференціації	Легкість і простота у використанні. Доступність через ціну та технічні характеристики. Вирішення важливих поставлених задач швидко, легко та	стандарти якості; метрики програмного забезпечення.

		зрозуміло навіть без інструкцій.	

Отже, робота стартап-компанії на ринку повинна бути спланована орієнтовано таким чином: за стратегією диференціації виконаний і буде поширюватись відмінний програмний продукт, дотримуючись у конкурентній поведінці стратегії «виклику лідера», тобто випускається один товар для усіх можливих споживачів.

5.5. Розроблення маркетингової програми стартап-проекту

Роздивимось процес розробки маркетингової програми нашого страртап-проекту. Слід визначити і сформувати маркетингові концепції товару. У таблиці 5.18 наведено дослідження та їх результати щодо конкурентоспроможності продукту [37]
.

Таблиця 5.18. Визначення ключових переваг концепції потенційного товару

Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами
Оцінка якості ПП	Оцінка за метриками. Удосконалення оцінки будь-якої з обраних характеристик.	Розрахункові показники, точність та достовірність яких можна оцінювати; самостійність програмної системи.

Надалі розроблена трирівнева маркетингова модель товару: уточняються ідея продукту, його фізичні складові та особливості процесу його надання (таблиця 5.19).

Визначення ринкових можливостей, які можна використати під час ринкового впровадження проекту, та ринкових загроз, які можуть перешкодити реалізації проекту, дозволяє спланувати напрями розвитку проекту із урахуванням стану ринкового середовища, потреб потенційних клієнтів та пропозицій проектів-конкурентів.

Таблиця 5.19. Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові
Товар за задумом	Створення системи моделювання процесу випромінювання звуку при русі вісесиметричних тіл в морському середовищі. Можливість задання вхідних параметрів, побудова графіків досліджень.
Товар у реальному виконанні	Властивості/характеристики
	Реалізовано систему моделювання моделювання процесу випромінювання звуку при русі вісесиметричних тіл в морському середовищі. Реалізовано графічне представлення результатів.
Товар із підкріпленням	До продажу: стандартна розроблена система (модуль для моделювання процесу випромінювання звуку при русі вісесиметричних тіл в морському середовищі)
	Після продажу: додані додаткові можливості

Розроблена математични модель, на якій базується програмна система, публікувалась лише у загальних рисах, а без математичної моделі цей ПП лише набір рядків коду. Але створення системи моделювання моделювання процесу випромінювання звуку при русі вісесиметричних тіл в морському середовищі не реалізовувалось раніше, а тому є необхідність у фіксуванні авторських прав або отриманні патенту.

Визначення цінових меж, якими необхідно керуватись при встановленні ціни на потенційний товар, яке передбачає аналіз ціни на товари-аналоги, а також аналіз рівня доходів цільової групи споживачів описано в таблиці 5.20.

Виявлено, що результати доцільно використати при подальшому дослідженні, а отже, зважаючи на непопулярність даної теми, слід продовжити дослідження й надалі розвивати ідею проекту.

Таблиця 5.20. Визначення меж встановлення ціни

Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни
20000 – 5000 $	500 – 1000 $	10 – 15 $

Наступним кроком є визначення оптимальної системи збуту, в межах якого приймається рішення (таблиця 5.21): чи потрібно проводити збут власними силами або залучати сторонніх посередників, вибір та обґрунтування оптимальної глибини каналу збуту, вибір та обґрунтування виду посередників.

Таблиця 5.21. Формування системи збуту

Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
Бажання отримати більше за менші гроші	Пошук клієнтської бази та продаж	Нульовий рівень: тільки виробник	Вертикальна маркетингова система

Оскільки різниця між цільовими групами зовсім незначна, а також враховуючи той факт, що компанія має бажання почати продажі (а відповідно і отримання прибутку) як найшвидше, то доцільно враховувати обидві цільові групи, тобто використовувати масовий маркетинг, пропонуючи стандартизовану програму.

Останньою складової маркетингової програми є розроблення концепції маркетингових комунікацій, що спирається на попередньо обрану основу для позиціонування, визначену специфіку поведінки клієнтів (таблиця 5.22).

Робимо висновок, що робота стартап-компанії на ринку повинна бути спланована орієнтовано таким чином: за стратегією диференціації виконаний і буде поширюватись відмінний програмний продукт.

Таблиця 5.22. Концепція маркетингових комунікацій

Поведінка цільових клієнтів	Канали комунікацій цільових клієнтів	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення
Бажання отримати більше за менші гроші	Будь-які	Низька ціна Легкий і простий у використанні продукт	Донести до користувача суть продукту, його якість, та залучити якомога більше зацікавлених клієнтів

Висновки до розділу 5

- Розроблений план побудови маркетингової компанії, який включає наступні кроки, як розробка програми, моніторинг ринку та пошук потенційних купців та користувачів, створюється кілька відмінних характеристик на основі власного продукту, він базується на побажаннях;
- розробка стратегії, спрямованої на всіх можливих конкурентів. Після цього ставляться цілі обійти конкурентів;
- переваги проекту заключаються в тому, що на ринку в нашій країні відсутні конкуренти, а значить конкуренції не виявлено. На разі попит на продукт великий, так як система може робити аналіз звуку, який випромінюються. Подібні проекти існують за кордоном, але аналоги не повністю відповідають вимогам, а також є дорогими;
- перспективи впровадження з огляду на потенційні групи користувачів, стан конкуренції та конкурентоспроможності проекту – прямі, і тільки доводять можливість впровадження, та не марну розробку створеного програмного продукту.

ВИСНОВКИ

За результатами виконання дипломної роботи можна зробити наступні висновки:

- розглянуто існуючі системи для програмної реалізації побудови графіків у середовищі програмування Microsoft Visual Studio, які містять набір інструментів для створення та редагування графіків;
- розширено знання плагіну ZedGraph, завдяки ZedGraph програміст має можливість автоматизувати певні деталі інтерфейсу, зокрема, масштабування та відображення на екранах, що мають різні розміри, підключення додаткових бібліотек для різних потреб, зокрема, побудови графіків;
- проведено моделювання траєкторії руху вихорових кілець, звукового тиску та його спектру при взаємодії локалізованих вихорових структур;
- проведено аналіз звуку, що розповсюджується за допомогою кореляційної функції;
- досліджено, що вихори генерують максимальне звукове поле в момент проходження одного з кілець скрізь інше, у момент, коли їх взаємний вплив максимальний;
- даний програмний продукт може бути використаний у різноманітних сферах діяльності та галузях, що досліджують поширення звукових полів у різних суцільних середовищах. Результати моделювання можуть надати певну допомогу в проблемі ідентифікації рухомих тіл.

Після ознайомлення з темою роботи та аналізу конкурентних продуктів, стало зрозуміло, що на ринку, особливо на національному, майже немає доступних програмних продуктів, які були б націлені на моделювання звукового поля, яке виникає внаслідок взаємодії тонких вісесиметричних структур.

Було вирішено розробити пакет програм, націлений на розв’язання даної проблеми за допомогою сучасних технологій та засобів збору інформації, яка є у вільному доступі.

Для виконання магістерської роботи було проведено аналіз вже існуючих рішень та інструментів, за допомогою яких можна виконати поставлені задачі.

При розробці програмного продукту було використано середу розробки Microsoft Visual Studio, плагін Resharper та графічний плагін Zed Graph, який дозволяє швидко та зручно графічно відобразити проведені дослідження та їх результати.

Було описано роботу з програмним продуктом. У розділі є інструкція роботи з програмою. Зазначено всі можливі функції, які може використовувати користувач програмного продукту.

Даний програмний продукт має суттєву перевагу над існуючими аналогами та конкурентами, які представлені на внутрішньому ринку. Продукт має шанси розвиватись, наповнюватись новим функціоналом, а отже було визначено основні маркетингові стратегії. Потенційними користувачами програмного продукту є морські воєнні і інженери-риболови.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Амосов А. А. Вычислительные методы для инженеров / А. А. Амосов, Ю. А. Дубинский, Н. П. Копченова. — М: Мир, 1998. – 644 с.
2. Бахвалов Н. С. Численные методы / Н. С. Бахвалов, Н. П. Жидков, Г. М. Кобельков. — Москва: Бином, 2001. — 363 с.
3. Боровков А. И. Компьютерный инжиниринг / А. И. Боровков. — СПб: Изд-во Политехн. ун-та, 2012. — 93 с.
4. Лойцянский Л. Г. Механика жидкости и газа / Л. Г. Лойцянский. — М: Дрофа, 2003. — 840 с.
5. Троелсен Э. Язык программирования C# и платформа .NET 4.5 / Эндрю Троелсен. — М: Вильямс, 2014. — 208 с.
6. Карпенко Д.І. Комп'ютерне моделювання процесу випромінювання звуку при русі вісесиметричних тіл в морському середовищі / Д.І. Карпенко // Сучасні проблеми наукового забезпечення енергетики, 2018. — 137 с.
7. Эдвардс Ч. Дифференциальное уравнение и проблема собственных значений / Ч. Эдвардс, Д. Пенни. — М: Вильямс, 2007. — 1104 с.
8. Dounis AI., Caraiscos C. Advanced control systems engineering energy in a building environment // Renewable Energy Reviews. – 2009. – V. 13(6-7). - p. 1246–1261.
9. Карпенко Д.І. Моделювання процесу випромінювання звуку при русі вісесиметричних тіл / Д.І. Карпенко // Інформаційне суспільство: технологічні аспекти становлення, 2018. — 33 с.
10. Шикин Е.В., Боресков А.В. Компьютерная графика. Создание реалистических изображений. Построение графиков — М.: Диалог-МИФИ, 1995. — 288 с.
11 Wikipedia. Інсталяція ПЗ [Electronic resource]. – Mode of access: https://uk.wikipedia.org/wiki/ Інсталяція_ПЗ.
12 Белоусов А. И., Ткачев С. Б. Дискретная математика. — М.: МГТУ, 2006. — С. 460-587.

13. Нейгл К. C# 5,0 и платформа .NET 4.5 для профессионалов/ Кристиан Нейл. — М: Вильямс, 2013. — 1440 с.
14. Просиз Д. Программирование для Microsoft .NET / Джеф Просиз. — М: Русская редакция, 2003. — 704 с.
15. Скит Д. C# для профессионалов: тонкости программирования, 3-е издание, новый перевод C# in Depth / Джонни Скит. — М: Вильямс, 2014. — 608 с.
16. Хейлсберг А., Торгерсен М., Вилтамут С., Голд П. Язык программирования для Microsoft .NET. Классика Computers Science, . — СПб: Питер, 2012. — 784 с.
17. Стилмен Э. Изучаем C# / Энтони Стилмен. — СПб: Питер, 2003. — 704 с.
18. Албахари Д. C# 5.0. Справочник. Полное описание языка/ Джозеф Албахари. — М: Вильямс, 2013. — 1008 с.
19. Уотсон К., Нейгел К., Язык программирования C#: полный курс. — М.: Диалектика, 2010. — 288 с.
20. Mickey Williams Microsoft Visual C# (Core Reference) – Microsoft Press Redmond, WA, USA, 2002. – 750 p.
21. Jeff Prosise Programming Microsoft .NET – Microsoft Press Redmond, WA, USA, 2002. – 800 p.
22. Иан Грэхем. Объектно-ориентированные методы. Принципы и практика = Object-Oriented Methods: Principles & Practice. — 3-е изд. — М.: «Вильямс», 2004. — С. 880.
23. Martin Fowler. [Електронний ресурс]. – Режим доступу: https://martinfowler.com/articles/microservices.html
24. Iserles A. A First Course in the Numerical Analysis of Differential Equations — Cambridge: Cambridge University Press, 1996. — 408 с.
25. Hairer E., Wanner G., Solving ordinary differential equations II: Stiff and differential-algebraic problems. — Berlin, New York: Springer-Verlag, 1996. — 738 с.
26. Süli E., Mayers D., An Introduction to Numerical Analysis. — Cambridge: Cambridge University Press, 2003. — 148 с.

27. Thompson G., Lordan M. A review of creativity principles applied to engineering design. Proceedings of the Institution of Mechanical Engineers // Part E: Journal of Process Mechanical Engineering, 1999. – V.213, №1. – P.17-31.
28. Чекмарев А. А. Начертательная геометрия и черчение: Учеб. для студ. высш. учеб. заведений. – 2-е изд., перераб. и.доп. – М.: Гуманит. изд. центр ВЛАДОС, 2002. – 472 с.
29. Samuel R. Buss 2D Computer Graphics: A Mathematical Introduction. – Cambridge University Press, 2003. – 397 p.
30. Черкасов Д. О., Сайбель Н. Ю. Стартап: характеристика понятия и этапы развития // Современное состояние и перспективы развития научной мысли: сборник статей Международной научно-практической конференции (25 мая 2015 г., г. Уфа) в 2 ч. – Уфа: АЭТЕРНА, 2015. – Ч. 1. – С.141-143.
31. Розроблення стартап-проекту: Методичні рекомендації до виконання розділу магістерських дисертацій для студентів інженерних спеціальностей / За заг. ред. О. А. Гавриша. – Київ: НТУУ «КПІ», 2016. – 28 с.
32. Харниш, В. Правила прибыльных стартапов : как расти и зарабатывать деньги / В. Харниш ; пер. с англ. В. Хозинского. – Москва : Манн, Иванов и Фербер, 2012. – 279 с.
33. Тиль, П. От нуля к единице : как создать стартап, который изменит будущее / П. Тиль, Б. Мастерс; перевод с англ. – Москва : Альпина паблишер, 2015. – 188 с.

    Петруненков А. Оценка коммерческой привлекательности проекта [Електронний ресурс].– Режим доступу: http://www.techbusiness.ru/tb/archiv/number2/page01.htm
34. Каталог автоматизированных систем [Електроний ресурс]. – Электрон. дан. - Режим доступа: http://www.erponline.ru/software/open/
35. Филип Котлер, Роланд Бергер, Нильс Бикхофф. Стратегический менеджмент по Котлеру. Лучшие приемы и методы = The Quintessence of Strategic

Management: What You Really Need to Know to Survive in Business. — М.: Альпина Паблишер, 2012. — 144 с.

36. Бариленко В. И. Бизнес-анализ как важный вид консалтинговых услуг // РИСК: Ресурсы, Информация, Снабжение, Конкуренция. — № 4. — 2012. — С.202-207.
37. Квашнин А. Как продвигать проекты коммерциализации технологий: серия методических материалов «Практические руководства для центров коммерциализации технологий» / М. Катешова, А. Квашнин, под рук. П. Линдхольма, проект EuropeAid «Наука и коммерциализация технологий», 2006. – 52 с.

ДОДАТОК А

Публікація

Комп'ютерне моделювання процесу випромінювання звуку при русі вісесиметричних тіл в морському середовищі

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ ТВ3260_18М

Аркушів 7

2018

ДОДАТОК Б

Акт впровадження

Комп'ютерне моделювання процесу випромінювання звуку при русі вісесиметричних тіл в морському середовищі

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ ТВ3260_18М

Аркушів 2

2018