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

Навчально-науковий інститут телекомунікаційних систем

Кафедра телекомунікацій

«На правах рукопису»
УДК _____

«До захисту допущено»
Завідувач кафедри
_____ Сергій КРАВЧУК
«___»_____2025 р.

Магістерська дисертація

на здобуття ступеня магістра
за освітньо-професійною програмою «Інженерія та програмування інфокомунікацій»
зі спеціальності 172 «Електронні комунікації та радіотехніка»

на тему: «Проєктування адаптивної антени для приймача систем глобальної супутникової радіонавігації»

Виконала:
студентка ІІ курсу, групи ЦК-41мп
Поворознік Марія Володимирівна _____

Керівник:
Доцент кафедри ТК НН ІТС, к.т.н.
Авдєєнко Гліб Леонідович _____

Консультант:
Доцент кафедри ТК НН ІТС, к.т.н.
Капштик Сергій Володимирович _____

Рецензент:
Доцент кафедри ІТТ, к.т.н., доцент
Правило Валерій Володимирович _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студентка _____

Київ – 2025 року

Національний технічний університет України

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Навчально-науковий інститут телекомунікаційних систем
Кафедра телекомунікацій

Рівень вищої освіти – другий (магістерський)

Спеціальність – 172 «Електронні комунікації та радіотехніка»

Освітньо-професійна програма «Інженерія та програмування інфокомунікацій»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Сергій КРАВЧУК

«___»_____2025 р.

ЗАВДАННЯ
на магістерську дисертацію студентці

Поворознік Марії Володимирівні

1. Тема дисертації «Проєктування адаптивної антени для приймача систем глобальної супутникової радіонавігації», науковий керівник дисертації Авдєєнко Гліб Леонідович, к.т.н., затверджені наказом по університету від «03» листопада 2025 р. № 4772-с.

2. Термін подання студентом дисертації 15.12.2025 р.

3. Об’єкт дослідження: Дослідження алгоритмів адаптивного придушення радіозавад системам супутникової навігації GPS.

4. Предмет дослідження: Характеристики діаграми спрямованості фазованих антенних решіток з подвійною поляризацією, адаптивні алгоритми формування вагових коефіцієнтів на основі модифікованого LMS з циклічним витоком.

5. Перелік завдань, які потрібно розробити:

1) Ознайомлення з теоретичними основами функціонування адаптивних антенних систем;

2) Дослідження структурних схем побудови адаптивних антенних систем;

3) Порівняльна характеристика основних алгоритмів адаптивного формування променів антенних решіток;

4) Імітаційне моделювання роботи адаптивної антенної решітки для систем супутникової радіонавігації GPS.

6. Орієнтовний перелік ілюстративного матеріалу:
Слайд №1 Постановка задачі дослідження: тема роботи, мета та завдання роботи, об'єкт дослідження, практична цінність.
Слайд №2 Фазована антенна решітка: ключові принципи, що використовуються в роботі.
Слайд №3 Адаптивна багатоканальна обробка сигналів у антенних решітках.
Слайд №4 Архітектура SDR-приймача для CRPA.
Слайд №5 Імітаційні GPS-моделі в середовищі MATLAB.
Слайд №6 Формування та робота моделі GPS-приймача в MATLAB.
Слайд №7 Розроблення та моделювання завад навігаційного сигналу.
Слайд №8 Розроблення адаптивної антенної решітки для подавлення завад.
Слайд №9 Стартап-проєкт на основі адаптивних CRPA-рішень.
Слайд №10 Висновки.

7. Консультанти розділів дисертації*

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Розділ №1	Доцент кафедри ТК НН ІТС, Капштик С.В.	24.01.2025	17.03.2025
Розділ №2	Доцент кафедри ТК НН ІТС, Капштик С.В.	18.03.2025	20.05.2025
Розділ №3	Доцент кафедри ТК НН ІТС, Капштик С.В.	21.05.2025	25.07.2025
Розділ №4	Доцент кафедри ТК НН ІТС, Капштик С.В.	26.07.2025	31.10.2025
Розділ №5	Доцент кафедри ТК НН ІТС, Капштик С.В.	01.11.2025	13.11.2025

8. Дата видачі завдання "01" вересня 2024 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1.	Визначення мети та постановка завдань дослідження	01.09.2024 – 15.09.2024	Виконано
2.	Підбір та ознайомлення з технічною літературою.	16.09.2024 – 18.11.2024	Виконано
3.	Аналіз існуючих рішень. Формулювання структури дослідження та проєктування.	19.11.2024 – 23.01.2025	Виконано
4.	Розділ 1: Функціонування фазованої антенної решітки	24.01.2025 – 17.03.2025	Виконано
5.	Розділ 2: Адаптивна багатоканальна обробка сигналів	18.03.2025 – 20.05.2025	Виконано
6.	Розділ 3: Архітектура програмно-	21.05.2025 – 25.07.2025	Виконано

* Якщо визначені консультанти. Консультантом не може бути зазначено наукового керівника магістерської дисертації.

	конфігурованого приймача та методи адаптивної обробки сигналів для CPRA-антени		
7.	Розділ 4: Моделювання адаптивної антени для приймача систем глобальної супутникової навігації	26.07.2025 – 31.10.2025	Виконано
8.	Розділ 5: Розроблення стартап-проєкту	01.11.2025 – 13.11.2025	Виконано
9.	Підготовка висновків, оформлення дипломної роботи.	14.11.2025 – 12.12.2025	Виконано
10.	Підготовка звітних та презентаційних матеріалів.	12.12.2025 – 22.12.2025	Виконано

Студентка Марія ПОВОРОЗНІК

Науковий керівник дисертації Гліб АВДЄЄНКО

РЕФЕРАТ

Робота містить 119 сторінок, 132 рисунки, 39 формул, 25 таблиць. Для написання було використано 11 джерел.

Актуальність теми обумовлена тим, що традиційні одноантенні GNSS-приймачі не здатні ефективно протидіяти потужним завадам, тоді як адаптивні антенні решітки дозволяють формувати просторові нулі діаграми спрямованості у напрямках джерел завад, забезпечуючи безперервну роботу навігаційного обладнання.

Зв'язок роботи з науковими програмами, планами, темами: Дисертаційна робота виконана в рамках науково-дослідної роботи кафедри телекомунікацій нн ІТС, що спрямована на розробку перспективних методів та засобів підвищення завадозахищеності систем супутникової навігації.

Мета роботи: Розроблення моделі адаптивної антенної решітки (CRPA), що забезпечує просторово-часову селекцію радіосигналів систем супутникової радіонавігації GPS в умовах впливу навмисних радіозавад.

Завдання дослідження:

1) Ознайомлення з теоретичними основами функціонування адаптивних антенних систем;

2) Дослідження структурних схем побудови адаптивних антенних систем;

3) Порівняльна характеристика основних алгоритмів адаптивного формування променів антенних решіток;

4) Імітаційне моделювання роботи адаптивної антенної решітки для систем супутникової радіонавігації GPS.

Об'єкт дослідження: Дослідження алгоритмів адаптивного придушення радіозавад системам супутникової навігації GPS.

Предмет дослідження: характеристики діаграми спрямованості фазованих антенних решіток з подвійною поляризацією, адаптивні алгоритми формування вагових коефіцієнтів на основі модифікованого LMS з циклічним витоком.

Методи дослідження: методи теорії антен та електродинаміки, методи цифрової обробки сигналів та адаптивної фільтрації, методи комп'ютерного моделювання у середовищі MATLAB

Наукова новизна одержаних результатів: полягає у комплексному підході до проєктування адаптивної антенної системи, що поєднує оптимізацію геометричних параметрів фазованої решітки, розробку модифікованих адаптивних алгоритмів обробки сигналів та архітектуру програмно-конфігурованого приймача.

Практичне значення одержаних результатів: визначається можливістю безпосереднього впровадження розроблених методів та алгоритмів у проєктування комерційних GNSS-приймачів нового покоління з підвищеною стійкістю до завад. У сфері автономного транспорту, в умовах щільної міської забудови та багатопроменевого поширення сигналів. Результати дослідження можуть застосовуватися в об’єктах критичної інфраструктури для високоточної часової синхронізації, у геодезії та картографії, а також у наукових дослідженнях і навчальному процесі.

Ключові слова: Адаптивна антена, фазована антенна решітка, GNSS, GPS, супутникова навігація, придушення завад, CRPA, MATLAB-моделювання, захист від глушіння, навігаційна точність.

ABSTRACT

The work contains 119 pages, 132 figures, 39 formulas, and 25 tables. 11 sources were used in its preparation.

The relevance of the topic is due to the fact that traditional single-antenna GNSS receivers are unable to effectively counteract powerful interference, while adaptive antenna arrays allow the formation of spatial nulls in the directional pattern in the directions of interference sources, ensuring the continuous operation of navigation equipment.

Connection of the work with scientific programs, plans, topics: The dissertation was completed as part of the research work of the Department of Telecommunications of the Institute of Telecommunications Systems, which is aimed at developing promising methods and means of improving the interference immunity of satellite navigation systems.

The purpose of diploma: Development of a model of a adaptive antenna array (CRPA) that provides spatial and temporal selection of radio signals from GPS satellite radio navigation systems under conditions of intentional radio interference.

Research objectives:

1) Familiarization with the theoretical foundations of adaptive antenna systems;

2) To study the structural diagrams of adaptive antenna systems;

3) To compare the main algorithms of adaptive beamforming in antenna arrays;

4) To simulate the operation of an adaptive antenna array for GPS satellite navigation systems.

Research subject: Investigation of algorithms for adaptive suppression of radio interference in GPS satellite navigation systems.

Subject of research: characteristics of the directional pattern of phased antenna arrays with dual polarization, adaptive algorithms for forming weight coefficients based on modified LMS with cyclic leakage.

Research methods: methods of antenna theory and electrodynamics, methods of digital signal processing and adaptive filtering, methods of computer modeling in the MATLAB environment

Scientific novelty of the results obtained: consists in a comprehensive approach to the design of an adaptive antenna system that combines the optimization of the geometric parameters of the phased array, the development of modified adaptive signal processing algorithms, and the architecture of a software-configurable receiver.

Practical significance of the results obtained: determined by the possibility of direct implementation of the developed methods and algorithms in the design of new-generation commercial GNSS receivers with increased resistance to interference. In the field of autonomous transport, in conditions of dense urban development and multi-beam signal propagation. The research results can be applied in critical infrastructure facilities for high-precision time synchronization, in geodesy and cartography, as well as in scientific research and the educational process.

Keywords: Adaptive antenna, phased antenna array, GNSS, GPS, satellite navigation, interference suppression, CRPA, MATLAB modeling, jamming protection, navigation accuracy.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

CNAV	Civil Navigation
CPU	Central Processing Unit
CRPA	Controlled Reception Pattern Antenna
CU	Compute Unit
DLL	Delay-Locked Loop
FLL	Frequency-Locked Loop
FNBW	First Null BeamWidth
FPGA	Field-Programmable Gate Array
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GPU	Graphics Processing Unit
HPBW	Half-Power BeamWidth
HT	Hilbert Transform
LDS	Local Data Share
LHCP	Left-Hand Circular Polarization
LMS	Least Mean Squares
LNAV	Legacy Navigation
MVDR	Minimum Variance Distortionless Response
PLL	Phase-Locked Loop
PRN	Pseudo Random Noise
RHCP	Right-Hand Circular Polarization
RINEX	Receiver Independent Exchange format
SDR	Software-Defined Radio
SNR	Signal-to-Noise Ratio
VRAM	Video Random Access Memory
ГНСС	Глобальна Навігаційна Супутникова Система
ПЗ	Пропускна здатність

ВСТУП

Актуальність роботи: обумовлена зростаючими вимогами до надійності ГНСС-систем у критичних застосуваннях та необхідністю захисту від навмисних та ненавмисних завад, що підтверджує значення роботи як для розвитку теорії адаптивних антенних систем, так і для практичного впровадження у сферах авіації, оборони, транспорту та телекомунікацій.

Мета роботи: розроблення моделі адаптивної антенної решітки (CRPA), що забезпечує просторово-часову селекцію радіосигналів систем супутникової радіонавігації GPS в умовах впливу навмисних радіозавад.

Об’єкт дослідження: Дослідження алгоритмів адаптивного придушення радіозавад системам супутникової навігації GPS

Методи дослідження: методи теорії антен і поширення радіохвиль, цифрової обробки сигналів, теорії адаптивних систем та статистичної радіотехніки, аналітичні методи розрахунку діаграм спрямованості та методи просторової фільтрації. Адаптивна багатоканальна обробка сигналів досліджувалася з використанням алгоритмів LMS та їх модифікацій. Моделювання роботи GNSS-приймача з адаптивною антеною та різними типами завад виконано в середовищі MATLAB.

Практична цінність: полягає у можливості використання отриманих результатів для проєктування завадостійких ГНСС-приймачів з адаптивними антенними решітками. Може бути застосовано в навігаційних системах оборонного та цивільного призначення, автономному транспорті, авіаційних і морських навігаційних комплексах, а також у системах критичної інфраструктури. Розроблена MATLAB-модель може використовуватися для інженерного проєктування, тестування та навчальних цілей.

РОЗДІЛ 1
ФУНКЦІОНУВАННЯ ФАЗОВАНОЇ АНТЕННОЇ РЕШТІКИ

1.1. Принцип роботи фазованої антеної решітки з використанням зсуву фаз

Для фазованої антенної решітки, (рис 1.1.) вісь візування $\theta = 0°$ спрямована перпендикулярно до лицьової поверхні антени. При цьому значення кута θ вважається позитивним, якщо відхилення відбувається праворуч від осі візування, і негативним – у випадку відхилення ліворуч. [1]

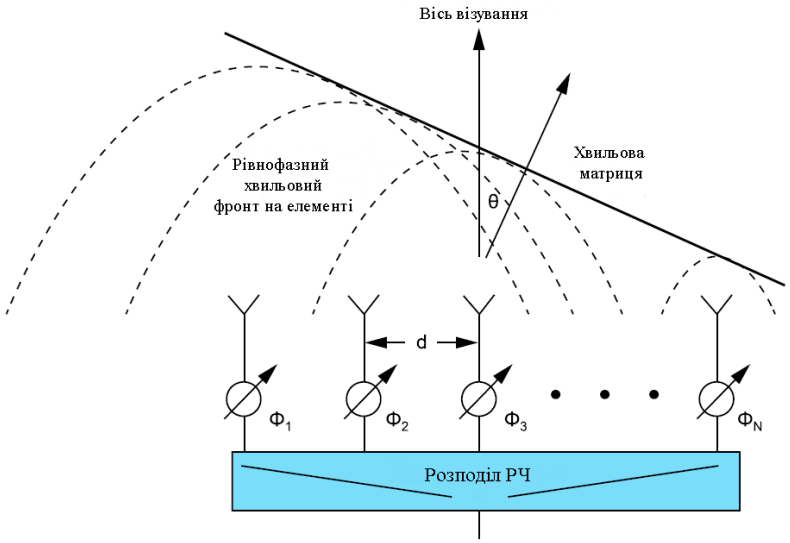

Рис. 1.1. Принцип роботи фазованої антенної решітки з використанням зсуву фаз [1]

Розрахунок фазового зсуву $\Delta\Phi$ залежно від кута повороту променю показано на Рис 1.2. На Рис.1.2.а) наведено геометричне співвідношення між сусідніми елементами антени, де d – крок антенної решітки. Промінь може бути відхилений від осі візування на кут θ або ж на кут φ, якщо вважати відхилення від умовної горизонтальної лінії. З Рис.1.2.б) видно, що виконується співвідношення $\theta + \varphi = 90°$. Це співвідношення дозволяє визнати величину L –

відстань, на яку зміщується хвильовий фронт під час поширення, при чому $L = dsin(\theta)$. Відповідно, тимчасова затримка дорівнює часу, необхідному для проходження хвилею відстані L. Якщо прийняти L за довжину хвилі, яка відповідає визначенню, то тимчасова затримка буде вирахувана за формулою $\Delta t = L/c$, де $c = 3 \cdot 10^8$ м/с. Тоді відповідно з малюнком 1.2.в) величина фазового зсуву буде визначатись за формулою:

$$\Delta\Phi = (2\pi dsin\theta)/\lambda \qquad [1]\ (1.1)$$

Рівняння фазового зсуву, при крокові антенної решітки, що становить рівно половину довжини хвилі сигналу матиме вигляд [1]:

$$\Delta\Phi = \pi sin\theta, \text{ при } d = \lambda/2 \qquad [1]\ (1.2)$$

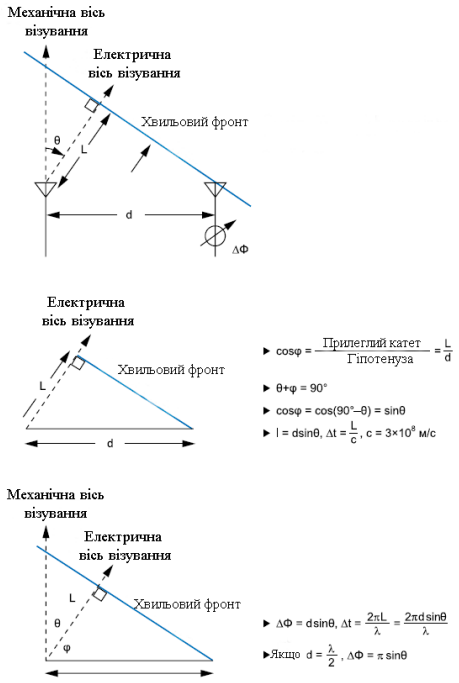

Рис.1.2. Розрахунок фазового зсуву ΔΦ залежно від кута повороту променю [1]

На Рис.1.3. подано графіки, побудовані за формулою (1.1.) для різних співвідношень d/λ. При $d = \lambda/2$, графік проявляє нахил 3:1 у зоні поблизу осі

візування. Це спричинено множником π, як зазначено у формулі (1.2.). Цей графік також ілюструє формування теоретичного відхилення променю від осі візування на 90°, фазовим зсувом між елементами у 180°. Такий випадок вважається ідеальним та неможливим у реальних умовах. При $d > \lambda/2$, незалежно від величини фазового зсуву досягти відхилення променю на 90° не можливо.

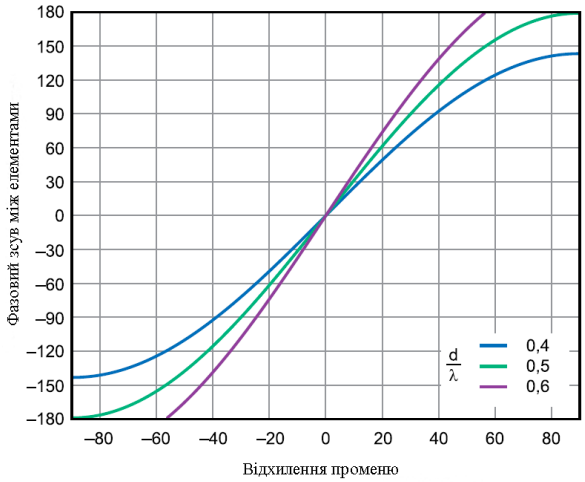

Рис.1.3. Фазовий зсув ΔФ залежно від кута повороту променю для трьох випадків співвідношення d/λ. [1]

1.2. Посилення, спрямованість і апертура антени

Спрямованість антени визначає здатність фокусувати енергію в одному напрямку. Спрямованість антени – це порівняння максимальної виміряної потужності P_{max} в певному напрямку, із середньою потужністю P_{av} у всіх напрямках. У випадку невизначеності напрямку, спрямованість обчислюється за формулою:

$$D = P_{max}/P_{av} \qquad [1]\ (1.3)$$

Підсилення розраховується за тією ж самою формулою, що й спрямованість, однак ще додається коефіцієнт втрат:

$$G = kD \qquad [1]\ (1.4)$$

Де, $k = P_{rad}/P_{in}$, P_{rad} – загальна випромінювана потужність, P_{in} – вхідна потужність антени, k – коефіцієнт втрат.

Діаграма спрямованості антени у тривимірному просторі матиме вигляд, як на Рис.1.4.

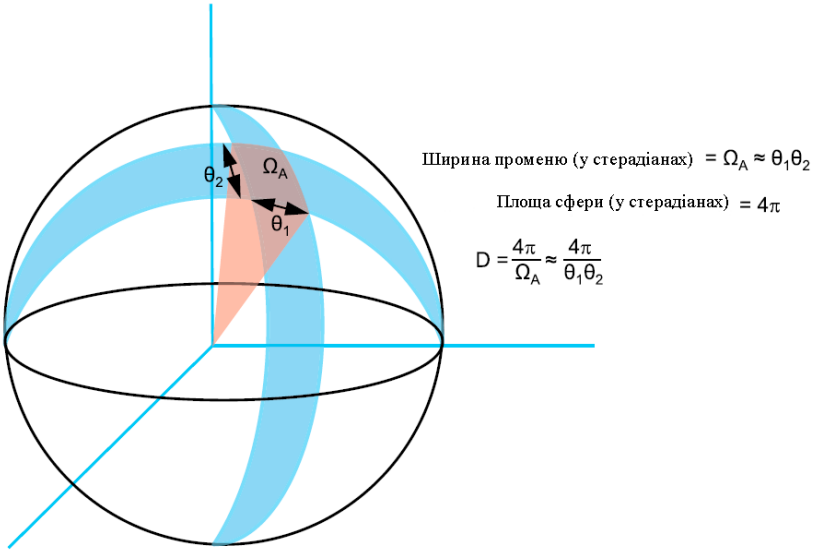

Рис.1.4. Діаграма спрямованості антени, спроектована на сферу [1]

Поверхня сфери, яка спостерігається з центру, утворює тілесний кут у 4π стерадіан, при цьому сама площа поверхні дорівнює $4\pi^2$. Таким чином, щільність потужності ізотропної антени можна визначити як:

$$P_{rad}/4\pi r^2 \qquad [1]\ (1.5)$$

На сфері існують два кути положення, що в радіолокаційних системах позначають як азимут і кут місця. Ширину променю можна описати функцією кутів напрямку θ_1 і θ_2, які формують на сфері область Ω_A – просторову ширину променю в стерадіанах. Приблизно значення Ω_A можна подати у вигляді $\Omega_A \approx \theta_1 \times \theta_2$. [1]

Оскільки область Ω_A визначається на сфері кутами напрямку, то з цим підходом спрямованість антени обчислюється за формулою:

$$D = 4\pi/\Omega_A \approx 4\pi/(\theta_1 \times \theta_2) \quad [1] (1.6)$$

Апертура антени трактується як частина сферичної поверхні, що призначена для приймання електромагнітних хвиль. Розмір апертури безпосередньо залежить від довжини хвилі. Для ізотропної антени апертура визначається за формулою [1]:

$$A_{isotropic} = \lambda^2/4\pi \quad [1] (1.7)$$

Крім того, дійсна апертура антени визначається не лише довжиною хвилі, а й коефіцієнтом посилення, і виражається формулою:

$$A_e = G\lambda^2/4\pi \quad [1] (1.8)$$

Якщо поєднати всі наведені складові, то стає очевидно: посилення антени слід розглядати як функцію кута, що задає діаграму спрямованості, при цьому враховуючи ефективність чи втрати в антені.

1.3. Множник антенної решітки

Щоб отримати повне посилення, необхідно врахувати два параметри: коефіцієнт підсилення кожного окремого елемента решітки G_E (множник елемента), а також вплив формування променю за допомогою всіх елементів решітки G_A (множник масиву елементів). Таким чином, повне посилення антенної решітки визначається за формулою:

$$G(\theta) = G_E(\theta) + G_A(\theta) \qquad [1]\ (1.9)$$

Множник елементу G_E – діаграма спрямованості одного окремого елементу решітки. Цей множник не змінюється залежно від умов роботи чи електричних характеристик, але його визначають геометрія і конструктивні особливості антени. Його врахування є особливо важливим, адже саме він може обмежувати посилення всієї антенної решітки. G_E доцільно розглядати як певну константу, яка, однак, безпосередньо впливає на повне посилення фазованої решітки. [1]

Множник решітки G_A обчислюється на основі геометрії решітки (d для еквідистантно-лінійної структури) та «ваги» променю, що задаються амплітудою та фазою.

Для того, щоб визначити зміну посилення решітки, потрібно побудувати графік залежності нормалізованого множника від одиничного посилення. Нормалізований множник решітки визначається за формулою:

$$AF[\theta] = \frac{\sin\left(\frac{N\pi d}{\lambda}[\sin(\theta)-\sin(\theta_0)]\right)}{N\sin(\frac{\pi d}{\lambda}[\sin(\theta)-\sin(\theta_0)])} \qquad [1]\ (1.9)$$

Якщо взяти до уваги, що θ_0 – функція фазового зсуву між елементами, то формула нормалізованого множника решітки буде мати вигляд:

$$AF[\theta, \Delta\Phi] = \frac{\sin\left(N\left[\frac{\pi d}{\lambda}\sin(\theta)-\frac{\Delta\Phi}{2}\right]\right)}{N\sin(\left[\frac{\pi d}{\lambda}\sin(\theta)-\frac{\Delta\Phi}{2}\right])} \qquad [1]\ (1.10)$$

Для того щоб дане рівняння (1.10) могло бути використане в реальній системі, повинні виконуватись ці умови [1]:

1) Елементи решітки повинні бути розташовані на однаковій відстані один від одного;
2) Фазовий зсув між усіма елементами повинен бути рівномірним;
3) Амплітуди окремих елементів повинні залишатись однаковими.

Застосувавши отримані рівняння, можна побудувати графіки множника решітки для різних розмірів антенних структур. Рис. 1.5. та Рис. 1.6.

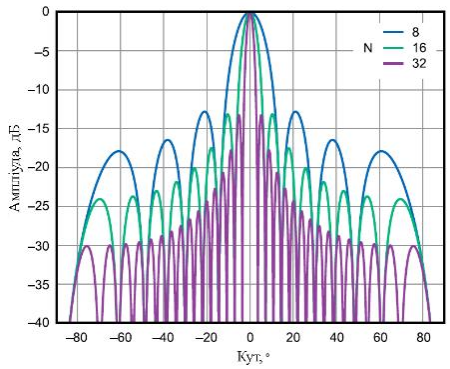

Рис.1.5. Графік функції нормалізованого множника лінійної решітки з кроком елементів $d = \lambda/2$ і кількістю елементів 8, 16 і 32 відповідно [1]

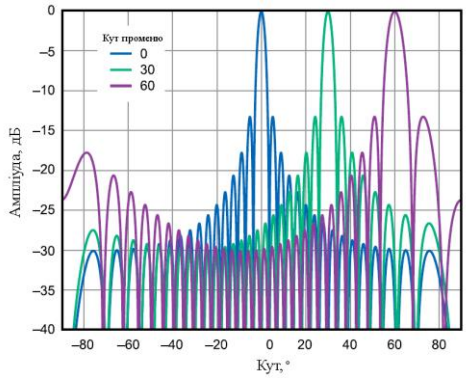

Рис.1.6. Графік функції нормалізованого множника лінійної решітки з кроком елементів $d = \lambda/2$ і кількістю елементів 32 при різних кутах променю [1]

Аналізуючи дані, подані на Рис.1.5. та Рис. 1.6. робимо такі висновки [1]:

1) Амплітуда першого бічного пелюстка відносно осі візування становить -13 дБ і не залежить від кількості елементів, причиною цього є функція синуса, присутня в рівнянні множника решітки. Поступово підвищуючи коефіцієнт підсилення елементів можна збільшити амплітуду пелюсток.
2) Зі зростанням числа елементів решітки ширина головного променю зменшується.
3) При відхиленні променю від осі візування його ширина збільшується.

4) Збільшення кількості елементів решітки призводить до зростанню кількості променів.

1.4. Ширина променю

Ширина променю виступає показником кутової роздільної здатності антен. Найчастіше її визначають або за шириною променю половинної потужності (HPBW) або за шириною першого нульового променю (FNBW).

Спираючись на рівняння (1.10) можна розрахувати точне значення HPBW, прирівнявши нормалізований множник до рівня половинної потужності (3 дБ або $1/\sqrt{2}$). Якщо $\theta = 0°$, $N = 8$ та $d = \lambda/2$, тоді:

$$\frac{1}{\sqrt{2}} = \frac{\sin\left(8\left[\frac{\pi d}{2\lambda}\sin(\theta) - \frac{\Delta\Phi}{2}\right]\right)}{8\sin\left(\left[\frac{\pi d}{2\lambda}\sin(\theta) - \frac{\Delta\Phi}{2}\right]\right)} \quad [1]\ (1.11)$$

Із рівняння $\Delta\Phi$ випливає значення 0,35 рад. Підставивши його у формулу (1.1), можна визначити кут θ:

$$0{,}35 = (2\pi d \sin\theta)\frac{}{\lambda} \rightarrow \theta = 0{,}11 \text{ рад} = 6{,}4\,° \quad [1]\ (1.12)$$

У цьому випадку θ відповідає піковому значенню до точки 3 дБ, тобто половині HPBW. Для знаходження остаточного значення HPBW потрібно подвоїти цей результат, що дає HPBW $= 12{,}8\,°$. Якщо виконати аналогічний розрахунок для множника решітки, який дорівнює нулю, то отримаємо ширину першого нульового променю FNBW$= 28{,}5°$. [1]

Для еквідистантних лінійних антенних решіток апроксимацію HPBW подають у вигляді такої формули:

$$\theta_B \sim 0{,}886\lambda/Nd\cos\theta \quad [1]\ (1.13)$$

На Рис.1.7. наведено залежність ширини променю від кута відхилення для решіток різних розмірів при між елементній відстані $\lambda/2$.

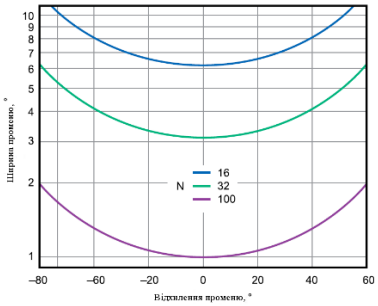

Рис.1.7. Залежність ширини променю від кута відхилення при відстані $\lambda/2$ та кількості елементів 16, 32 і 100 відповідно [1]

Аналізуючи графік, можна виділити кілька аспектів, що стосуються розмірності решіток:

1) Щоб досягти точності керування променем на рівні 1°, потрібно щонайменше 100 елементів. Якщо така точність потрібна одночасно по азимуту й куту місця, решітка повинна містини 10 000 елементів. При цьому точність 1° можлива лише по осі візування за умов, близьких до ідеальних. Для підтримки такої ж точності в польових умовах потрібне ще більше збільшення кількості елементів. [1]
2) Решітки, що складаються з 1000 елементів, мають досить широке практичне застосування. Прикладом може бути квадратна решітка зі стороною у 32 елементи, що дає сумарно 1024 елементи. Подібні решітки здатні забезпечити точність керування променем близько 4° поблизу осі візування.
3) Серійні решітки з 256 елементів, відносно доступні за ціною, можуть досягати точності наведення менше 10°, що робить їх прийнятними для широкого спектру застосувань.
4) Варто зазначити, що у всіх описаних випадках ширина променю подвоюється при відхиленні на 60°. Це пояснюється наявністю $\cos\theta$ у

знаменнику та зміною кута падіння фронту сигналу на елементи: якщо дивитись на решітку під кутом, її поперечний розмір візуально зменшується.

1.5. Об’єднання множників елемента та решітки

На Рис.1.8. наведено приклад, де множник елемента $G_E(\theta)$ описується косинусом. Косинусний спад часто трапляється під час аналізу фазованих решіток і легко піддається візуалізації. Біля осі візування площа графіку досягає максимального значення, поступово зменшуючись і зменшуючись зі збільшенням кута відповідно до закону косинуса.

Множник решітки $G_A(\theta)$ розглядався для лінійної решітки, що складається з 16 елементів з інтервалом $\lambda/2$ та однорідною діаграмою спрямованості. Сукупний графік, показаний синім кольором на Рис.1.8. є результатом перемноження множника елемента і множника решітки, тобто підсумовування їхніх графіків у шкалі дБ. [1]

Провівши аналіз отриманого графіку та відхилення променю від осі візування, можна сформувати такі висновки:

1. Амплітуда сумарного графіку зменшується відповідно до форми графіку множника елементу;
2. Бічні пелюстки, що знаходиться симетрично відносно осі візування, не втрачають своєї амплітуди;
3. Поза віссю візування спостерігається погіршення характеристик бічних пелюсток елементів решітки.

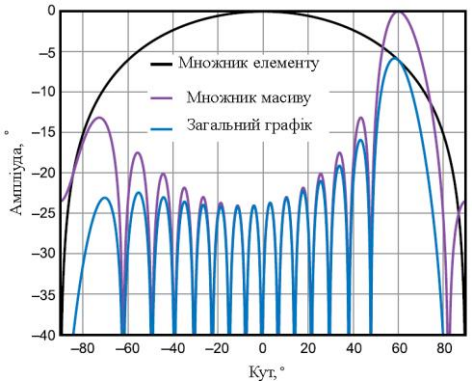

Рис.1.8. Множник елементу і множник решітки утворюють загальну діаграму спрямованості антени [1]

1.6. Розрахунок положення пелюсток решітки

Виходячи з формули (1.1) кут відхилення променю буде:

$$\theta = arc\sin(\Delta\Phi/2\pi \times (\lambda/d)) \qquad [2]\ (1.14)$$

Слід звернути увагу, що функція $arc\sin$ повертає правильні значення лише в межах $-1 \ldots + 1$. Якщо ж аргумент виходить за межі цього інтервалу, розв'язання стає неможливим. Важливо також зазначити, що фазовий зсув у рівнянні (1.14) є періодичною величиною, що повторюється кожні 2π. Отже, $\Delta\Phi$ може бути замінена виразом $(m \times 2\pi + \Delta\Phi)$, що дозволяє записати рівняння у наступному вигляді:

$$\theta = arc\sin((m \times 2\pi + \Delta\Phi)/2\pi \times (\lambda/d)) \qquad [2]\ (1.15)$$

Де $m = 0, \pm 1, \pm 2$ і т. д.

Щоб уникнути появи небажаних пелюсток решітки, необхідно дотримуватись такого обмеження:

$$|(m \times 2\pi + \Delta\Phi)/2\pi \times (\lambda/d)| > 1 \text{ для всіх } m \geq 1 \qquad [2]\ (1.16)$$

Прийнявши дане обмеження, ми приходимо до ситуації, коли пелюстки решітки з номерами $m = \pm 1, \pm 2$ тощо, дають некоректні значення функції $arc\sin$ і відповідно, можуть бути вилучені з діаграми. Водночас, значення m, що потрапляють у діапазон $0 \dots 1$, залишаються коректними для функції $arc\sin$, і саме тому в діаграмі все ж з'являється кілька додаткових пелюсток решітки Рис.1.9.

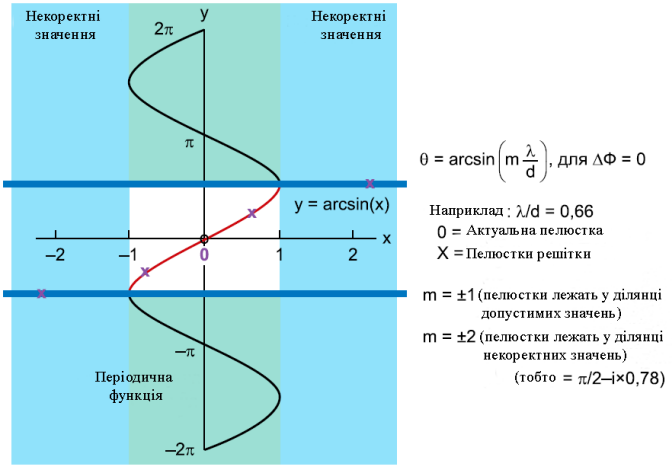

Рис.1.9. Область значення функції $arc\sin$ та пелюстки решітки [2]

1.7. Вплив кроку елементів на виникнення пелюсток решітки

Для зменшення кількості пелюсток решітки потрібно дотримуватися певного обмеження – формула (1.16.). Завдяки цьому ж виразу можна обчислити положення першої пелюстки решітки при $m = \pm 1$. Якщо в нести відповідні зміни та скористатися формулою (1.1.), то отримаємо наступний результат:

$$\frac{\pm 1 \times 2\pi + \frac{2\pi d_{max} \sin \theta_{max}}{\lambda}}{2\pi} \times \frac{\lambda}{d_{max}} = 1 \qquad [2]\ (1.17)$$

Після скорочення даного рівняння отримуємо:

$$\pm\lambda + d_{max} \sin\theta_{max} = d_{max} \qquad [2]\ (1.18)$$

Таким чином значення d_{max} визначається за формулою:

$$d_{max} = \lambda/(1 + |\sin\theta_{max}|), \text{для } \theta_{max} \text{ від 0 до } \pm\pi/2 \qquad [2]\ (1.18)$$

Отримане значення d_{max} забезпечує відсутність пелюсток решітки в діапазоні $\theta_{max} > \pi/2$.

Отже, обмеження максимально допустимого кута сканування відкриє можливість збільшувати відстань між елементами, а також канал і апертуру антенної решітки.

Згідно формули (1.15.) максимальна відстань між елементами може дорівнювати довжині хвилі навіть при нульовому куті відхилення променю. Це може бути застосовано коли неприпустима поява пелюсток решітки у видимій півкулі. У випадку, коли пелюстки решітки не доходять до поверхні Землі, допустимо використовувати відстань між елементами, що перевищує довжину хвилі, що призводить до звуження променю.

1.8. Зсув променю

Тимчасова затримка фактично є лінійним зсувом фази залежно від частоти. Звідси випливає: для конкретного напрямку променю фазовий зсув змінюється зі зміною частоти, а у зворотному випадку – при фіксованому фазовому зсуві, напрямок променю стає функцією частоти. Зсув променю – зміна кута променю залежно від частоти.

Слід враховувати, що фазовий зсув між елементами в точці візування $\theta = 0$ дорівнює нулю, тому зсув променю не виникає. Оскільки дане явище безпосередньо пов’язане з фазовими зсувами, воно залежить і від кута

відхилення θ і від зміни робочої частоти. Рівень зсуву променю може бути визначений за допомогою формул (1.1.) (1.14.):

$$\Delta\theta = arc\sin(f_0/f)\sin\theta_0) - \theta_0 \qquad [2]\ (1.19)$$

Графік даного рівняння поданий на Рис.1.10. При цьому замість відношення f_0/f використано співвідношення f/f_0, яке надає простіший спосіб візуалізації відхилення відносно центральної частоти.

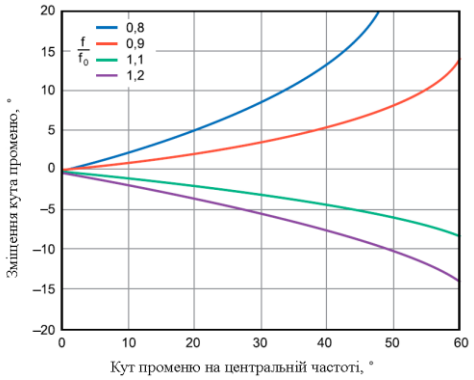

Рис.1.10. Зміщення променю в залежності від кута відхилення в центральній частоті [2]

Аналізуючи даний графік можна зробити такі висновки щодо зсуву:

1) Величина зміщення кута зростає зі збільшенням відхилення променю від осі візування;
2) Частоти, що лежать нижче центральної, викликають більший зсув, ніж ті, що вищі за неї;
3) При частотах нижчих за центральну відхилення променю від осі візування є більшим. [2]

1.8.1. Вирішення проблеми зсуву променю

У випадку формування діаграми спрямованості з використанням цифрових компонентів часову затримку можна реалізувати засобами логіки центрального процесора. Таким чином, створення фазованої решітки на базі цифрових елементів управління дозволяє усунути проблему зсуву променю, водночас забезпечуючи гнучкість завдяки програмній зміні параметрів затримки.

У гібридних системах управління застосовується інший підхід: аналогове керування призначене для окремих елементів решітки, тоді як цифрова обробка використовується для всього масиву загалом. Використання цього принципу дозволяє значно зменшити зміщення променю, тому він заслуговує особливої уваги при реалізації практичних проектів. Величина зсуву залежить виключно від тих складових решітки, які управляються аналоговим способом.

Гібридна архітектура надає можливість комбінувати аналогові фазообертачі для основного управління з цифровими елементами, що виконують подальшу обробку та коригування зсуву.

Висновки

У першому розділі було розглянуто функціонування фазованої антенної решітки. Було описано принцип роботи фазованої антенної решітки з використанням зсуву фаз, введено відповідні формули та побудовані графіки на їх основі. Також були введені поняття посилення, спрямованості і апертури антени. Було зроблено висновки, як на роботу антени впливає кількість елементів, відстань між елементами та кут променю за допомогою введення таких понять, як множник решітки та множник елементу. Також було описано основні фактори, що впливають на появу пелюсток решітки. Було розглянуто проблему зсуву променю та варіанти вирішення даної проблеми.

РОЗДІЛ 2
АДАПТИВНА БАГАТОКАНАЛЬНА ОБРОБКА СИГНАЛІВ

2.1. Структура та принципи формування вагових коефіцієнтів антенної решітки з подвійною поляризацією

2.1.1. Архітектура системи

Система починається з поляриметричної адаптивної антенної решітки, розташованої перед GPS-приймачем. Кожен елемент решітки є патч-антеною з подвійною поляризацією, що має два виходи, які відповідають горизонтальній (H) і вертикальній (V) поляризації прийнятого сигналу. I позначає дійсну частину комплексного сигналу, а Q позначає уявну частину комплексного сигналу. Рис.2.1.

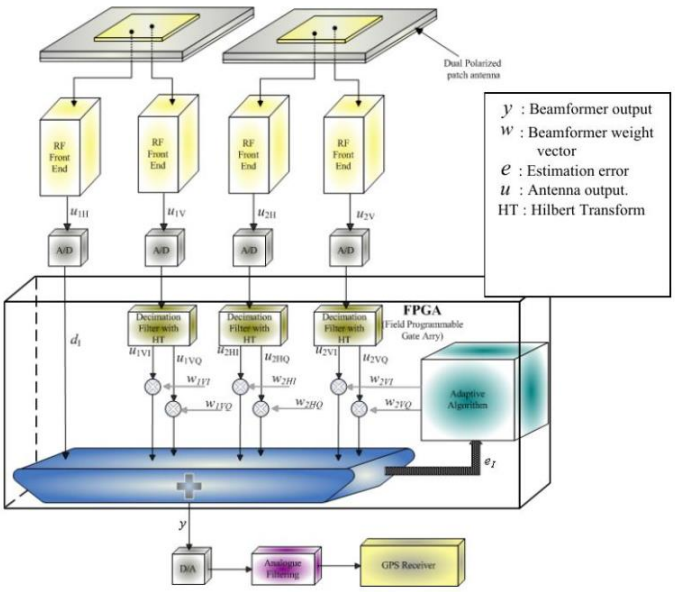

Рис.2.1. Базова структура цифрового нуль-керувальника з використанням поляриметричної антенної решітки [3]

Вихід кожного поляризаційного каналу оцифровується. Оскільки GPS L1 сигнал (1.575 ГГц) вимагає значної смуги пропускання, сигнал спочатку

перетвориться на нижчу проміжну частоту, і потім отриманий сигнал передається на мікросхему FPGA, яка виконує подальшу фільтрацію та повну обробку адаптивної антенної решітки.

Ключовим етапом початкової обробки є перетворення Гільберта (Hilbert Transform, HT), яке застосовується до кожного каналу. Це дозволяє використовувати аналітичне (комплексне) представлення сигналу, що значно спрощує математичні операції адаптивного алгоритму.

2.1.2 Адаптивний алгоритм

Оптимальні алгоритми керування нулем зазвичай мінімізують вихідну потужність зваженої суми виходів приймача за умови, що вага опорного каналу фіксована, наприклад, дорівнює одиниці. Це призводить до вектору оптимальних ваг, що задається як [3]:

$$\underline{w} = \frac{R^{-1}\underline{c}}{\underline{c}^H R^{-1}\underline{c}} \qquad [3]\,(2.1)$$

Де, $\underline{c} = [1,0,0,\dots,0]^T$,

а R – коваріаційна матриця виходів приймача, визначена як

$$R = E\left\{\underline{u}(k)\underline{u}^H(k)\right\} \qquad [3]\,(2.2)$$

Для поляриметричних решіток використовується мінімізація вихідної потужності решітки при збереженні відгуку ПКП на опорній антені. Такий підхід запобігає втраті 3 Дб відношення сигнал/шум корисного сигналу GPS за відсутності перешкод. Вектор комплексних виходів приймача для обробки подвійних каналів кожної поляризованої антени записується як:

$$\underline{u}(k) = (u_{1H}[k], u_{1V}[k], u_{2H}[k], u_{2V}[k], \dots u_{NH}[k], u_{NV}[k])^T \qquad [3]\,(2.3)$$

Де, $u_{nH}[k] = u_{nH1}[k] + ju_{nHQ}[k]$ та $u_{nV}[k] = u_{nV1}[k] + ju_{nVQ}[k]$

Щоб накласти цю вимогу, комплексні вагові коефіцієнти w_{1H} (горизонтальний) та w_{1V} (вертикальний) першого опорного елемента мають задовольняти умову:

$$w_{1H}^{*} - jw_{1V}^{*} = 1 \qquad [3]\,(2.4)$$

Узагальнюючи для повного масиву, обмеження стає:

$$\underline{w}^{H}\underline{c} = 1 \qquad [3]\ (2.5)$$

Де $\underline{c} = [1, -j, 0, \ldots, 0]^{T}$

Вектор оптимальної ваги стає:

$$\underline{w} = \frac{R^{-1}\underline{c}}{\underline{c}^{H}R^{-1}\underline{c}} \qquad [3]\ (2.6)$$

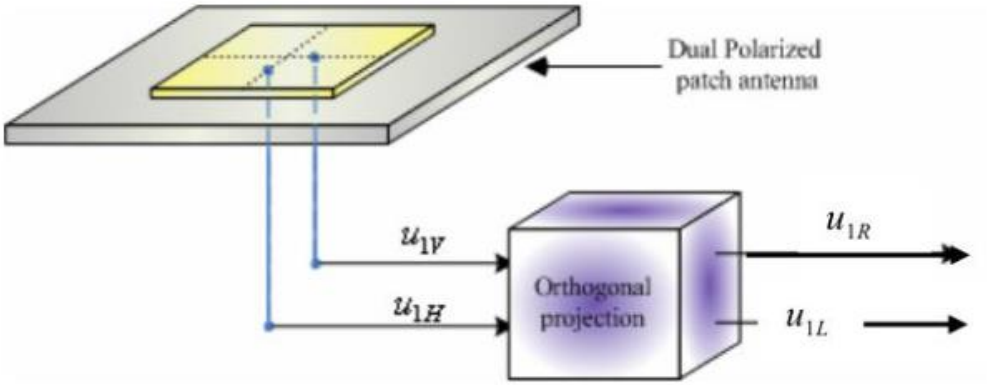

Рис.2.2. Ортогональна проекційна матриця [3]

Це обмеження дозволяє для N-канального приймача сформувати N-1 нулів. Для реалізацій цього обмеження в простому необмеженому адаптивному LMS-алгоритмі використовується ортогональна проєкційна матриця. Рис.2.2. [3] Це

перетворення є критично важливим, тому що воно приймає комплексні сигнали горизонтальної та вертикальної поляризації від антени і перетворює їх на два ортогональні канали – RHCP-сигнал (права кругова поляризація), який стає опорним каналом та LHCP-сигнал (ліва кругова поляризація), який є ортогональним до RHCP і використовується як перший допоміжний канал для адаптивного алгоритму.

Ортогональне проекційне матричне перетворення задається як:

$$\begin{bmatrix} u_{1R}[k] \\ u_{1L}[k] \end{bmatrix} = \begin{bmatrix} 1 & -j \\ 1 & j \end{bmatrix} \begin{bmatrix} u_{1V}[k] \\ u_{1H}[k] \end{bmatrix} \quad [3]\,(2.7)$$

2.2. Алгоритм LMS

Апаратна реалізація LMS-алгоритму з використанням обмеженої точності обчислень схильна до дрейфу вагових векторів, особливо в умовах значних перешкод. З метою забезпечення стабільності та точності функціонування, до класичного LMS-алгоритму додається спеціальний компонент витоку, що вимагає детального дослідження його варіацій.

2.2.1. Алгоритм LMS з витоком

Алгоритм LMS з витоком запобігає дрейфу вагового вектору, вставляючи фактор витоку α_L у цикл оновлення вектору. Цей фактор ефективно обмежує енергію імпульсної характеристики адаптивного фільтру. Рівняння оновлення вагового вектору для LMS з витоком матиме такий вигляд:

$$\underline{w}(k+1) = (1 - \mu\alpha_L)\underline{w}(k) + \mu\{\underline{u}(k)e[k]\} \quad [3]\,(2.8)$$

Де, $e[k]$ комплексно спряжена похибка, що дорівнює:

$$e[k] = d[k] - \underline{w}^H(k)\underline{\acute{u}}(k) = u_{1R}(k) - \underline{w}^H(k)\underline{\acute{u}}(k) \quad [3]\ (2.8)$$

Та

$$\underline{\acute{u}}(k) = (u_{1L}[k], u_{2H}[k], u_{2V}[k], \dots u_{NH}[k], u_{NV}[k])^T \quad [3]\ (2.9)$$

2.2.2. Алгоритм LMS з круговим витоком

Алгоритм LMS з круговим витоком має перевагу над алгоритмом LMS з витоком, оскільки не вносить зміщення в оцінки ваг. Він використовує менше апаратних ресурсів, зберігаючи при цьому ту саму продуктивність стабілізації в адаптивному фільтрі. Рівняння оновлення ваги для LMS з круговим витоком виглядає як [3]:

$$\underline{w}(k+1) = (1 - \mu\alpha_C(k))\underline{w}(k) + \mu\left[\underline{u}(k)e[k]\right] \quad [3]\ (2.8)$$

$$\text{Де } \alpha_C(k) = \begin{cases} a_0 & if \quad |w_n(k)| \geq C_2 \\ a_0 - \frac{a_0}{2}\left(\frac{C_2 - |w_n(k)|}{D}\right)^2 & if \quad 0.5\{C_1 + C_2\} \leq |w_n(k)| < C_2 \\ \frac{a_0}{2}\left(\frac{|w_n(k)| - C_1}{D}\right)^2 & if \quad C_1 < |w_n(k)| < 0.5\{C_1 + C_2\} \\ 0, & otherwise \end{cases} \quad [3]\ (2.9)$$

Циклічний член витоку є нелінійним і змінним у часі параметром, значення якого визначаються однією з чотирьох різних областей, виходячи з поточного стану вагового вектору. Ключова його особливість полягає в тому, що цей механізм витоку застосовується вибірково – лише до одного каналу на кожній ітерації.

На першій ітерації алгоритм перевіряє величину компонента вагового вектору $w_n(k)$ лише для одного поточного каналу, це дозволяє визначити в який із 4-х попередньо встановлених діапазонів він потрапляє. На наступній ітерації відповідно до визначеного діапазону, до рівняння оновлення ваги цього конкретного каналу застосовується відповідне значення циклічного члена витоку. Наступна ітерація перевіряє величину компоненту вагового вектору, що

відповідає другому каналу, тобто $|w_2(k+1)|$, щоб визначити який циклічний член витоку застосувати до рівняння оновлення. [3] Процедура послідовно продовжується, перевіряючи та оновлюючи вагу наступного каналу на наступній ітерації. Після перевірки ваг усіх каналів у багатоканальній решітці процес повертається до першого каналу і повторюється циклічно. Типовий графік кругового витоку ілюстровано на Рис.2.3., де вибрані додатні константи $a_0 = 0.3,\ C_1 = 0.6,\ C_2 = 0.9$

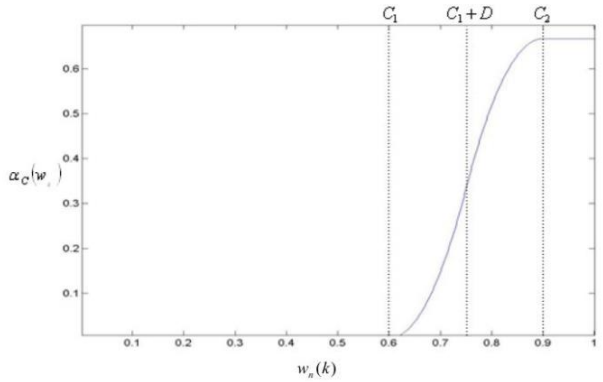

Рис.2.3. Функція кругового витоку [3]

2.2.3. Модифікований алгоритм циклічного витоку LMS

Для забезпечення роботи адаптивної системи керування нулем в умовах жорстких вимог реального часу, стандартний алгоритм циклічного витоку було замінено на модифікований лінійний варіант. Це спрощення дозволило суттєво зменшити обчислювальне навантаження та мінімізувати кількість необхідних арифметичних операцій. Лінійні рівняння для визначення коефіцієнта витоку α_{C} задаються як:

$$\alpha_{\mathrm{C}} = \begin{cases} a_0 & if \quad |w_n(k)| \geq C_2 \\ 5.2(w - 0.9133) + 0.58 & if \quad 0.5\{C_1 + C_2\} \leq |w_n(k)| < C_2 \\ 17.4(w - C_1) & if \quad C_1 < |w_n(k)| < 0.5\{C_1 + C_2\} \\ 0, & otherwise \end{cases} \quad [3](2.10)$$

На відміну від рівняння (2.9), рівняння (2.10) зберігає ті самі чотири області регулювання, але досягає значного зменшення обчислювальної складності. Це стало можливим завдяки заміні квадратичних членів на лінійні та виключенню операції ділення. В результаті, функція вимагає менше операцій множення і може виконати повний розрахунок оновлення вагового вектору за один такт. Ця висока швидкість виконання робить її ідеально придатною для апаратної реального часу.

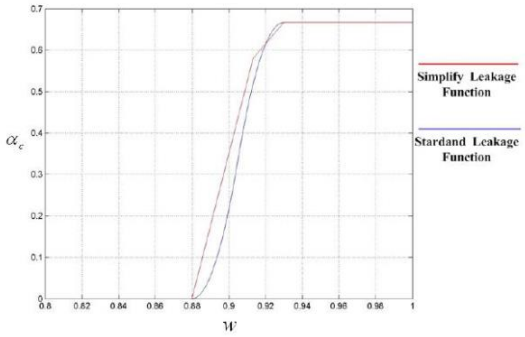

Рис.2.4. Спрощена/стандартна кругова функція витоку з $C_1 = 0.88$, $C_2 = 0.93$

На Рис.2.4. візуально порівнюється функція коефіцієнта витоку α_C для стандартного кругового витоку (синій) та модифікованого кругового витоку (червоний). Графік демонструє, що функція має нелінійну, сегментовану поведінку. Функція різко зростає, щойно оцінка ваги досягає першого порогового значення $C_1 = 0.88$. Після цього вона продовжує зростати майже лінійно зі збільшенням ваги, і потім поступово спадає, не доходячи до верхньої межі $C_2 = 0.93$. [3]

2.3. Реалізація FPGA

Програмована вентильна матриця (Field-Programmable Gate Array, FPGA) – тип інтегральної мікросхеми, яка, на відмінну від звичайних процесорів (CPU) чи спеціалізованих мікросхем (ASIC), не має фіксованої внутрішньої структури. FPGA має чотири основні модулі обробки, включаючи цифрову децимаційну

фільтрацію, фільтрацію HP та реалізацію адаптивного алгоритму LMS з використанням спрощеного алгоритму циклічного витоку. Діаграма потоку сигналу показана на Рис.2.5.

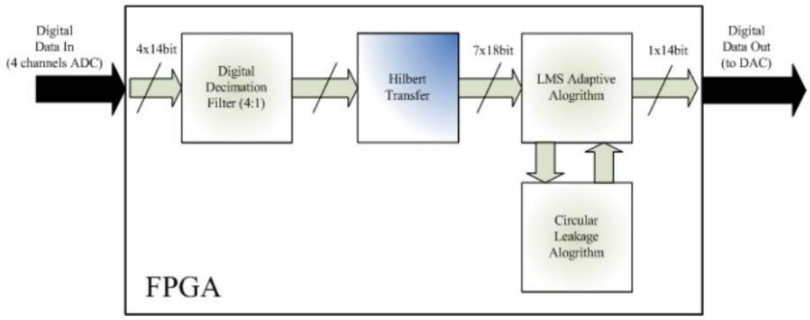

Рис.2.5. Схема потоку сигналу FPGA [3]

Для реалізації коду адаптивного пристрою керування нулем було використано інструментальне середовище Xilinx System Generator. Схематична блок-схема цієї структури, зображена на Рис.2.6., що демонструє, що система приймає 8 вхідних сигналів дійсні та уявні компоненти горизонтальної та вертикальної поляризації від двох антенних елементів – і генерує один вихідний сигнал.

Обробка сигналу починається з того, що 4 верхні входи подаються на блок ортогональної проєкції, який перетворює горизонтальну та вертикальну поляризацію опорної антени та канали RHCP та LHCP. Канал RHCP представлений як u_RHCP_I обирається як опорний канал та який обробляється як бажаний сигнал $d_1(k)$. Решта сигналів формують вектор допоміжних каналів, куди входять дійсна та уявна частини LHCP-сигналу, таким чином система оперує єдиним опорним та трьома парами допоміжних каналів.

Кожен із дійсних та уявних сигналів допоміжних каналів подається на відповідний блок адаптивного алгоритму на основі LMS. Усі виходи цих блоків потім надходять на суматор і віднімаються від опорного сигналу, утворюючи сигнал похибки. Цей сигнал повертається назад до блоків LMS, запускаючи безперервний цикл для обчислення оновлення вагових коефіцієнтів. Цикл LMS

працює безперервно, з вихідною частотою дискретизації, забезпечуючи постійну та швидку оцінку оптимальних ваг у режимі реального часу.

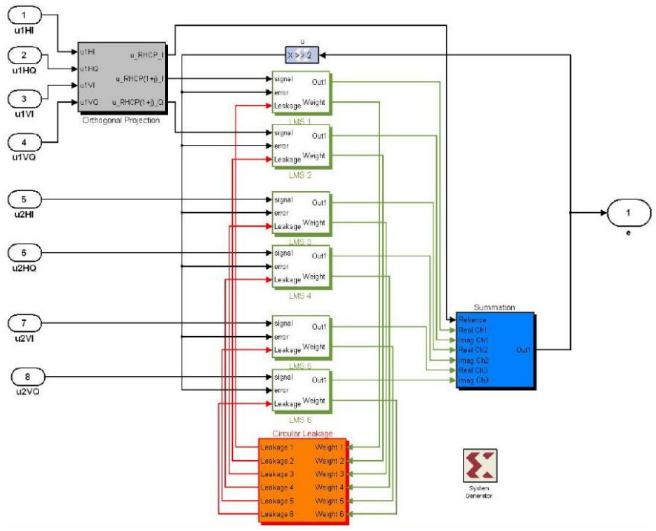

Рис.2.6. Чотириканальний цифровий нульовий направляючий пристрій - схема FPGA [3]

Алгоритм кругового витоку реалізовано мовою VHDL, як показано на Рис.2.6., у блоці під назвою "круговий витік". Цей блок містить 6 входів праворуч та таку ж кількість виходів ліворуч. Його основна функція полягає у реалізації логіки оператора «якщо/то» з рівняння (2.10.) На кожній ітерації алгоритм вимірює поточну оцінку ваги лише для одного каналу й обчислює відповідне значення витоку потім подається назад, як вихід leak_out на Рис.2.7. і віднімається від поточної складової вектору ваг. [3] На наступній ітерації блок циклічного витоку виконує перевірку та обробку ваги вже для наступного каналу в решітці, забезпечуючи послідовне циклічне керування дрейфом.

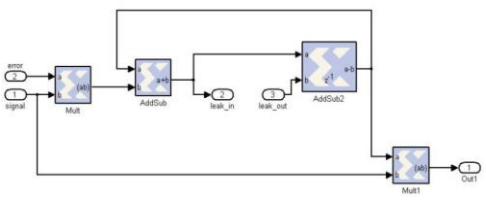

Рис.2.7. Базовий алгоритм LMS-блок - схема FPGA [3]

2.4. Аналіз завад та переваги адаптивних антенних решіток

Сигнали Глобальних супутникових радіонавігаційних систем (ГНСС) є вразливими до перешкод. Як первинний захід протидії використовується характеристика поляризації корисного сигналу. Сигнал GPS передається з правою круговою поляризацією (ПКП), яка формується горизонтальною та вертикальною складовими, зміщеними по фазі на 90°.

Одна GPS антена з адаптивною поляризаційною характеристикою може ефективно відхиляти перешкоди з поляризацією, відмінною від ПКП. Ключова ідея компенсаторів поляризації полягає в адаптивному узгодженні відгуку антени з поляризацією завади, це дозволяє придушити будь-який лінійно-поляризований сигнал з мінімальною втратою потужності корисного сигналу GPS лише на 3дБ. Реалізується це шляхом виходів подвійної поляризаційної патч-антени.

Обробка сигналів лише в поляризаційній області має критичне обмеження: вона стає неефективною, якщо перешкода має таку ж поляризацію, як і сигнал GPS. Водночас, адаптивні формувачі променю працюють виключно у кутовій області, ігноруючи поляризаційні відмінності. Для подолання цих недоліків використовується комбінований підхід, що об’єднує обидві області в одному алгоритмі. Це досягається застосуванням методів формування променю до антенної решітки з подвійно поляризованими елементами. Завдяки цьому рішенню досягається збільшення ступенів свободи. Комбінація просторової та поляризаційної обробки приблизно подвоює ступені свободи, доступні для придушення завад, порівняно зі звичайною (одно поляризованою) антенною решіткою з тією ж кількістю елементів.

2.4.1. Типи завад

1) Одночастотна завада

Є найпростішим та найфундаментальнішим типом серед усіх завад. Описується математичними виразами часової та частотної областей:

$$J(t) = A\cos(2\pi f_c t) \quad [4]\ (2.11)$$

$$J(f) = \delta(f - f_c) \quad [4]\ (2.12)$$

Де A – амплітуда сигналу, а f_c – його несуча частота.

Основна характеристика цього типу завади полягає у відносно концентрованій енергії. Завдяки цій концентрації одночастотна завада має високу ефективність глушіння для навігаційного сигналу, що знаходиться поблизу цієї частотної точки. Проте, її основним недоліком є вузька смуга пропускання, що дозволяє легко придушити заваду до рівня теплового шуму за допомогою частотної фільтрації. Рис.2.8.

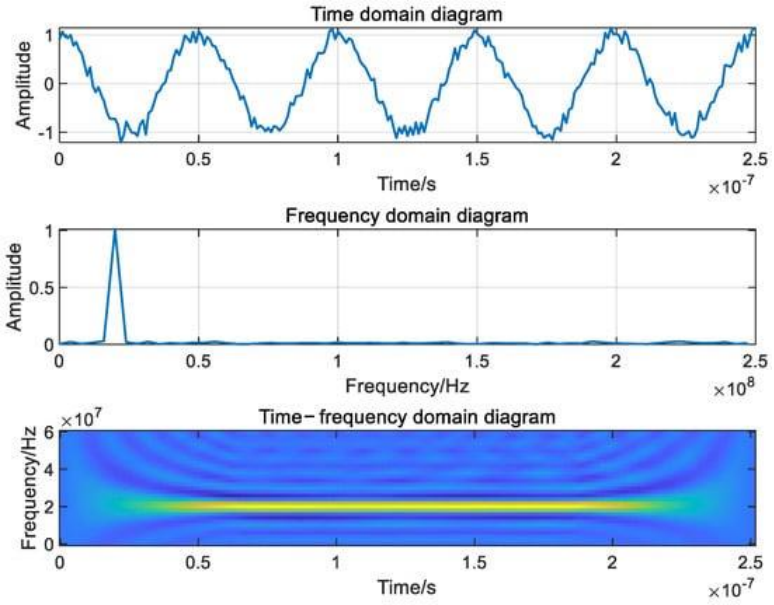

Рис.2.8. Діаграма характеристик сигналу одночастотного глушіння [4]

2) Імпульсна завада

Імпульсна завада – це тип завади, який складається з послідовності ідеальних прямокутних імпульсів. Математично цей сигнал описується

виразом, що включає амплітуду завади A, несучу частоту навігаційного сигналу f_c, ідеальний прямокутний імпульсний сигнал $s(t)$, а також ширину імпульсу τ, та період імпульсу T. При цьому частота завади контролюється параметром T, а коефіцієнт заповнення контролюється співвідношенням τ та T.

$$J(t) = A\cos(2\pi f_c t)s(t) \quad [4]\ (2.13)$$

$$s_i(t) = \begin{cases} 1 - \frac{\tau}{2} + nT \le t \le \frac{\tau}{2} + nT, n = 1,2 \dots \\ 0 \qquad\qquad else \end{cases} \quad [4]\ (2.14)$$

$$S_i(f) = \sum_{-\infty}^{+\infty} 2\frac{\sin\left(\frac{n\pi\tau}{T}\right)}{n}\delta\left(f - \frac{n}{T}\right) \quad [4]\ (2.15)$$

Імпульсна завада відрізняється невеликою смугою пропускання, але високою ефективністю глушіння. Контролюючи ширину та період імпульсу, можна зміщувати смугу частот завади, досягаючи ефективного покриття діапазону частот цільового навігаційного сигналу. Рис.2.9.

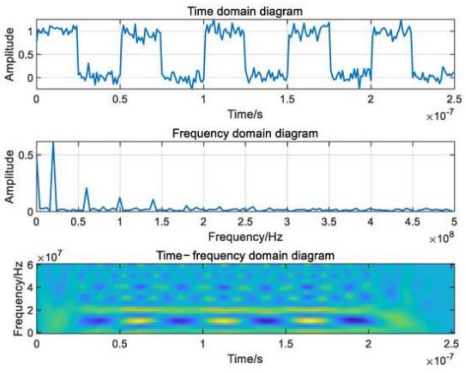

Рис.2.9. Діаграма характеристик сигналу імпульсного заглушення [4]

3) Смугова завада

Смугова завада, також відома як сигнал лінійної частотної модуляції, схожа за формою на одночастотну заваду, але має принципову відмінність: несуча частота смугової завади змінюється з часом, тоді як у одночастотної завади вона є фіксованою. Загальний математичний вираз форми смугової завади в частотній області:

$$J(t) = A\cos(2\pi(f_c + f_{sweep}t)t) \qquad [4]\ (2.16)$$

Де, A – амплітуда сигналу смугової завади, f_c – несуча частота сигналу супутникової навігації, f_{sweep} – швидкість зміни частоти.

Діаграма часової області, діаграма частотної області та часово-частотна область смугової завади показані на Рис.2.10.

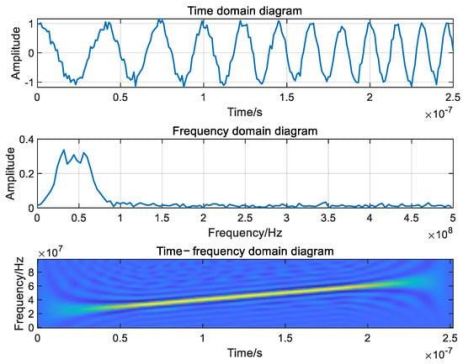

Рис.2.10. Діаграма характеристик сигналу смугової завади [4]

4) Завада з узгодженим спектром

Цей тип завади поєднує в собі характеристики навігаційного супутникового сигналу для точного націлювання на низхідний канал зв'язку. Завада з узгодженим спектром вводиться безпосередньо у спред-спектральний код навігаційного супутникового сигналу. Тому для генерації узгодженого спектру необхідно попередньо отримати конкретний формат спред-спектрального коду навігаційного сигналу. Ця завада має ті ж характеристики спектральної щільності потужності, що й цільовий навігаційний сигнал, що забезпечує високий сигнал глушіння.

На прикладі сигналу з модуляцією BPSK (Binary Phase Shift Keying), вираз форми сигналу в часовій області є таким:

$$J(t) = A\cos(2\pi f_c t)p(t) \qquad [4]\ (2.17)$$

Де, $p(t)$ псевдокодова послідовність супутникового навігаційного сигналу.

Діаграма часової області, діаграма частотної області та часово-частотна область модульованого BPSK завади з узгодженим спектром показано на Рис.2.11.

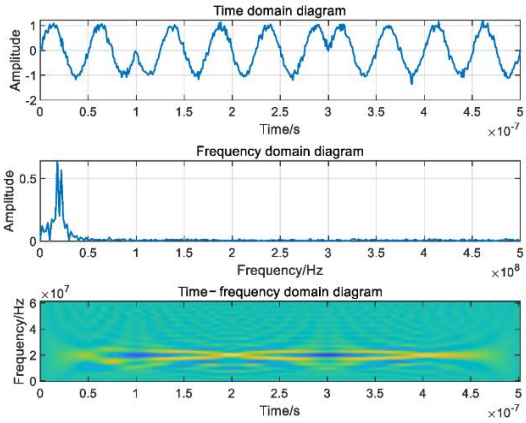

Рис.2.11. Діаграма характеристик сигналу модульованого BPSK узгодженого спектрального заглушення. [4]

2.4.2. Порівняння ефективності антенних решіток та одноантенних систем у боротьбі із завадами

Наразі, технологія антенних решіток вважається одним із найефективніших засобів придушення завад у навігаційних приймачах, оскільки вона здатна ефективно усувати численні завади. Дослідження та вдосконалення цієї технології тривають. Адаптивні антенні решітки демонструють високу ефективність проти стабільних і неперервних завад. Однак, вплив нестабільних та неперервних завад на ефективність навігаційних приймачів залишається недослідженими. Зокрема, коли приймач стикається з розподіленою інтермітуючою (переривчастою) завадою, ключовим фактором, що впливає на продуктивність протидії, стає узгодженість між коваріацією тренувальних вибірок та коваріацією фактичних оброблювальних сигналів.

Вибір технології протидії завадам для супутникової навігаційної антени, як правило, залежить від ширини смуги завади. За кількістю антен приймача,

технології протидії завадам поділяються на одноантенні та технології антенних решіток. Технологія антенних решіток використовує векторне зважування для скасування завад за допомогою кількох елементів антени. Завдяки цьому технологія антенних решіток нечутлива до ширини смуги завади і здатна ефективно придушувати завади з різною шириною смуги. На противагу цьому, при придушенні широкосмугових завад одноантенна технологія спричиняє значене погіршення якості сигналу. Однак, саме одноантенна технологія є найкращим вибором, коли необхідно придушити вузькосмугові завади.

2.4.3. Дослідження методів придушення завади антенними решітками

Технологія протидії завадам з використанням антенних решіток наразі вважається найбільш ефективним методом, здатним придушувати численні як вузькосмугові, так і широкосмугові завади. У результаті, найважливіше навігаційне обладнання оснащується GNSS-приймачами з антенними решітками, що своєю чергою, призводить до послаблення загального ефекту придушення завад. Таким чином, виникає необхідність досліджувати методи придушення завад, спеціально розроблені для протидії антенним решіткам.

Дослідження методів придушення завади, спрямованої проти антенних решіток, можна розглядати під двома кутами: кількість джерел завади та тип завади. По-перше, згідно з принципом глушіння, коли кількість джерел завади перевищує кількість елементів антени приймача GNSS (тобто ступінь свободи решітки), ефективність протидії завадам різко знижується. Більшість наявних алгоритмів протидії завадам розглядають лише сценарії, де кількість джерел менша або дорівнює кількості елементів антени, тоді як дослідження методів глушіння з надмірним ступенем свободи (super degree of freedom jamming) бракує як з точки зору теоретичного обґрунтування, так і аналізу. Хоча розглядається використання багатьох джерел завад для реалізації глушіння, основна увага наразі приділяється досягненню покриття зони через мережеве кооперативне розгортання, ігноруючи вплив завад з надмірним ступенем

свободи на приймач GNSS з антенною решіткою. По-друге, сучасна технологія придушення завад переважно зосереджена на оцінці впливу одиничного типу завади. Насправді ж, GNSS функціонує в дуже складному середовищі радіоелектронної боротьби, і сценарій з однією завадою існує лише в симуляційних аналізах. У майбутньому основним напрямком досліджень завад для антенних решіток стануть нові стилі, такі як композитні завади з надмірним ступенем свободи та нестаціонарні неперервні завади.

Висновки

В цьому розділі було описано адаптивну багатоканальну обробку сигналів. Спочатку було проведено аналіз завад та переваги адаптивних антенних решіток. Було розглянуто структуру та принципи формування вагових коефіцієнтів антенної решітки з подвійною поляризацією, а саме досліджено архітектуру системи та адаптивний алгоритм. Окремо було розглянуто алгоритм LMS, а саме його такі види як: алгоритм LMS з витоком, алгоритм LMS з круговим витоком та модифікований алгоритм циклічного витоку LMS. Також було описано реалізацію FPGA. Також було проведено аналіз завад та переваги адаптивних антенних решіток, зокрема описано типи завад та здійснено порівняння ефективності антенних решіток та одноантенних систем у боротьбі із завадами. Також було досліджено методи супресивної завади для антенних решіток.

РОЗДІЛ 3
АРХІТЕКТУРА ПРОГРАМНО-КОНФІГУРОВАНОГО ПРИЙМАЧА ТА МЕТОДИ АДАПТИВНОЇ ОБРОБКИ СИГНАЛІВ ДЛЯ CPRA-АНТЕНИ

3.1. Проблема завад у ГНСС

ГНСС широко інтегровані в сфери навігації, спостереження та точного часу. У зв’язку з цим, їхня вразливість до радіочастотних завад (РЧЗ) набуває особливої актуальності. Для підвищення стійкості систем і гарантування їхньої надійності пропонується використання адаптивних приймачів з керуванням променем на основі багатоелементних антен (Controlled Reception Pattern Antenna, CRPA). Ця технологія дозволяє активно формувати діаграму спрямованості антени, щоб підсилювати корисний сигнал у напрямку супутника та ефективно пригнічувати завади, які надходять з інших напрямків.

Важливим кроком у розвитку цієї технології стало впровадження програмно-конфігурованого приймача (Software-Defined Radio, SDR). На відміну від традиційних апаратних рішень, як-от FPGA, програмна реалізація пропонує вищу гнучкість, скорочує час розробки та є більш економічно ефективною, оскільки може використовувати комерційно доступні процесори загального призначення (COTS). Проте розробка SDR для CRPA є складним завданням через значно вищу вимогу до обчислювальної потужності порівняно зі звичайним приймачем. Його частота дискретизації становить 16 Msps, що є достатнім для обробки сигналу GPS L1 C/A з шириною смуги коду близько 2 МГц. Однак ця швидкість не придатна для високошвидкісного сигналу GPS L5, смуга частот коду якого становить 20 МГц. Крім того, застосована 2-бітна роздільна здатність дискретизації є недостатньою для забезпечення необхідного динамічного діапазону сигналу, що критично важливо для ефективної роботи систем захисту від потужних радіочастотних завад. Для вирішення цієї проблеми і забезпечення роботи в реальному часі пропонується нова архітектура SDR, що ґрунтується на паралельній обробці даних з

використанням високопродуктивних графічних процесорів (GPU) та багатоядерних центральних процесорів (CPU). Це дозволяє досягти високої обчислювальної пропускної здатності, необхідної для повноцінної адаптивної протидії завадам.

3.2. Архітектура GPS SDR для обробки CRPA та обчислювальних задач

З метою підвищення обчислювальної ефективності у SDR для оновлення вектору вагових коефіцієнтів використовується ітераційний підхід, який описується рівнянням:

$$W_{n+1} = W_n + \Delta W_n \qquad [5]\ (3.1)$$

Де,

$$\Delta W_n = \gamma[\mu]T^* - \Phi_n W_n] \qquad [5]\ (3.2)$$

Цей метод дозволяє уникнути необхідності розраховувати обернену величину Ф. У цьому контексті γ є параметром розбіжності, який регулює швидкість збіжності алгоритму та рівень неузгодженості в стаціонарному режимі.

Витрати обчислювальної потужності для SDR GPS із чотириелементною антеною CRPA значно зростають через необхідність фазового калібрування «на льоту». Це вимагає залучення 5-ти каналів відстеження на кожен канал керування променем. Як наслідок, вартість операцій скидання несучої та коду зростає до $5 \times 216N = 1080N$. Для формування променю необхідно об’єднати вхідні дані з 4-х елементів антени, застосовуючи до них відповідні вагові коефіцієнти. Процес зважування реалізовується як комплексне множення, де для N пар синфазних та N квадратурних вибірок з однієї антени потрібно 6N операцій множення та додавання. Відповідно, зважування даних з усіх 4-х антен вимагає $4 \times 6N = 24N$ операцій на канал керування променем. [5]

Таким чином, загальна кількість операцій для 12 каналів керування променем кожен з яких має свій вектор ваг, становить $12 \times (24N + 3N + 3N) = 360N$ операцій. [5]

Адаптивна обробка MVDR (Minimum Variance Distortionless Response) вимагає обчислення коваріаційної матриці сигналу. Ця матриця будується на основі вектору вимірювань X, який є комплексною матрицею розміром $4N$.

$$X^*X^T = (A - jB)(A^T + jB^T) = (AA^T + BB^T) + j(AB^T - A^TB) \qquad [5] (3.3)$$

Обчислення повного добутку AB^T у рівнянні (3.3) займає приблизно $32N$ операцій множення та додавання, але обчислення AA^T та BB^T займає половину операцій через його симетрію. Таким чином, X^*X^T коштує приблизно $(16N + 16N + 32N) = 64N$. Таким чином, сукупна кількість операцій множення та додавання, необхідних для виконання 3-х ключових етапів – стирання несучої та коду, синтезу даних антени та розрахунку коваріаційної матриці становить приблизно $1080N + 360N + 64N = 1504N$. [5]

3.3. Розробка GPS SDR для CRPA

У реалізованому SDR весь процес обробки сигналу, починаючи після аналого-цифрового перетворення виконується на процесорах загального призначення. Як імпровізована багатоелементна антена CRPA використовується набір з 4-х антен Trimble Zephyr. Рис.3.1.

Сигнали GPS L1 з цих 4-х антен знижуються до основної смуги та оцифровуються за допомогою 4-х комерційних дигітайзерів – USRP2. Для запису необроблених даних частота дискретизації кожного USRP2 встановлена рівні 40 Msps. При цьому кожна вибірка має розмір 2 байти з 14-ти бітною роздільною здатністю. Оцифровані дані передаються через Gigabit Ethernet на твердотілі накопичувачі SSD, які необхідні, оскільки швидкість потоку даних кожної антени 80 МБ/с перевищує швидкість запису традиційних жорстких

дисків. Обчислювальна спроможність SDR у реальному часі демонструється шляхом відтворення записаних даних швидше, ніж відбувався запис. Для роботи приймача використовуються процесори загального призначення – CPU та GPU.

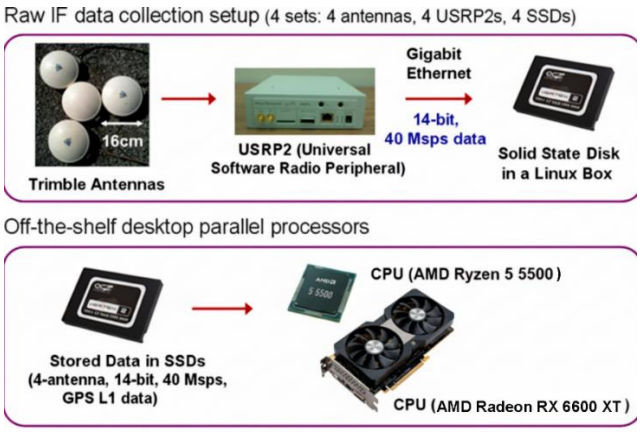

Рис.3.1. Налаштування апаратного забезпечення GPS SDR для обробки чотириелементної CRPA

3.3.1. Паралельний корелятор на основі графічного процесора

Графічний процесор AMD Radeon RX 6600 XT, який використовується в цій роботі, побудований на архітектурі RDNA 2 і має 32 обчислювальні блоки (CU) та 2048 потокових процесорів. Оскільки кожен CU може мати певну максимальну кількість активних потоків, загальна теоретична кількість активних потоків для RX 6600 XT, наприклад при 10 активних вейвфронтах по 64 потоки, може досягати до $32 \times 10 \times 64 = 20480$ активних потоків.

Однак, цього максимально можливого завантаження графічного процесора може бути не досягнуто з різних причин. Завантаження графічного процесора визначається кількістю потоків на групу потоків, кількістю регістрів, що використовуються на потік, та виділеною локальною пам’яттю (LDS) для CU. Після визначення цих трьох параметрів, завантаження обчислювального блоку можна розраховувати.

Після кількох ітерацій проектування призначаємо певну кількість потоків, що формує 4 вейвфронти на групу потоків та певну кількість груп потоків на CU. В архітектурі RDNA2 потоки виконуються групами по 64 потоки, які називають вейвфронатми. Кожен CU AMD Radeon RX 6600 XT може мати максимальну кількість активних вейвфронтів. Таким чином, коли група потоків має 4 вейвфронти і на кожному CU призначено 2 групи потоків, можна досягти максимального завантаження. Однак, наше проектування може бути обмежене кількістю регістрів на CU, так як фізичне обмеження становить 65 536 32-х бітних регістрів на CU, і кожному CU може бути призначено лише обмежену кількість груп потоків. Отже, призначається, наприклад, 8 активних вейвфронтів на CU. Завантаження графічного процесора в цьому випадку становить $8\ \text{вейфронтів}\ /10\ \text{вейфронтів} = 80\%$.

Якщо ми можемо зменшити кількість призначених регістрів на потік, то завантаження можна збільшити, але зазвичай важко зменшити кількість регістрових змінних у вихідному коді під час виконання самого завдання. Максимальну кількість регістрів на потік можна встановити вручну за допомогою відповідного прапорця під час компіляції, аналогічно ‘-maxrregcount` Хоча цей прапорець може зменшити використання регістрів та збільшити завантаженість, він не завжди покращує продуктивність, оскільки розлиті регістри зберігаються в набагато повільнішій локальній пам’яті, а вища завантаженість вище певного порогу не дає додаткового приросту продуктивності. Примусово використовуємо оптимальну кількість регістрів, що призводить до найкращої продуктивності в нашому випадку.

Для повного приховування затримки залежності читання та запису регістрів потрібно лише помірне заповнення, наприклад 50% або менше. Якщо заповнення перевищує цей поріг, подальша оптимізація параметрів для отримання ще вищого заповнення може не мати сенсу. При заповненні 80% на один CU має $64\ \text{потоки} \times 8\text{вейвфронтів} = 512\ \text{ативних потоків}$. А графічний процесор AMD Radeon RX 6600 XT з 32 CU має $512 \times 32 = 16364\ \text{активні потоки}$. Таким чином, очевидно, що потокові процесори не

можуть виконувати операції над усіма активними потоками одночасно в один і той самий такт. Однак, важливо генерувати велику кількість активних потоків, щоб приховати затримку пам’яті. Тому, ядра постійно матимуть достатньо потоків для обробки, поки інші потоки чекають на читання пам’яті.

3.3.2. Обсяг пам'яті на пристрої AMD Radeon RX 6600 XT

На Рис.3.2. зображено головний комп'ютер та відеокарту, або пристрій, з графічним процесором AMD Radeon RX 6600 XT.

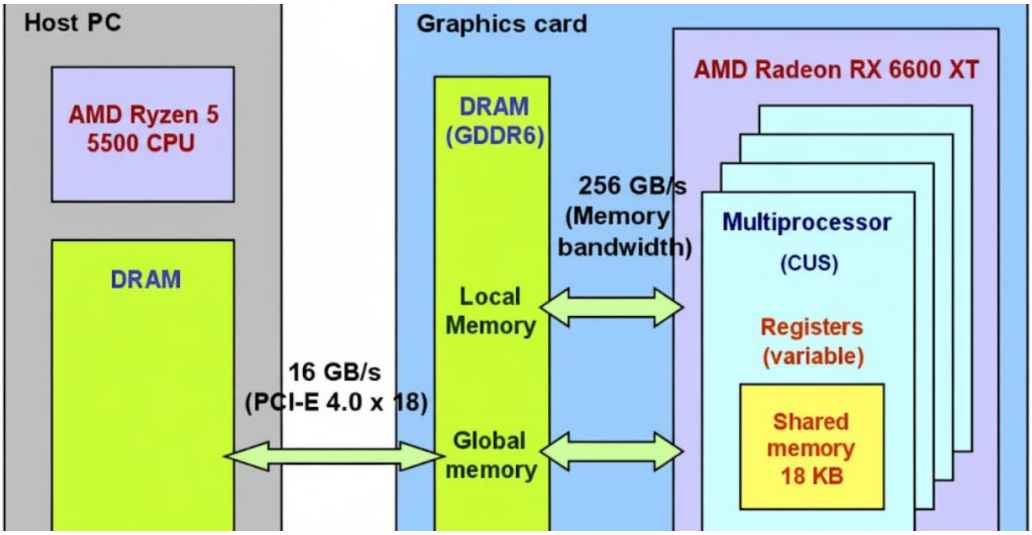

Рис.3.2. Обсяг пам'яті на головному комп'ютері та пристрої AMD Radeon RX 6600 XT

Для обробки даних, що знаходяться в пам’яті головного комп’ютера за допомогою GPU, дані необхідно скопіювати в глобальну пам'ять пристрою (VRAM). Шина даних для GPU RX 6600 XT – це PCI-E 4.0 x8, максимальна теоретична пропускна здатність якої становить 15.8 ГБ/с.

Після копіювання даних із пам’яті хоста в глобальну пам'ять, обчислювальні блоки CU отримують доступ до цих даних для обробки. Теоретична максимальна глобальна пропускна здатність (ПЗ) пам’яті RX 6600 XT розраховується на основі його специфікацій.

$$\text{ПЗ} = \text{Шина інтерфейсу пам′яті} \times \frac{1}{\text{Байт/біт}} \times \text{ефективна частота} \quad [5]$$

$$\text{ПЗ} = 128 \text{ біт} \times \frac{1}{8} \frac{\text{Байт}}{\text{біт}} \times 16000\text{МГц} = 256\text{Гб/с}$$

Локальна пам’ять призначена для великих структур, масивів або розлитих регістрів. Якщо код GPU запитує більше, ніж доступно регістрів, ці змінні розміщуються в локальній пам’яті.

Доступ до DRAM має високу затримку через обмежену пропускну здатність, але доступ до вбудованої пам’яті набагато швидший. Затримка некешованої спільної пам’яті на кристалі приблизно в 100 разів нижча, ніж затримка глобальної пам’яті. Тому базова стратегія оптимізації полягає в мінімізації доступу до пам’яті з високою затримкою, що включає копіювання пам’яті з хоста на пристрій та доступ до глобальної пам’яті. GPU оптимізований для високої пропускної здатності обробки, і він має слабкі місця в завданнях, що вимагають високого доступу до пам’яті, але низької обчислювальної інтенсивності.

Що стосується ПЗ обробки, теоретична максимальна ПЗ 32-х бітних операцій з плаваючою комою (FP32) на RX 6600 XT становить 10.60 Tflops. Це значно перевищує ПЗ сучасних CPU. Теоретична максимальна ПЗ RX 6600 XT розраховується на основі його специфікацій.

$$2048 \text{ ядер} \times 2589 \text{ МГц} \times \text{Операцій/цикл} \approx 10.60 Tflops$$

3.3.3. Передача та синтез даних

Щомілісекунди необроблені синфазні та квадратурні вибірки, що зберігаються в пам’яті хоста, копіюються до VRAM RX 6600 XT для виконання операцій стирання несучої та коду. Затримка цього копіювання може бути ефективно прихована шляхом паралельного запуску рушіїв прямого доступу до

пам’яті та обчислювальних рушіїв, затримка доступу до VRAM все ще є ключовою проблемою проектування. Загальноприйнята стратегія оптимізації полягає в тому, щоб звернутися до глобальної пам’яті лише один раз і зберегти прочитані дані у швидкій вбудованій локальній пам’яті даних (LDS), оскільки доступ до неї значно швидший. В архітектурі RX 6600 XT LDS є обмеженим ресурсом (64КБ на кожен CU). У нашій реалізації розподілено робочі групи таким чином, щоб копіювати 768 синфазних вибірок, 768 квадратурних вибірок та 788 значень коду C/A, що являє собою 9.3КБ, з VRAM до LDS щомілісекунди. Додаткові 20 вибірок коду C/A копіюються для забезпечення ранньої, точної та пізньої реплік коду, необхідних для відстеження.

Важливою приміткою оптимізації також є мінімізація розбіжних вейвфронтів (груп по 64 потоки), оскільки це негативно впливає на обчислювальну ПЗ. З цієї причини кількість призначених вибірок 768 вибрано кратною кількістю потоків у робочій групі, щоб кожен потік виконував ідентичні операції над різними даними. Хоча 788 значень коду C/A зберігаються в LDS, для обробки вибираються відповідні 768 вибірок як репліки коду.

Під час копіювання даних з VRAM до LDS бажано використовувати коалесцентний доступ до пам’яті. Він досягається індексацією потоків, щоб суміжні потоки зверталися до суміжних вибірок у глобальній пам’яті. Хоча вибірки у VRAM є 16-ти бітовими цілими числами, вони копіюються до LDS як 32-х бітні числа з плаваючою комою для подальшої обробки. Оскільки кожен канал відстеження має різну доплерівську частоту, межа коду C/A кожного каналу у потоці вибірок не вирівняна. Отже, нам потрібно копіювати 1мс вибірок, що являє загалом 40000 вибірок для кожного каналу відстеження з різної початкової точки у VRAM, і ця початкова точка може не бути вирівняна із сегментом пам’яті. Якщо доступ до VRAM не вирівняний, можуть виконуватися додаткові транзакції пам’яті.

Синтез необроблених даних із 4-х елементів антени з відповідними ваговими коефіцієнтами є важливою частиною обробки GPS CRPA, і його обчислювальна частка становить приблизно 24% від загального навантаження.

У нашій реалізації синтез виконується паралельно за допомогою масивної кількості активних потоків. Цей паралелізм реалізується шляхом додавання операцій множення на відповідні вагові коефіцієнти до синфазних та квадратурних вибірок кожним потоком під час копіювання даних із глобальної пам’яті до LDS, що ефективно інтегрує формування променю в операцію передачі даних.

3.3.4. Паралельне скорочення в перерозподіленій спільній пам'яті

Наступною необхідною операцією для завершення кореляції є підсумовування значень, отриманих після видалення несучої та коду. Паралельне скорочення використовується для швидкого підсумовування багатьох значень. На Рис.3.3. показано простий приклад паралельного скорочення, здійсненого для підсумовування 4-х чисел. У цьому прикладі перший потік додає числа 2 та 3, тоді як другий потік додає числа -1 та 1 протягом одного циклу інструкцій. Слід звернути увагу, що сусідні потоки звертаються до сусідніх чисел, тобто 2 та -1 або 3 та 1, запобігаючи цим конфліктам банків спільної пам’яті.

Перед виконанням паралельного скорочення кожен потік підсумовує 3 значення, присвоюючи отриману суму перерозподіленій спільній пам’яті. Таким чином 256 чисел на блок потоку стають готовими до паралельного скорочення. Кількість вибірок, призначених для паралельного скорочення, повинна бути степенем 2, оскільки половина чисел додається до іншої половини чисел на кожному циклі інструкції. На першому циклі потоки з 1 по 128 додають по 2 числа, на другому потоки з 1 по 54 додають по 2 числа і так далі. Отже 256 чисел додаються після 8-ми циклів інструкцій у цьому паралельному алгоритмі. Виконуючи це паралельне скорочення у спільній пам’яті, а не в глобальній, ми враховуємо необхідність алгоритму в повторному доступі до пам’яті. Щойно підсумовування завершено, накопичене число

зберігається як перший елемент масиву з 256 елементів, що використовувався для паралельного скорочення. [5]

Оскільки 27 блоків потоків виконують кореляцію для одного каналу відстеження, їхні результати паралельного скорочення повинні бути об’єднанні. Через відсутність прямої взаємодії між блоками через спільну пам'ять, ці числа атомарно додаються до змінної VRAM. Атомарне додавання для 32-х бітних чисел з плаваючою комою є послідовним, але оскільки підсумовуються лише 27 чисел, тобто невеликий обсяг, ця операція не є ресурсоємною. Основне інтенсивне підсумовування вже виконано паралельно всередині кожного блоку. Кінцевий результат кореляції це 6 чисел на канал відстеження. Ядро GPU обробляє 1мс даних приблизно за 0.59мс, що підтверджує можливість роботи в реальному часі.

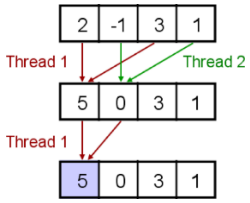

Рис.3.3. Простий приклад паралельного скорочення для підсумовування чотирьох чисел [5]

3.3.5. Апаратний паралелізм

Ядро GPU, виконуючи синтез даних, видалення несучої та коду обробляє найбільшу частину обчислювального навантаження CRPA, приблизно $(1080N + 360N)/1504N \approx 96\%$. Решта обчислень, що становить $64N/1504N \approx 4\%$, призначені для розрахунку коваріації сигналу. Оскільки CPU та GPU є різними апаратними засобами, вони можуть виконувати завдання повністю паралельно. Шестиядерний CPU AMD Ryzen 5 5500 достатньо потужний, щоб завершити розрахунок коваріації, поки ядро GPU виконує інші

операції. Таким чином, призначаємо розрахунок коваріації CPU та запускаємо його паралельно з GPU. Рис.3.4. [5].

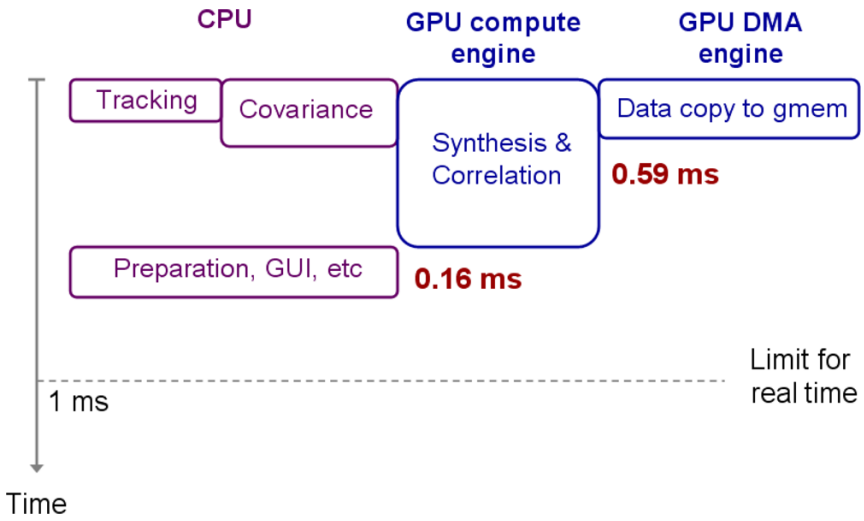

Рис.3.4. Апаратний паралелізм з обчислювальним механізмом CPU

Для паралельної обробки на CPU генеруємо 10 потоків CPU для розрахунку коваріації. Матриця коваріації $X * X^T$ дорівнює власній спряженій транспозиції, розміром 4 на 4. Завдяки її симетрії, ми обчислюємо лише верхню трикутну частину, тобто 10 чисел, призначаючи один потік CPU для обчислення кожного з цих чисел. Додатково ми генеруємо 5 потоків CPU для операцій відстеження сигналу на основі результатів кореляції за попередню мілісекунду (один потік на антену та один потік для каналів керування променем). Хоча генерується 15 потоків загалом, Ryzen 5 5500 завдяки технології SMT (Simultaneous Multi-Threading) може одночасно виконувати до 12 потоків, що являє собою 6 ядр. Отже, 15 згенерованих потоків виконуються, розподіляючи час процесора. Багатопотоковий розрахунок коваріації на CPU додатково оптимізований інструкціями SIMD. Час виконання 10 потоків CPU для розрахунку коваріації становить близько 0.29 мс, а 5 потоків для відстеження займають близько 0.07мс. [5]

Крім того, існує важливий апаратний паралелізм на відеокарті: затримка копіювання вхідних даних з пам’яті хоста в VRAM є значною мірою через обмежену ПЗ PCI-E. Однак GPU AMD мають виділені механізми прямого

доступу до пам’яті (DMA). Затримка копіювання пам’яті механізмом DMA може бути повністю прихована за обчисленнями GPU, за умови використання двох буферів даних. Це дозволяє виконувати обчислення GPU над 1мс даних на одному буфері, поки механізм DMA копіює наступні 1мс даних в інший буфер і навпаки. Для реалізації цього паралелізму використовуються асинхронні потоки обчислень, наприклад HIP/ROCm та заблокована пам'ять.

Після паралельного виконання всіх операцій для вхідних даних тривалістю 1мс, CPU також виконує підготовку наступних 1мс даних та оновлення графічного інтерфейсу користувача. Ці додаткові дані займають близько 0.16 мс Рис.3.4. Загальний час обробки даних тривалістю 1 мс становить близько 0.75 мс, що підтверджує можливість роботи GPS SDR для CRPA в режимі реального часу з достатнім запасом часу.

Висновки

У цьому розділі було розглянуто проблему вразливості глобальних навігаційних супутникових систем до радіочастотних завад, що обґрунтувало необхідність проєктування адаптивного програмно-конфігурованого приймача (SDR) з використанням багатоелементної антени (CRPA) для забезпечення стійкості. Детально описано архітектуру GPS SDR для обробки CRPA та обчислювальних задач, акцентуючи увагу на надзвичайно високій інтенсивності операцій необхідних для фазової калібровки. Для вирішення цього було описано розробку GPS SDR для CRPA на гібридній платформі, на базі AMD Radeon RX 6600 XT. Було описано ефективну реалізацію передачі та синтезу даних, а також функціонування паралельного корелятора на основі GPU з використанням паралельного скорочення в перерозподіленій спільній пам’яті, що значно прискорює підсумовування результатів кореляції. Також було проаналізовано апаратний паралелізм між CPU та GPU.

РОЗДІЛ 4
МОДЕЛЮВАННЯ АДАПТИВНОЇ АНТЕНИ ДЛЯ ПРИЙМАЧА СИСТЕМ ГЛОБАЛЬНОЇ СУПУТНИКОВОЇ НАВІГАЦІЇ

4.1. Імітаційні GPS моделі у середовищі MATLAB

4.1.1. Основи GPS

GPS (Global Positioning System) – супутникова навігаційна система, що дає можливість користувачам визначати своє точне місцеположення. Складається з 3-х сегментів: космічного, сегменту керування, сегменту користувачів. Космічний сегмент GPS складається із 31-го супутника, що знаходяться в 24-х слотовій констеляції на середній орбіті Землі на висоті 20 200 км. Основними функціями супутників є прийом та зберігання даних, що надаються керуючим сегментом. За допомогою бортових атомних годинників супутники визначають точний час і поширюють інформацію та сигнали користувачам через частоти. Таблиця 4.1.

Таблиця 4.1. [6]

Основні частотні діапазони та характеристики сигналів системи GPS

Діапазон	Частота	Опис
L1	1575,42 МГц	Коди C/A, P, L1 цивільний (L1C) та військовий (M)
L2	1227,60 МГц	P, L2C та M-код
L3	1381,05 МГц	Використовується для виявлення ядерної детонації
L4	1379,91 МГц	Вивчали для іоносферних поправок
L5	1176,45 МГц	Підтримка програм, критично важливих для безпеки життя цивільного населення (SoL)

Сегмент керування складається з мережі станцій стеження, головних станцій керування та наземних антен. Тут здійснюється визначення та

прогнозування положення супутників GPS, дається оцінка цілісності системи, роботи бортових атомних годинників, збір атмосферних даних, оновлення супутникового альманаху. Потім ці дані передаються на супутники за допомогою S-діапазону. Засоби моніторингу розташовані стратегічно по всьому світу.

Сегмент користувачів включає в себе пристрої оснащені можливостями GPS. Ці GPS-приймачі створенні для перетворення сигналів від супутників на точні оцінки місцезнаходження, швидкості та часових даних.

Робота GPS базується на принципах визначення дальності та трилатерації, використовуючи супутники як точні точки відліку. Приймач завантажує орбітальну інформацію (ефемериди та альманах) від супутників. Супутники, оснащені атомними годинниками, безперервно транслюють сигнали, що містять точний час передачі та інформацію про своє положення. Приймач обчислює відстань до кожного супутника за формулою:

$$\text{Відстань} = \text{Швидкість світла} \times (\text{час отримання сигналу} - \text{час передачі сигналу})$$ [6]

Для визначення 2D положення (на площині) достатньо 3 супутники, оскільки відстань до кожного формує коло, і 3-тє коло визначає єдину точку перетину. Для 3D-положення (з висотою) та корекції неточності годинника приймача необхідні сигнали щонайменше від 4-х супутників.

4.1.2. Генерація сигналів GPS у середовищі MATLAB

Структура GPS сигналу

Стандарт GPS описує передачу сигналів на двох частотах L1 та L2, які генеруються на основі базової частоти 10,23 МГц. Оскільки усталений сигнал передається на обох частотах, цей приклад MATLAB дозволяє створювати форми хвилі базової смуги для як усталеного сигналу так і для сигналу L2C, не включаючи при цьому інформацію про несучу частоту. Зв'язок між структурою

сигналу за стандартом та опціями генерації в MATLAB проілюстрований на Рис.4.1.

GPS standard IS-GPS-200 specification of L1 signal		GPS standard IS-GPS-200 specification of L2 signal	
In-Phase	Quadrature-Phase	In-Phase	Quadrature-Phase
P(Y) ⊕ D(t)	C/A ⊕ D(t)	P(Y) ⊕ D(t) Or P(Y)	L2 CM ⊕ Dc(t) with L2 CL Or C/A ⊕ D(t) Or C/A

MATLAB baseband signal type - "legacy"		MATLAB baseband signal type - "l2c"	
In-Phase	Quadrature-Phase	In-Phase	Quadrature-Phase
P(Y) ⊕ D(t) Or P(Y)	C/A ⊕ D(t) Or C/A	P(Y) ⊕ D(t) Or P(Y)	L2 CM ⊕ Dc(t) with L2 CL

Рис.4.1. Зв'язок між структурою сигналу за стандартом та опціями генерації в MATLAB [7]

Ініціалізація даних GPS

Навігаційні дані GPS (LNAV та CNAV) передають важливу інформацію про годинник і положення супутників, але мають різні формати кадру.

LNAV-дані організовані в кадри по 1500 біт (5 підкадрів по 300 біт). Оскільки швидкість даних становить 50 біт/с, передача цілого кадру займає 30 секунд. Кожен підкадр складається з 10 слів по 30 біт. Рис.4.2.

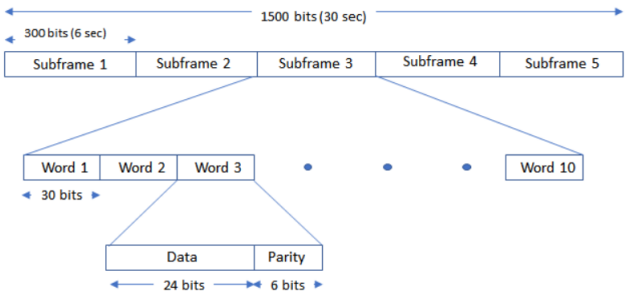

Рис.4.2. Структура кадру даних LNAV [7]

CNAV-дані передаються безперервно як типи повідомлень. Вихідні 300 біт при 25 біт/с кодуються згортковим кодером, що збільшує їх до 600 біт при 50 біт/с. передача одного типу повідомлення займає 12 секунд (Рис.4.3.).

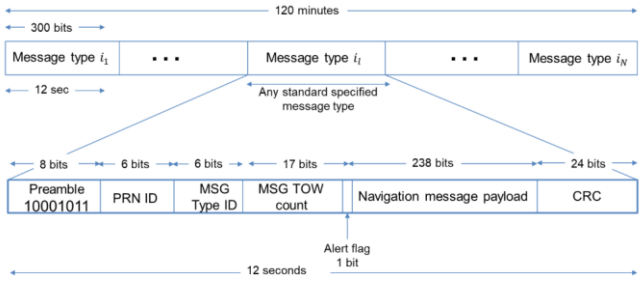

Рис.4.3. Структура повідомлення CNAV [7]

```
cnavConfig = HelperGPSNavigationConfig(ТипСигналу = "CNAV" , PRNID = PRNID)
```

Рис.4.4. Ініціалізація через об’єкт HelperGPSNavigationConfig [7]

Орбітальне положення кожного супутника GPS описується ефемеридами. Ці параметри, як-от довжина великої пів осі, кут нахилу передаються супутником (у підкадрах LNAV або повідомленнях CNAV) і використовуються приймачем для розрахунку місцеположення. Подібний набір конфігураційних властивостей існує для підготовки даних LNAV, для яких в MATLAB створюється об’єкт конфігурації. Рис.4.5.

```
lnavConfig = HelperGPSNavigationConfig(ТипСигналу = "LNAV" , PRNID = PRNID)
```

Рис.4.5. Об'єкт конфігурації для зберігання даних LNAV [7]

Генерація GPS-сигналу

Для створення GPS-сигналу у базовій смузі потрібно:

1. Сформувати біти навігаційних даних швидкість 50 біт/с
2. На основі встановлених параметрів відтворити C/A-код, P-код, L2 CM-/L2 CL-код або їх поєднання. Відбувається в gpsWaveformGenerator

3. Розширити біт даних CNAV або LNAV за допомогою відповідних делокомірних кодів. Відбувається в gpsWaveformGenerator
4. Співставити біти на двох каналах так: біт 0 з +1 та біт 1 з –1. Відбувається в gpsWaveformGenerator
5. Накопичувати інформацію для синфазного (I) та квадратурно-фазного (Q) каналів шляхом узгодження швидкості кодів у кожному каналі. Відбувається в gpsWaveformGenerator
6. Зафіксувати цю форму сигналу базової смуги у файл, залежно від writeWaveformToFilезначення властивості [7]

Візуаліція сигналу

Візуалізація вмикається showVisualizations = true. Рис.4.6.

```
if showVisualizations
    IBranchData = real(gpsBBWaveform);
    QBranchData = imag(gpsBBWaveform);
    corrdata = QBranchData(1:1e-3*sampleRate,1); % Auto-correlate with 1 millisecond of data
    lags = linspace(-1023,1023,2*size(corrdata,1)-1);
    plot(lags,xcorr(corrdata))
    grid on
    xlabel("Number of C/A-code chips Delayed")
    ylabel("Autocorrelation")
    title("Autocorrelation of GPS Spreading Code")
end
```

Рис.4.6. Візуалізація сигналу [7]

4.1.3. Захоплення та відстеження GPS-приймача

Встановлення початкових параметрів

Спочатку потрібно вказати тип сигналу:

signalType = "GPS C/A" ; % Можливі значення: "GPS C/A" | "GPS L1CD" | "GPS L5I" | "GPS L5Q" [8]

Потім вказати про намір отримувати пошук лише з видимих супутників, поставивши галочку біля acquireActiveSatellitesOnly. Потім потрібно вказати кількість видимих супутників: numSat = n; кількість псевдовипадкового шуму (PRN) видимих супутників: PRNIDs = [n1; n2; n3; n4]; кількість бітів даних для обробки в цьому прикладі: numDataBits = n; доплерівський ефект, затримку та співвідношення сигнал/шум (SNR) для кожного з видимих супутників та

встановити проміжну частоту та частоту дискретизації сигналу. Також під час здійснення збору даних потрібно встановити розмір кроку частоти на дійсне додатне число і потім встановити ширину смуги пропускання шуму петлі для фазового автопідстроювання частоти (ФАПЧ), частотного автопідстроювання частоти (ФППЧ) та автопідстроювання частоти із затримкою (ДППЧ). Та потрібно встановити початкове значення seed = n; [8]

Налаштування симуляції

Потрібно створити об’єкт для генерації форми хвилі, використовуючи задані початкові параметри. Рис.4.7.

```
gpswavegen = gpsWaveformGenerator(PRNID=PRNIDs,SampleRate=SampleRate);

% Configure various parameters based on the signalType.
switch(signalType)
    case "GPS C/A"
        numBitsPerStep = 1;
        numCodeBlocksPerBit = 20;
        samplesPerCodeBlock = SampleRate*1e-3;
        gpswavegen.EnablePCode = true;
        gpswavegen.HasDataWithPCode = true;
        gpswavegen.HasDataWithCACode = true;
        gpswavegen.SignalType = "legacy";
        navDataType = "LNAV";
        defaultAcqFqyStep = 500;
    case "GPS L1CD"
        numBitsPerStep = 2;
        numCodeBlocksPerBit = 1;
        samplesPerCodeBlock = SampleRate*10e-3;
        gpswavegen.SignalType = "l1c";
        navDataType = "CNAV2";
        defaultAcqFqyStep = 50;
    case {"GPS L5I" "GPS L5Q"}
        numBitsPerStep = 1;
        numCodeBlocksPerBit = 10;
        samplesPerCodeBlock = SampleRate*1e-3;
        gpswavegen.SignalType = "l5";
        navDataType = "L5";
        defaultAcqFqyStep = 500;
    otherwise
        error("Invalid signalType.")
end
numCodeBlocks = numDataBits*numCodeBlocksPerBit;
numSteps = numDataBits/numBitsPerStep;
numCodeBlocksPerStep = numCodeBlocksPerBit*numBitsPerStep;
if acquisitionFrequencyStep == -1 % Set default value
    acqfqystep = defaultAcqFqyStep;
else
    acqfqystep = acquisitionFrequencyStep;
end

if writeWaveformToFile == true
    % Initialize the baseband file writer object
    bbwriter = comm.BasebandFileWriter("gpsWaveform.bb",SampleRate,IntermediateFrequency);
end
```

Рис.4.7. Налаштування симуляції [8]

Отримання сигналів GPS

Потрібно згенерувати форму сигналу, придатну для захоплення, і використати об’єкт захоплення приймача для її обробки Рис.4.8

```
% Initialize data bits for the waveform
b = transmittedBits(1:numBitsPerStep,:);
% Generate waveform for those data bits
txwaveform = gpswavegen(b);

% pass the transmitted signal through the channel
rxwaveform = gnssChannel(txwaveform);

if writeWaveformToFile == true
    bbwriter(rxwaveform)
end
% Perform acquisition
if acquireActiveSatellitesOnly == true
    prntable = gsa(rxwaveform(end-2*samplesPerCodeBlock+1:end),PRNIDs)
else
    % Searching for all satellites requires more system memory
    prntable = gsa(rxwaveform(end-2*samplesPerCodeBlock+1:end),1:32)
end
```

Рис.4.8. Отримання сигналів GPS [8]

Відстеження сигналів GPS

Для цього потрібно виконати створення та налаштування об’єкту відстеження сигналу. Рис.4.9.

```
if isempty(PRNIDsToSearch)
    % Because no satellites are detected, stop the simulation
    return
end
carrierCodeTrack                        = gnssSignalTracker;
carrierCodeTrack.GNSSSignalType         = signalType;
carrierCodeTrack.SampleRate             = SampleRate;
carrierCodeTrack.IntermediateFrequency  = IntermediateFrequency;
carrierCodeTrack.PLLNoiseBandwidth      = PLLNoiseBandwidth;
carrierCodeTrack.FLLNoiseBandwidth      = FLLNoiseBandwidth;
carrierCodeTrack.DLLNoiseBandwidth      = DLLNoiseBandwidth;
carrierCodeTrack.PRNID                  = PRNIDsToSearch;
carrierCodeTrack.InitialFrequencyOffset = dopplerоffsets;
carrierCodeTrack.InitialCodePhaseOffset = codephoffsets;
longDisplay(carrierCodeTrack)
```

Рис.4.9. Ініціалізація об’єкту відстеження сигналу [8]

4.1.4. Наскрізний GPS-навігаційний приймач старого покоління з використанням C/A-коду

Ініціалізація параметрів

Включає налаштування тривалості симуляції, частоти дискретизації, позиції приймача, параметрів трекінгу, синхронізації та фізичних констант. Встановлюється значення для simulatedDataDuration, samplingRate,

rxlat/rxlon/rxalt, шумових смуг PLL/FLL/DLL, мінімального часу для оцінки позиції та констант як швидкість світла, частота L1, потужність передачі.

Конфігурація симуляції

Конфігурація створює сценарій супутників за допомогою satelliteScenario та rinexdata, встановлює позицію приймача, генерує навігаційні дані через HelperGPSRINEX2Config та HelperGPSDataEncode, обчислює доплерівський зсув, затримки та SNR для видимих супутників. Ініціалізуються генератор хвиль gpsWaveformGenerator та канал HelperGNSSChannel, а також обєкт синхронізації gnssSignalAcquirer. [9]

Наскрізний ланцюг моделювання

Наскрізний ланцюг реалізує генерацію хвилі в блоках по 1 мс, моделювання каналу з доплерівським зсувом, затримкою та шумом, обробку приймачем, а саме: початкова синхронізація, трекінг, синхронізація бітів/фреймів та декодування даних. Симуляція виконується в кроках по 1 мс, з початковою синхронізацією після затримки, трекінгом через carrierCodeTrack та декодуванням через HelperGPSLNAVDataDecode.

Оцінка положення GPS-приймача

Оцінка позиції використовує декодовані дані для обчислення часу передачі, псевдодалей, декодування ефемерид та позицій супутників через gnssconstellation на основі псевдодалей, позицій супутників та маски висоти.

Модель на Рис.4.10. створює значення X, Y, Z для позиції та швидкості у вигляді окремих синусоїдальних хвиль, а потім об’єднує їх за допомогою блоків Mux. Оскільки блок GPS вимагає дискретних сигналів, об’єднані дані позиції та швидкості пропускаються через блоки Rate Transition для підключення до портів Position і Velocity блоку GPS. Блок GPS використовує стандартні налаштування параметрів, за винятком точності вертикальної позиції, яка встановлена на 1,5 через особливості масштабу позиції та швидкості.

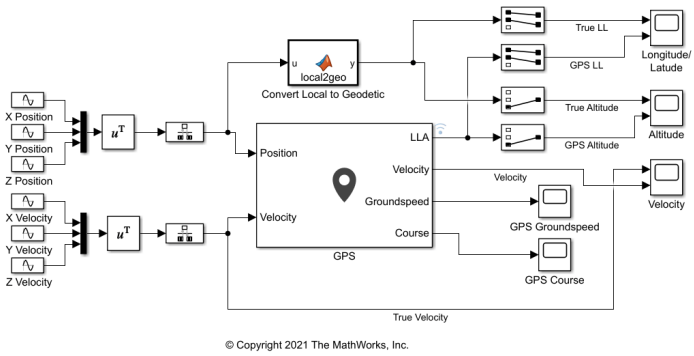

Рис.4.10. Модель шуму GPS-сенсора [10]

Для порівняння виходів блоку GPS зі справжніми значеннями сигналів застосовуються блоки Scope. У випадку з позицією локальні координати потрібно перетворити на координати LLA. Для цього в блоці MATLAB Function використовується функція ned2lla, яка конвертує координати NED у LLA. [10]

4.1.5. Захоплення та відстеження GPS-приймача за допомогою C/A-коду

Налаштування параметрів моделювання передбачає встановлення всіх конфігураційних параметрів, що керують роботою прикладу. Спочатку задаються загальні параметри, такі як прапорець ShowVisualizations, який активує візуалізації та WriteWaveformToFile, що контролює запис згенерованої хвилі в файл.

У конфігурації передавача вказується кількість супутників GPS: numSat = 4; ідентифікатори PRN для видимих супутників: PRNIDs = [n1; n2; n3; n4]; початковий індекс бітів навігаційних даних: NavDataBitStartIndex = n; та кількість бітів NumNavDataBits = 20; Також визначається центральна частота IF: CenterFrequency = 10e6; та частота дискретизації SampleRate = 38.192e6; [11]

Для конфігурації каналу задаються значення SNR для кожного супутника, з урахуванням мінімальної потужності сигналу – 158.5 дБВт та теплового шуму близкл – 130 дБВт, затримки в чіпах C/A-коду sigdelay = [n1; n2; n3; n4]; для імітації різних відстаней, пікові доплерівські зсуви peakDoppler = [n1; n2; n3; n4]; та їхні швидкості змін dopplerRate = [n1; n2; n3; n4]; [11]

У конфігурації приймача налаштовуються смуги пропускання шумів для циклів відстеження: PLLNoiseBandwidth = 90; (фазового), FLLNoiseBandwidth = 4; (частотного), DLLNoiseBandwidth = 1; (для затримки) [11]

Процеси в GPS-приймачі

Мета початкової синхронізації – виявлення видимих супутників та оцінка грубого доплерівського зсуву частоти та зсуву фази коду в сигналі C/A-коду. Алгоритм базується на паралельному пошуку по фазі коду з використанням швидкого перетворення Фур'є (FFT) . C/A-код перетворюється в частотну область, множиться на спектр сигналу, а потім застосовується зворотне FFT для повернення в часовий домен. При цьому приймач обробляє IF-хвилю, для кожного PRN ID здійснюється пошук по діапазону доплера та фаз коду. Обчислюються кореляційні значення для виявлення піків, що показують на наявність сигналу.

Після пошуку здійснюється підтримка захвату для кожного супутника шляхом постійної оцінки та кореляції зсуву частоти несучої (FLL), фази несучої (PLL) та затримки фази коду (DLL). Таблиця 4.2. Кожен параметр відстежується незалежно за допомогою спеціальних петель. Структура петлі включає в себе комбайнер, дискримінатор, фільтр петлі та генератор контрольного сигналу. Ширша смуга петлі забезпечує швидке сходження, але чутлива до шуму, вужча стабільніша при низькому SNR, але повільніша.

Таблиця 4.2. [11]

Характеристики дискримінаторів та петлевих фільтрів систем PLL, FLL та DLL

	Алгоритм дискримінатора	Ширина полоси пропускання контурного фільтра (Гц)	Порядок циклу
PLL	2 quadrant atan	90	2
FLL	4 quadrant atan2	4	1
DLL	(E-L)/(2*(E+L))	1	1

Для багатосупутникового відстеження використовуємо об’єкт gnssSignalTracker. Рис.4.11.

```
    carrierCodeTrack = gnssSignalTracker;
    carrierCodeTrack.SampleRate = SampleRate;
    carrierCodeTrack.IntermediateFrequency = CenterFrequency;
    carrierCodeTrack.PLLNoiseBandwidth = PLLNoiseBandwidth;
    carrierCodeTrack.FLLNoiseBandwidth = FLLNoiseBandwidth;
    carrierCodeTrack.DLLNoiseBandwidth = DLLNoiseBandwidth;
    carrierCodeTrack.IntegrationTime = IntegrationTime;
    carrierCodeTrack.PRNID = PRNIDsToSearch;
    carrierCodeTrack.InitialFrequencyOffset = doppleroffsets;
    carrierCodeTrack.InitialCodePhaseOffset = codephoffsets;
end

[integwave,trackinfo] = carrierCodeTrack(bufferWave);
```

Рис.4.11. Об’єкт gnssSignalTracker [11]

Графіки для кожного супутника – оцінки та помилки для FLL, PLL, DLL, показують сходження та варіації доплера. Також констеляційний графік [11] демодульованого сигналу після нормалізації. Приклад коду для FLL Рис.4.12.

```
if ShowVisualizations == 1
    % for isat = 1:numdetectsat
    for isat = 1 % See tracking results of all the detected satellites by using above line
        groupTitle = "Tracking Loop Results for Satellite PRN ID:" + PRNIDsToSearch(isat);

        figure

        % Plot the frequency discriminator output
        subplot(2,1,1)
        plot(accufqyerr(:,isat))
        xlabel("Milliseconds")
        ylabel("Frequency Error")
        title("Frequency Discriminator Output")

        % Plot the FLL output
        subplot(2,1,2)
        plot(accufqy(:,isat))
        xlabel("Milliseconds")
        ylabel("Estimated Frequency Offset")
        title("FLL Output")
        sgtitle("FLL " + groupTitle)
```

Рис.4.12. Петля відстеження FLL [11]

Після виконання наведеного вище коду для візуалізації результатів петлі захоплення частоти FLL, ми отримуємр наступний графік, який демонструє вихід дискримінатора частоти та оцінену частотну офсету. Рис.4.13. [11]

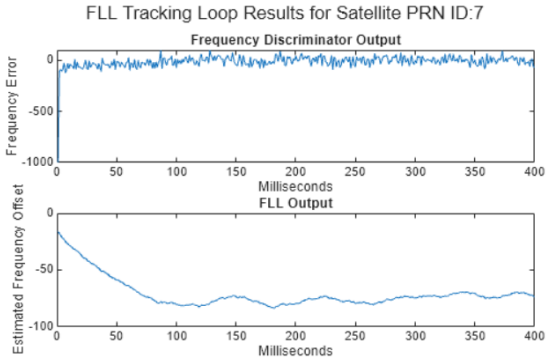

Рис.4.13. Результати петлі відстеження FLL для супутника PRN ID:7 [11]

4.2. MATLAB-моделювання GPS-приймача з адаптивною антенною решіткою

Спочатку, в самому заголовку файлу задається, що це скрипт run_gps_interference_array_mvdr.m, який моделює три сценарії приймача GPS: чистий сигнал з однією антеною, сигнал із завадами з однією антеною, і сигнал із завадами, але з багатоканальною антенною решіткою та просторовою обробкою MVDR, що прямо видно з коментаря на початку коду. Рис.4.14.

```
%% run_gps_interference_array_mvdr.m
%
% Три сценарії GPS-приймача:
%  A) чистий сигнал (1 антена)
%  B) сигнал із завадами (1 антена)
%  C) сигнал із завадами + багатоканальна АР (MVDR)
%
% Створено на основі макету EndtoEndGPSLNAVReceiverExample.

```

Рис.4.14. Три сценарій приймача GPS

Далі одразу готуємо середовище виконання, очищаємо змінні/вікна і фіксуємо генератор випадкових чисел, щоб результати симуляції повторювалися при кожному запуску, що видно в рядках Рис.4.15.

```
clear; close all; clc;
seed = 23;
rng(seed);

```

Рис.4.15. Генератор випадкових чисел

Після цього задаємо параметри симуляції в часовій області: тривалість даних 69 секунд, частота дискретизації 10.23 МГц (типова для обробки C/A-коду), крок моделювання 1 мс, і кількість кроків numSteps, що задано Рис.4.16.

```
%% ------------------ ПАРАМЕТРИ ------------------
simulatedDataDuration = 69;      % seconds
samplingRate = 10.23e6;          % Hz
stepTime = 1e-3;                 % seconds
numSteps = (simulatedDataDuration/stepTime) + 1;

```

Рис.4.16. Параметри симуляції в часовій області

Далі вказуються вихідні дані, на яких базується модель супутників/навігаційних повідомлень: використовується RINEX-файл та альманах GPS, і саме це є вхідними даними експерименту, що видно з Рис.4.17.

```
rinexFileName   = "GODS00USA_R_20211750000_01D_GN.rnx";
almanacFileName = "gpsAlmanac.txt";

```

Рис.4.17. Вихідні дані

Після цього задається місцевість/точка дослідження через координати приймача, тому що приймач у сценарії фіксується за широтою, довготою і висотою над рівнем моря. Рис.4.18.

```
% Положення приймача
rxlat = 39.021;
rxlon = -76.827;
rxalt = 19;

```

Рис.4.18. Положення приймача

Ці значення використовуються далі як істинна позиція приймача (ground truth) у порівнянні з оцінками позиціонування, а також вони передаються в

groundStation(sc, rxlat, rxlon) і в pseudoranges([rxlat,rxlon,rxalt], ...), тобто геометрія супутник-приймач моделюється відносно саме цієї точки.

Далі задаються параметри ведення супутників і приймача: скільки блоків C/A-коду на 1 навігаційний біт (20), час очікування перед запуском трекінгу 100 мс, прапорець початкової синхронізації, а також максимальна кількість трекінг-каналів 8 (це кількість супутників, які одночасно буде вести приймач у моделі). Рис.4.19.

```
% Параметри ведення супутників
numCACodeBlocksPerBit = 20;
rxWaitTime = 100;                % ms
performInitSync_init = 1;
maxNumTrackingChannels = 8;

```

Рис.4.19. Параметри виведення супутників

Потім задаються смуги шуму контурів PLL/FLL/DLL для моделювання трекінгу, це впливає на динаміку помилки частоти/фази/затримки в трекері. Рис.4.20.

```
PLLNoiseBW = 90;  % Hz
FLLNoiseBW = 4;   % Hz
DLLNoiseBW = 3;   % Hz

```

Рис.4.20. Смуги шуму контурів PLL/FLL/DLL

Далі задаються параметри побітної синхронізації: скільки бітів використовувати для bit sync, і скільки мілісекундних кроків треба накопичити, щоб виконати синхронізацію. Рис.4.21.

```
% Параметри побітної синхронізації
numBitsForBitSync = 100;
numWaitingStepsForBitSync = numCACodeBlocksPerBit * numBitsForBitSync;

```

Рис.4.21. Параметри побітної синхронізації

Після цього визначається мінімальний час, після якого дозволяємо оцінювати позицію (48.5 секунд), і переводимо його в кроки симуляції. Рис.4.22.

```
41
42      minTimeForPosEst = 48.5;          % seconds
43      minStepsForPosEst = minTimeForPosEst/stepTime;
44
```

Рис.4.22. Мінімальний необхідний час

Далі задаємо параметри тривалості субкадру LNAV (6 секунд) і кількість кроків на субкадр. Рис.4.23.

```
44
45      subframeDuration = 6;             % seconds
46      numStepsPerSubframe = subframeDuration/stepTime;
47
```

Рис.4.23. Параметри тривалості субкадру LNAV

Після цього вводяться фізичні константи та частота L1, які будуть потрібні і для хвильових параметрів антенної решітки (довжина хвилі), і для доплеру. Рис.4.24.а)

Далі вводяться параметри лінії зв’язку: підсилення приймача в dB та лінійно, потужність Pt, стала Больцмана, температура, смуга приймача, та теплова потужність шуму Nr, щоб порахувати SNR для кожного супутника. Рис.4.24.б)

```
47
48      % Константи
49      c  = physconst("LightSpeed");
50      fe = 1575.42e6;
51
```

а)

```
51
52      % Ресурси лінії зв'язку
53      Dt = 12; DtLin = db2pow(Dt);
54      Dr = 4;  DrLin = db2pow(Dr);
55      Pt = 44.8;                                  % W
56      kB = physconst("boltzmann");
57      T0 = 300;
58      rxBW = samplingRate;
59      Nr = kB*T0*rxBW;
60
```

б)

Рис.4.24. а) Хвильові параметри антенної решітки; б) Параметри лінії зв’язку

Далі задаються саме завади, і тут чітко видно, що модель включає кілька завад одночасно, а їх назви прямо зашиті в структурі intf: гармонічна (tone), вузькосмугова модульована (narrowband), та гаусівський шум (AWGN). Рис.4.25.

```
%% ------------------ ПАРАМЕТРИ ЗАВАД ------------------
% 1) Гармонічна
intf.tone.freq = 200e3;      % Hz (baseband)
intf.tone.amp  = 8;

% 2) Вузькосмугова модульована
intf.narrow.bw  = 40e3;      % Hz
intf.narrow.amp = 6;

% 3) Гаусівський шум
% Інтерпретуємо як "додатковий" AWGN до композиту
intf.awgn.snr_dB = 0;        % 0 dB -> сильна завада; 10-20 dB -> слабша

```

Рис.4.25. Параметри завад

Далі задається момент появи завад: перші 18 секунд завад немає, а потім завади вмикаються, що видно з Рис.4.26.

```

jamStartSec = 18;                % перші 18 секунд без завад
jamStartStep = round(jamStartSec/stepTime);

```

Рис.4.26. Момент появи завад

Після цього задаються параметри антенної решітки та MVDR. Тут прямо вказано кількість елементів решітки numArrayElements = 6, а також крок між елементами як половина довжини хвилі на L1. Рис.4.27.

```
%% ------------------ ПАРАМЕТРИ МАСИВУ / MVDR ------------------
useArray = true;
numArrayElements = 6;

lambdaL1 = c/fe;
arrayElementSpacing = 0.5*lambdaL1;   % ~0.095 м

```

Рис.4.27. Параметри антенної решітки

Далі задається напрямок приходу завади (DOA) як штучно вибрані азимут і кут місця. Це важливо: у просторовій моделі задається один напрямок doaInt, тобто всі згенеровані складові завад далі в масиві трактуються як такі, що приходять з одного напрямку. Рис.4.28.

```
83
84      % "напрямок" інтерферера (задаємо штучно)
85      doaInt.az = deg2rad(45);
86      doaInt.el = deg2rad(20);
87
```

Рис.4.28. Напрямок проходу завади

Після цього задається коефіцієнт експоненційного усереднення коваріаційної матриці R_array, щоб MVDR пам’ятав статистику перешкод у часі, а не реагував лише на один 1-мс блок. Рис.4.29.

```
87
88      % Експоненційне усереднення коваріації
89      alphaR = 0.01;
90
```

Рис.4.29. Коефіцієнт експоненційного усереднення коваріації

Далі починається підготовка навігаційних даних: завантаження RINEX через rinexread, створення satelliteScenario, додавання супутників за RINEX, додавання наземної станції в координатах rxlat, rxlon, і визначення доступності супутників. Рис.4.30.

```
91      %% ------------------ ПІДГОТОВКА RINEX / NAV ------------------
92      disp("Завантаження RINEX...");
93      rinexdata = rinexread(rinexFileName);
94
95      % Визначення супутників для навконфіга
96      sc = satelliteScenario;
97      sat = satellite(sc, rinexdata, "OrbitPropagator", "gps");
98      rx  = groundStation(sc, rxlat, rxlon);
99
100     ac = access(sat, rx);
101
```

Рис.4.30. Підготовка навігаційних даних

Далі код підбирає правильні індекси записів навігаційних даних RINEX для кожного супутника, порівнюючи Toe та PRN, і формує конфігурацію navcfg через HelperGPSRINEX2Config, використовуючи альманах та відібрані ефемериди. Рис.4.31.

```
101
102  indices = ones(length(sat),1);
103  for isat = 1:length(sat)
104      ele = orbitalElements(sat(isat));
105      indices(isat) = find( ...
106          rinexdata.GPS(:,:).Toe == ele.GPSTimeOfApplicability & ...
107          rinexdata.GPS(:,:).SatelliteID == ele.PRN);
108  end
109
110  navcfg = HelperGPSRINEX2Config(almanacFileName, rinexdata.GPS(indices,:));
111
```

Рис.4.31. Дані RINEX

Далі виконується синхронізація HOWTOW, щоб усі супутники стартували узгоджено по часу слова/кадру, для цього береться мінімальний HOWTOW і округлюється до кратності, після чого це значення присвоюється всім супутникам. Рис.4.32.а)

Далі обчислюються індекси кадрів LNAV і також присвоюються всім супутникам. Рис.4.32.б)

```
112  % Синхронізація HOWTOW
113  [mintow, locmintow] = min([navcfg(:).HOWTOW]);
114  mintow = ceil((mintow-1)/5)*5 + 1;
115  [navcfg(:).HOWTOW] = deal(mintow);
116
```

а)

```
116
117  firstsubframeID = mod(mintow-1,125) + 1;
118  frameID = ceil(firstsubframeID/5);
119  allFrameIDs = [frameID:25, 1:(frameID-1)];
120  [navcfg(:).FrameIndices] = deal(allFrameIDs);
121
```

б)

Рис.4.32. а) Синхронізація HOWTOW; б) Індекси кадрів LNAV

Після цього формується послідовність навігаційних бітів navdata на 37500 біт для кожного супутника через HelperGPSNAVDataEncode. Рис.4.33.

```
122  % Навдані
123  numNavBits = 37500;
124  navdata = zeros(numNavBits, length(navcfg));
125  for isat = 1:length(navcfg)
126      navdata(:,isat) = HelperGPSNAVDataEncode(navcfg(isat));
127  end
128
```

Рис.4.33. Послідовність навігаційних бітів

Далі налаштовується часовий інтервал satelliteScenario: стартовий час з GPS-тижня і mintow, стоп-час як старт плюс 69 секунд, і частота семплювання сценарію 1 мс. Рис.4.34.

```
128
129     % Налаштування satelliteScenario часу
130     sc.StartTime  = HelperGPSConvertTime(navcfg(locmintow).WeekNumber, mintow*subframeDuration);
131     sc.StopTime   = sc.StartTime + seconds(simulatedDataDuration);
132     sc.SampleTime = stepTime;
133
```

Рис.4.34. Налаштування часового інтервалу

Потім визначаються видимі супутники через accessStatus, перевіряється, що їх не менше 4, інакше позицію не визначити, і виводяться їх PRN. Рис.4.35.а)

Далі обчислюється доплерівський зсув для кожного супутника відносно приймача, а також знімаються координати супутників у ECEF на кожному кроці. Потім для кожного кроку рахується псевдовідстань і затримка delays = satdist / c. Рис.4.35.б)

```
133
134     acstats = accessStatus(ac);
135     satindices = find(acstats(:,1));
136     numsat = length(satindices);
137     if numsat < 4
138         error("Мало видимих супутників: %d", numsat);
139     end
140
141     PRNIDs = [navcfg(:).PRNID];
142     disp("Available satellites - " + num2str(PRNIDs(satindices)));
143
```

а)

```
143
144     % Доплер
145     fShift = dopplershift(sat(satindices), rx, Frequency=fe);
146
147     initsatpos = permute(states(sat,"CoordinateFrame","ecef"), [3 1 2]);
148
149     % Дистанції/затримки
150     satdist = zeros(numsat, numSteps);
151     for istep = 1:numSteps
152         satdist(:,istep) = pseudoranges([rxlat,rxlon,rxalt], ...
153                 initsatpos(satindices,:,istep), "RangeAccuracy",0);
154     end
155     delays = satdist / c;
156
```

б)

Рис.4.35. а) Визначення видимих супутників; б) Обчислення доплерівського шуму

Далі оцінюється отримана потужність Pr і SNR для кожного супутника на кожному кроці. Це SNR далі подається в канал як SignalToNoiseRatio. Рис.4.36.

```
156
157     % Потужність/CNR
158     Pr  = (Pt*DtLin*DrLin) .* (1 ./ (4*pi*(fe + fShift) .* delays));
159     SNRs = 10*log10(Pr/Nr);
160
```

Рис.4.36. Оцінка SNR

Після цього створюється генератор GPS-хвилі gpsWaveformGenerator для legacy-сигналу з PRN видимих супутників, а також канал HelperGNSSChannel, який моделює доплер, затримку і шум. Далі визначається, що один блок C/A-коду має 1 мс, і скільки семплів у блоці. Рис.4.37.

```
161     %% ------------------ ГЕНЕРАТОР СИГНАЛУ/КАНАЛ ------------------
162     gpswavegen = gpsWaveformGenerator( ...
163         SignalType="legacy", PRNID=PRNIDs(satindices), SampleRate=samplingRate);
164
165     gnssChannelObj = HelperGNSSChannel( ...
166         SampleRate=samplingRate, RandomStream="mt19937ar with seed", Seed=seed);
167
168     caCodeBlockDuration = 1e-3;
169     SamplesPerCodeBlock = caCodeBlockDuration * samplingRate;
170     numcacodeblocks = stepTime / caCodeBlockDuration;
171
```

Рис.4.37. Створення генератора GPS-хвилі

Далі готується об’єкт acquisition для початкового захоплення, де проміжна частота 0, тобто одразу baseband. Рис.4.38.

```
171
172     % Acquisition obj
173     initialsync = gnssSignalAcquirer;
174     initialsync.SampleRate = samplingRate;
175     initialsync.IntermediateFrequency = 0;
176
```

Рис.4.38. Об’єкт acquisition

Після цього створюються три стани приймача, для трьох сценаріїв через initRxState, і в кожному виставляється прапорець початкової синхронізації. Рис.4.39.

```
178     %% ------------------ ПІДГОТОВКА ТРЬОХ RX-СТАНІВ ------------------
179     rxA = initRxState(maxNumTrackingChannels, numSteps);  % clean
180     rxB = initRxState(maxNumTrackingChannels, numSteps);  % jammed
181     rxC = initRxState(maxNumTrackingChannels, numSteps);  % jammed + MVDR
182
183     rxA.performInitSync = performInitSync_init;
184     rxB.performInitSync = performInitSync_init;
185     rxC.performInitSync = performInitSync_init;
186
```

Рис.4.39. Підготовка трьох станів приймача

Далі вводиться час в тижні, як стартовий HOWTOW помножений на 6 секунд, бо HOWTOW тут прив'язаний до субкадрів тривалістю 6 с, і задаються координати елементів решітки. Важливо, що решітка моделюється як ULA вздовж осі x, а координати елементів формуються матрицею arrayCoords розміру numArrayElements x 3. Рис.4.40.

```
186
187     % Час у тижні
188     timeinweek = mintow*6;
189
190     % Масив координат (ULA по осі x)
191     arrayCoords = [(0:numArrayElements-1).' * arrayElementSpacing, ...
192                    zeros(numArrayElements,1), zeros(numArrayElements,1)];
193
```

Рис.4.40. Час у тижні

Далі ініціалізується коваріаційна матриця масиву R_array (розмір 6x6), і задається steering-вектор корисного сигналу sv_sig як рівномірний, тобто дивимось так, ніби корисний сигнал приходить фронтом і однаково на всі елементи), що задано Рис.4.41.

```
194     % Коваріація масиву
195     R_array = zeros(numArrayElements, numArrayElements);
196
197     % Напрямок корисного сигналу:
198     % Візьмемо LOS на перший видимий супутник у першому кроці як "look"
199     sv_sig = ones(1, numArrayElements) / sqrt(numArrayElements);
200
```

Рис.4.41. Коваріація масиву

Далі формується steering-вектор завади sv_int з DOA, координат решітки, частоти fe і швидкості світла, що визначає фазовий нахил плоскої хвилі на елементах. Рис.4.42.

```
201     % Steering завади
202     sv_int = steeringVectorFromAzEl(doaInt.az, doaInt.el, arrayCoords, fe, c);
203
```

Рис.4.42. Steering завади

Далі починається основний цикл симуляції крок за кроком по 1 мс. Тут визначається, який навігаційний біт зараз передається (bitidx), і який номер C/A-блоку всередині біта (CodeBlockNumber). На початку кожного біта генерується форма хвилі OnebitWaveform, а потім з неї вирізається поточний 1-мс шматок iqsig. Рис.4.43.

```
204  %% ------------------ ОСНОВНИЙ ЛАНЦЮЖОК ------------------
205  disp("Start End-to-End Simulation Chain (3 scenarios)...");
206  tic;
207
208  for istep = 1:numSteps
209
210      %% ---------- Generate waveform ------------
211      bitidx = floor((istep-1)/numCACodeBlocksPerBit) + 1;
212      CodeBlockNumber = mod(istep-1, numCACodeBlocksPerBit);
213
214      if CodeBlockNumber == 0
215          OnebitWaveform = gpswavegen(navdata(bitidx, satindices));
216      end
217
218      iqsig = OnebitWaveform( ...
219          CodeBlockNumber*SamplesPerCodeBlock + (1:SamplesPerCodeBlock), :);
220
```

Рис.4.43. Основний цикл симуляції

Після цього 1-мс блок проходить канал: у канал на кожному кроці передаються доплер, затримка і SNR, і на виході отримується композитний сигнал waveform, який містить суму сигналів супутників плюс шум каналу. Рис.4.44.

```
221      %% ---------- Pass through channel ------------
222      % Як у оригіналі
223      gnssChannelObj.FrequencyOffset     = fShift(:,istep).';
224      gnssChannelObj.SignalDelay         = delays(:,istep).';
225      gnssChannelObj.SignalToNoiseRatio  = SNRs(:,istep).';
226
227      waveform = gnssChannelObj(iqsig);  % очікуємо композитний сигнал
228
```

Рис.4.44. Композитний сигнал waveform

Далі оцінюється потужність композитного сигналу sigp, щоб масштабувати гаусівську складову завади, і оновлюється змінна часу. Рис.4.45.

```
229         % Потужність корисного композиту для масштабування шумів/завад
230         sigp = mean(abs(waveform).^2) + eps;
231
232         timeinweek = timeinweek + stepTime;
233
```

Рис.4.45. Оцінка потужності композитного сигналу

Далі формується блок завад interference_block. Якщо крок ще до моменту jamStartStep, то завада нульова. Інакше створюється часовий вектор tblock, і будуються 3 незалежні складові: гармонічна tone_block, вузькосмугова модульована narrow_block, та гаусівський шум awgn_block. Рис. 4.46.

```
234         %% ---------- Build interference for this 1ms block ------------
235         if istep <= jamStartStep
236             interference_block = zeros(SamplesPerCodeBlock,1);
237         else
238             tblock = ((0:SamplesPerCodeBlock-1)'/samplingRate) + (istep-1)*caCodeBlockDuration;
239
240             tone_block   = intf.tone.amp * exp(1j*2*pi*intf.tone.freq * tblock);
241             narrow_block = generateNarrowbandBlock(SamplesPerCodeBlock, samplingRate, intf.narrow.bw, intf.narrow.amp);
242
243             noiseP = sigp / (10^(intf.awgn.snr_dB/10));
244             awgn_block = sqrt(noiseP/2) * (randn(SamplesPerCodeBlock,1) + 1j*randn(SamplesPerCodeBlock,1));
245
246             interference_block = tone_block + narrow_block + awgn_block;
247         end
248
```

Рис.4.46. Формування блоку завад

Далі формуються сигнали трьох сценаріїв: для A береться чистий waveform, для B до нього додається interference_block. Для C вмикається модель антенної решітки: корисний композит розкладається на M каналів через sv_sig, а завада також розкладається на M каналів через sv_int, після чого додається невеликий незалежний шум на кожному елементі elemNoise. Рис.4.47.

```
249         %% ---------- Scenario signals ------------
250         waveform_A = waveform;                          % clean
251         waveform_B = waveform + interference_block;     % jammed (single antenna)
252
253         % Array моделінг для сценарію C:
254         % Формуємо M-канальний прийом:
255         %   корисний композит як плоска хвиля з sv_sig
256         %   завада як плоска хвиля з sv_int
257         if useArray
258             % M-канальний блок
259             % (Samples x 1)*(1 x M)
260             sig_array  = waveform * sv_sig;
261             int_array  = interference_block * sv_int;
262
263             % Додатковий незалежний шум на елементах (малий)
264             elemNoise = 0.05 * sqrt(sigp/2) * ...
265                 (randn(SamplesPerCodeBlock, numArrayElements) + 1j*randn(SamplesPerCodeBlock, numArrayElements));
266
267             array_block = sig_array + int_array + elemNoise;
268
```

Рис.4.47. Формування трьох сценаріїв

Далі з array_block оцінюється миттєва коваріація R_inst, коваріаційна матриця усереднюється експоненційно, обчислюються ваги MVDR w_mvdr, і виконується beamforming до одного скалярного каналу waveform_C. Рис.4.48.

```
269             % Оновити коваріацію
270             R_inst = (array_block' * array_block) / size(array_block,1);
271             R_array = (1 - alphaR) * R_array + alphaR * R_inst;
272
273             % MVDR ваги з діагональним навантаженням
274             w_mvdr = mvdrWeights(R_array, sv_sig.');
275
276             % Beamformed scalar
277             waveform_C = array_block * conj(w_mvdr);
278         else
279             waveform_C = waveform_B;
280         end
281
```

Рис.4.48. Оцінка array_block

Решітка з M = numArrayElements = 6 дає 6 ступенів свободи для формування діаграми, а модель містить три складові завад у часі (гармонічну, вузькосмугову модульовану та AWGN). При цьому важлива чесна деталь саме цього коду: у просторовому сенсі задається лише один напрямок приходу завади doaInt, і всі три складові підсумовуються в interference_block та множаться на один steering-вектор sv_int, тобто в масиві вони поводяться як один просторовий джерело (ранг близький до 1), але вони присутні одночасно і реально псують прийом у сценарії B та C.

Далі дані подаються в приймачі для кожного сценарію, але лише після rxWaitTime мілісекунд. На кожному кроці викликається receiverStep для A/B/C, тобто один і той самий блок acquisition+tracking+decode працює на трьох різних вхідних сигналах. Рис.4.49.

```
282         %% ---------- Обробка на приймачі для трьох випадків ------------
283         if istep > rxWaitTime
284
285             rxA = receiverStep(rxA, waveform_A, initialsync, samplingRate, ...
286                 PLLNoiseBW, FLLNoiseBW, DLLNoiseBW, ...
287                 maxNumTrackingChannels, ...
288                 numStepsPerSubframe, numCACodeBlocksPerBit, numWaitingStepsForBitSync);
289
290             rxB = receiverStep(rxB, waveform_B, initialsync, samplingRate, ...
291                 PLLNoiseBW, FLLNoiseBW, DLLNoiseBW, ...
292                 maxNumTrackingChannels, ...
293                 numStepsPerSubframe, numCACodeBlocksPerBit, numWaitingStepsForBitSync);
294
295             rxC = receiverStep(rxC, waveform_C, initialsync, samplingRate, ...
296                 PLLNoiseBW, FLLNoiseBW, DLLNoiseBW, ...
297                 maxNumTrackingChannels, ...
298                 numStepsPerSubframe, numCACodeBlocksPerBit, numWaitingStepsForBitSync);
299
300             maxSimSteps = simulatedDataDuration/stepTime;
301             if istep > maxSimSteps
302                 break;
303             end
304         end
305
```

Рис.4.49. Обробка на приймачі трьох випадків

Далі виконується захист від зайвих кроків. Рис.4.50.

```
305
306         if mod(istep, 1000) == 0
307             disp("Processed " + (istep/1000) + " sec at signal generator.");
308         end
309
310     end
311
```

Рис.4.50.

Після завершення циклу вимірюється час виконання симуляції. Рис.4.51.

```
311
312     e2eTime = toc;
313     disp("End-to-End chain ran for " + e2eTime + " seconds.");
314
```

Рис.4.51. Час виконання симуляції

Далі відбувається оцінка координат приймача для трьох сценаріїв. Тут важливо, що функція estimatePositionFromRxState повертає дві речі: errX як норму помилки в локальній системі NED відносно істинних координат (rxlat, rxlon, rxalt), і rxposX як оцінені координати приймача (формату LLA), тобто саме тут отримуються координати на виході. Аналогічно викликається для B і C. Рис.4.52.

```
%% ------------------ ВИЗНАЧЕННЯ РОЗТАШУВАННЯ ------------------

[errA, rxposA] = estimatePositionFromRxState(rxA, ...
    rxlat, rxlon, rxalt, ...
    minStepsForPosEst, ...
    caCodeBlockDuration, ...
    numCACodeBlocksPerBit, ...
    maxNumTrackingChannels, ...
    rinexFileName);

[errB, rxposB] = estimatePositionFromRxState(rxB, ...
    rxlat, rxlon, rxalt, ...
    minStepsForPosEst, ...
    caCodeBlockDuration, ...
    numCACodeBlocksPerBit, ...
    maxNumTrackingChannels, ...
    rinexFileName);

[errC, rxposC] = estimatePositionFromRxState(rxC, ...
    rxlat, rxlon, rxalt, ...
    minStepsForPosEst, ...
    caCodeBlockDuration, ...
    numCACodeBlocksPerBit, ...
    maxNumTrackingChannels, ...
    rinexFileName);

```

Рис.4.52. Визначення розташування

Після цього код виводить на екран саме помилку відстані, що видно з блоку. Рис.4.53.

```
disp(" ");
disp("Distance error (NED norm):");
disp("A - clean:                " + num2str(errA) + " m");
disp("B - jammed:               " + num2str(errB) + " m");
disp("C - jammed + array MVDR:  " + num2str(errC) + " m");

```

Рис.4.53. Виведення помилки відстані

Далі будується стовпчикова діаграма помилок для трьох сценаріїв. Рис.4.54.

```
%% ------------------ Полотна ------------------
figure('Name','Position error comparison'); clf;
bar([errA, errB, errC]);
set(gca,'XTickLabel',{'clean','jammed','jammed+MVDR'});
ylabel('Position error (m)');
title('GPS position error comparison (3 scenarios)');
grid on;

```

Рис.4.54. Побудова діаграми

Після цього координати оцінок rxposA/B/C переводяться в локальну NED-систему відносно істинних координат trueLLA = [rxlat, rxlon, rxalt], і будується

3D-розсіювання: чорна точка (0,0,0) – істина, а кольорові точки – помилки для A/B/C. Саме тут координати фактично показуються, але в NED-вигляді на графіку. Рис.4.55.

```
354
355     % NED scatter
356     trueLLA = [rxlat, rxlon, rxalt];
357     nedA = lla2ned(rxposA, trueLLA, 'ellipsoid');
358     nedB = lla2ned(rxposB, trueLLA, 'ellipsoid');
359     nedC = lla2ned(rxposC, trueLLA, 'ellipsoid');
360
```

Рис.4.55. Показ координат

Після цього будується ще один сервісний графік, який порівнює амплітуду інтегрованого кореляторного виходу для першого каналу трекінгу в трьох сценаріях, щоб візуально побачити деградацію від завад і покращення від MVDR. Рис.4.56.

```
370
371     % Сервисний графік якості корелятора/інтегратора по 1-му каналу
372     figure('Name','Integrated wave (channel 1)'); clf;
373     plot(abs(rxA.accuIntegWave(:,1)), 'g'); hold on;
374     plot(abs(rxB.accuIntegWave(:,1)), 'r');
375     plot(abs(rxC.accuIntegWave(:,1)), 'b');
376     xlabel('Receiver step (ms after start of RX)');
377     ylabel('|Integrated wave|');
378     title('Tracking integrated wave magnitude (PRN #1 channel)');
379     legend('clean','jammed','jammed+MVDR');
380     grid on;
381
```

Рис.4.56. Побудова сервісного графіку

Після основної частини йдуть допоміжні функції. Спочатку initRxState створює структуру rx і ініціалізує всі поля, потрібні для acquisition/tracking/decoding, причому rx.accu... масиви мають розмір numSteps x maxCh, щоб накопичувати історію оцінок на кожному кроці. Рис.4.57.

```
%% ================== Допоміжні функції ==================

function rx = initRxState(maxCh, numSteps)
    rx.maxCh = maxCh;

    rx.performInitSync = 1;
    rx.numdetectsat = 0;

    rx.PRNIDsToSearch = [];
    rx.dopperoffsets = zeros(1,maxCh);
    rx.codephoffsets  = zeros(1,maxCh);

    rx.carrierCodeTrack = {};
    rx.framesync = {};
    rx.framesyncbuffer = {};
    rx.prev_cntr = [];
    rx.deccfg = cell(maxCh,1);

    rx.isBitSyncComplete = zeros(maxCh,1);
    rx.maxTransitionLocation = zeros(maxCh,1);
    rx.sampleCounter = zeros(maxCh,1);
    rx.syncidx = zeros(maxCh,1);

    rx.rxcntr = 1;

    rx.accuPh      = zeros(numSteps, maxCh);
    rx.accuFqy     = zeros(numSteps, maxCh);
    rx.accuFqyErr  = zeros(numSteps, maxCh);
    rx.accuPhErr   = zeros(numSteps, maxCh);
    rx.accuIntegWave = zeros(numSteps, maxCh);
    rx.accuDelay     = zeros(numSteps, maxCh);
    rx.accuDelayErr  = zeros(numSteps, maxCh);
end

function rx = receiverStep(rx, waveform, initialsync, samplingRate, ...
    PLLNoiseBW, FLLNoiseBW, DLLNoiseBW, ...
    maxNumTrackingChannels, ...
    numStepsPerSubframe, numCACodeBlocksPerBit, numWaitingStepsForBitSync)

```

Рис.4.57. Допоміжні функції

Далі receiverStep реалізує один крок обробки приймача. Спочатку, якщо rx.performInitSync == 1, виконується acquisition через initialsync, звідти беруться знайдені PRN, доплер і кодова фаза, обмежується число супутників до maxNumTrackingChannels, і для кожного супутника створюється трекер HelperGPSCACodeCarrierTracker та синхронізатор кадру HelperGPSLNAVFrameSynchronizer. Рис. 4.58.

Після цього в кожному кроці виконується трекінг для кожного супутника, і накопичуються оцінки частоти, фази, затримки та інтегрованої хвилі. Рис. 4.59.

```
421     % ---------- Початкова Синхронізація ----------
422     if rx.performInitSync == 1
423         rx.performInitSync = 0;
424
425         [y, ~] = initialsync(waveform, 1:32);
426
427         detected = y(y(:,4).IsDetected==1,1).PRNID.';
428         dopps    = y(y(:,4).IsDetected==1,2).FrequencyOffset;
429         codeph   = y(y(:,4).IsDetected==1,3).CodePhaseOffset;
430
431         rx.PRNIDsToSearch = detected;
432         rx.dopplerоffsets(1:length(dopps)) = dopps;
433         rx.codephoffsets(1:length(codeph)) = codeph;
434
435         rx.numdetectsat = length(detected);
436         if rx.numdetectsat > maxNumTrackingChannels
437             rx.numdetectsat = maxNumTrackingChannels;
438             rx.PRNIDsToSearch = rx.PRNIDsToSearch(1:maxNumTrackingChannels);
439         end
440
441         rx.framesyncbuffer = cell(1, rx.numdetectsat);
442
443         rx.carrierCodeTrack = cell(rx.numdetectsat,1);
444         rx.framesync = cell(rx.numdetectsat,1);
445         rx.prev_cntr = ones(rx.numdetectsat,1);
446
447         for isat = 1:rx.numdetectsat
448             rx.carrierCodeTrack{isat} = HelperGPSCACodeCarrierTracker;
449             rx.carrierCodeTrack{isat}.SampleRate = samplingRate;
450             rx.carrierCodeTrack{isat}.CenterFrequency = 0;
451             rx.carrierCodeTrack{isat}.PLLNoiseBandwidth = PLLNoiseBW;
452             rx.carrierCodeTrack{isat}.FLLNoiseBandwidth = FLLNoiseBW;
453             rx.carrierCodeTrack{isat}.DLLNoiseBandwidth = DLLNoiseBW;
454             rx.carrierCodeTrack{isat}.PLLIntegrationTime = 1; % ms
455             rx.carrierCodeTrack{isat}.PRNID = rx.PRNIDsToSearch(isat);
456             rx.carrierCodeTrack{isat}.InitialDopplerShift = rx.dopplerоffsets(isat);
457             rx.carrierCodeTrack{isat}.InitialCodePhaseOffset = rx.codephoffsets(isat);
458
459             rx.framesync{isat} = HelperGPSLNAVFrameSynchronizer;
460         end
461     end
462
```

Рис.4.58. Початкова синхронізація

```
463     % ---------- Трекінг ----------
464     for isat = 1:rx.numdetectsat
465         [integwave, fqyerr, fqyoffset, pherr, phoffset, derr, dnco] = ...
466             rx.carrierCodeTrack{isat}(waveform);
467
468         rx.accuFqyErr(rx.rxcntr, isat) = fqyerr;
469         rx.accuFqy(rx.rxcntr, isat)    = fqyoffset;
470         rx.accuPhErr(rx.rxcntr, isat)  = pherr;
471         rx.accuPh(rx.rxcntr, isat)     = phoffset;
472         rx.accuIntegWave(rx.rxcntr,isat) = sum(integwave);
473         rx.accuDelayErr(rx.rxcntr,isat)  = derr;
474         rx.accuDelay(rx.rxcntr,isat)     = dnco;
475     end
476
```

Рис.4.59. Трекінг

Спочатку в блоці побітної синхронізації та синхронізації кадру ми перевіряємо, що приймач уже накопичив достатньо мілісекундних кроків для пошуку меж бітів, і ця умова задана Рис.4.60.

```
477     % ---------- Bit + Frame sync + Decode ----------
478     if rx.rxcntr > numWaitingStepsForBitSync
```

Рис.4.60. Перевірка приймача

Далі запускаємо цикл по всіх супутниках, які були знайдені на етапі acquisition і занесені в rx.numdetectsat, тобто все, що далі відбувається, виконується окремо для кожного PRN-каналу. Рис. 4.61.

```
479            for isat = 1:rx.numdetectsat
```

Рис.4.61. Запуск циклу на всіх супутниках

Потім перевіряємо, чи для цього супутника вже завершена побітна синхронізація. Якщо ні, то ми шукаємо найкраще положення переходу бітів (місце зміни знаку), за перші numWaitingStepsForBitSync кроків, причому береться саме уявна частина imag(...), бо в цьому макеті біти витягуються зі знаку уявної складової. Рис.4.62.

```
480            if ~rx.isBitSyncComplete(isat)
481                rx.maxTransitionLocation(isat) = gnssBitSynchronize( ...
482                    imag(rx.accuIntegWave(1:numWaitingStepsForBitSync,isat)), ...
483                    numCACodeBlocksPerBit);
484
```

Рис.4.62. Перевірка побітної синхронізації

Після знаходження позиції переходу ми ставимо прапорець, що побітна синхронізація для цього супутника завершена, щоб наступні кроки вже не шукали перехід, а лише накопичували дані.

Далі обчислюємо лічильник rx.sampleCounter(isat), який по суті вирівнює поточний глобальний лічильник приймача rx.rxcntr відносно знайденого переходу біта, щоб буфер кадру надалі будувався з правильної точки. Рис.4.63.

```
486                rx.sampleCounter(isat) = rx.rxcntr - rx.maxTransitionLocation(isat) + 1;
```

Рис.4.63. Обчислення rx.sampleCounter(isat)

Після цього ініціалізуємо буфер rx.framesyncbuffer{isat} даними з rx.accuIntegWave, починаючи саме з rx.maxTransitionLocation(isat), тобто ми

відкидаємо все до переходу і залишаємо тільки вирівняну послідовність для майбутнього frame sync. Рис.4.64.

```
487                 rx.framesyncbuffer{isat} = rx.accuIntegWave( ...
488                     rx.maxTransitionLocation(isat):end, isat);
```

Рис.4.64. Ініціалізація rx.framesyncbuffer{isat}

Якщо ж побітна синхронізація для цього супутника вже зроблена, то ми просто нарощуємо лічильник вибірок у буфері. Рис.4.64.

```
489             else
490                 rx.sampleCounter(isat) = rx.sampleCounter(isat) + 1;
```

Рис.4.64. Нарощення лічильника вибірок

Потім записуємо в буфер нове значення хвилі за поточний мс-крок rx.rxcntr, щоб буфер framesyncbuffer постійно поповнювався

Далі перевіряємо, чи накопичили рівно один субкадр за кількістю кроків numStepsPerSubframe. Умова mod(..., numStepsPerSubframe) == 0 означає кожні 6 секунд (у перерахунку на мс-кроки) пробуємо виконати frame sync і декодування. Рис.4.65.

```
492
493             if mod(rx.sampleCounter(isat), numStepsPerSubframe) == 0
```

Рис.4.65. Перевірка накопичення субкадру

Як тільки умова спрацювала, ми вирізаємо останній блок розміром numStepsPerSubframe з буфера – це дані субкадру, підготовлені для перетворення в біти.

Потім групуємо samples у матрицю, де рядків рівно numCACodeBlocksPerBit (тобто 20 мс на 1 біт), і усереднюємо по цих 20 мс, щоб отримати символьні значення sym по бітах, це зменшує шум і стабілізує знак біта. Далі перетворюємо ці значення у біти за знаком уявної частини: якщо

imag(sym) < 0, то біт вважається 1 (або навпаки, залежно від прийнятого відображення), і так формується вектор bits. Рис.4.66.

```
496
497                     sym = mean(reshape(samples, numCACodeBlocksPerBit, []));
498                     bits = imag(sym) < 0;
499
```

Рис.4.66. Формування вектору bits

Після цього передаємо бітову послідовність у frame synchronizer для пошуку структури LNAV субкадрів. На виході отримуємо індекс синхронізації rx.syncidx(isat) та масив rxsubframes з виділеними субкадрами. Рис.4.67.

```
499
500                     [rx.syncidx(isat), rxsubframes, ~] = rx.framesync{isat}(bits(:));
501
```

Рис.4.67. Індекс синхронізації

Якщо rxsubframes не порожній, значить синхронізація кадру вдалася, і тоді ми записуємо PRN у config декодера та викликаємо HelperGPSLNAVDataDecode, який намагається витягнути параметри повідомлення. Рис.4.68.

```
502                     if ~isempty(rxsubframes)
503                         rx.deccfg{isat}.PRNID = rx.PRNIDsToSearch(isat);
504                         rx.deccfg{isat} = HelperGPSLNAVDataDecode(rxsubframes, rx.deccfg{isat});
505                         rx.prev_cntr(isat) = rx.rxcntr;
506                     end
```

Рис.4.68. Витягнення параметрів повідомлення

Після завершення всіх перевірок для даного isat ми виходимо з вкладених if, і цикл переходить до наступного супутника, а коли обробили всі супутники, то виходимо з if rx.rxcntr > numWaitingStepsForBitSync. Після цього наприкінці функції receiverStep збільшуємо лічильник кроків приймача rx.rxcntr, тобто переходимо до наступного 1-мс такту. Рис.4.69.

```
511
512         rx.rxcntr = rx.rxcntr + 1;
513     end
```

Рис.4.69. Перехід до наступного супутника по завершенню циклу

Після цього починається функція оцінки координат приймача estimatePositionFromRxState. Спочатку вона приймає стан приймача rx, істинні координати приймача rxlat, rxlon, rxalt, а також параметри, що потрібні для побудови псевдовідстаней і синхронізації. Рис.4.70.

```
515 function [distanceError, rxposest] = estimatePositionFromRxState(rx, ...
516     rxlat, rxlon, rxalt, ...
517     minStepsForPosEst, ...
518     caCodeBlockDuration, ...
519     numCACodeBlocksPerBit, ...
520     maxNumTrackingChannels, ...
521     rinexFileName)
522
```

Рис.4.70. Функція оцінки координат

Далі одразу перевіряється базова умова для позиціонування: якщо знайдено менше 4 супутників, то позицію розв’язати неможливо, тому повертається NaN і виводиться попередження. Рис.4.71.

```
523     if rx.numdetectsat < 4
524         distanceError = nan;
525         rxposest = [nan nan nan];
526         warning("Position estimation: insufficient detected satellites.");
527         return;
528     end
```

Рис.4.71. Перевірка базової умови позиціонування

Після цього задається частота чипів C/A-коду caChipRate, щоб перевести кодові зсуви у час. Далі перетворюємо початкові оцінки кодової фази, отримані на acquisition (rx.codephoffsets), у секунди, ділячи на caChipRate. Рис.4.72.

```
530         caChipRate = 1.023e6;
531
532         codeOffsetTime = rx.codephoffsets(1:rx.numdetectsat) / caChipRate;
533
```

Рис.4.72. Задання частоти чипів C/A коду

Потім беремо rx.prev_cntr – це крок приймача, на якому востаннє була успішна синхронізація/декодування для відповідного супутника. Якщо там трапляються значення ≤ 1, ми підправляємо їх на 2, щоб уникнути некоректної індексації. Рис.4.73.

```
533
534         prev_cntr = rx.prev_cntr;
535         prev_cntr(prev_cntr <= 1) = 2;
536
```

Рис.4.73. Перевірка синхронізації

Далі формуємо складову затримки, яка прийшла від трекінгу rx.accuDelay, це оцінка з DLL, переводимо її у секунди, і паралельно додаємо часові поправки від bit sync та frame sync. Час bit sync береться з rx.maxTransitionLocation у мілісекундних блоках caCodeBlockDuration, а час frame sync – з rx.syncidx у субкадрових блоках розміром numCACodeBlocksPerBit * caCodeBlockDuration. Рис.4.74.

```
536
537         trackingOffsetTime = rx.accuDelay(prev_cntr(1), 1:rx.numdetectsat) / caChipRate;
538         bitsyncTime  = (rx.maxTransitionLocation(1:rx.numdetectsat)-1) * caCodeBlockDuration;
539         framesyncTime = (rx.syncidx(1:rx.numdetectsat)-1) * numCACodeBlocksPerBit * caCodeBlockDuration;
540
```

Рис.4.74. Складова затримки

Після цього складаємо загальну оцінку затримки delayEst, комбінуючи acquisition-оцінку, трекінг-оцінку та поправки синхронізації. Рис. 4.75.

```
540
541         delayEst = codeOffsetTime(:) - trackingOffsetTime(:) + bitsyncTime + framesyncTime;
542
```

Рис.4.75. Загальна оцінка затримки

Далі відбираємо тільки ті супутники, де frame sync відбувся (rx.syncidx ~= 0), і перетворюємо затримки у псевдовідстані rho, множачи на швидкість світла. Рис.4.76.

```
542
543         validDelay = delayEst(rx.syncidx(1:rx.numdetectsat) ~= 0);
544         rho = validDelay * physconst("LightSpeed");
545
```

Рис.4.76. Відбирання потрібних супутників

Якщо після цього все одно вийшло менше 4 псевдовідстаней (наприклад, синхронізація для більшості супутників не вдалася), тоді повертаємо NaN і виходимо. Рис.4.77.

```
546         if numel(rho) < 4
547             distanceError = nan;
548             rxposest = [nan nan nan];
549             warning("Position estimation: not enough valid pseudoranges (sync failed).");
550             return;
551         end
552
```

Рис.4.77. Дії у випадку менше 4-х псевдовідстаней

Далі витягуємо TOW з декодованих повідомлень. Спочатку створюємо вектор tow розміром maxNumTrackingChannels, і для кожного можливого каналу перевіряємо, чи є поле HOWTOW у rx.deccfg{isat}. Якщо поле є, то переводимо HOWTOW у секунди, множачи на 6 (бо один субкадр 6 секунд) з поправкою +1 Рис.4.78.

```
553         % TOW from decoded message
554         tow = zeros(maxNumTrackingChannels,1);
555         for isat = 1:maxNumTrackingChannels
556             if isfield(rx.deccfg{isat},"HOWTOW")
557                 tow(isat) = (rx.deccfg{isat}.HOWTOW + 1)*6;
558             end
559         end
```

Рис.4.78. Витягування TOW з декодованих повідомлень

Після цього залишаємо в tow тільки ті супутники, для яких був успішний frame sync (rx.syncidx ~= 0), і так само відбираємо відповідні елементи конфігурації декодування у deccfg1. Рис.4.79.

```
560         tow = tow(rx.syncidx ~= 0);
561
562         deccfg1 = rx.deccfg(rx.syncidx ~= 0);
563
```

Рис.4.79. Виділення успішних супутників

Далі вводимо прапорець useDefaultSatPos, який вирішує, чи будемо рахувати координати супутників із реально декодованих ефермерид, чи впадемо в дефолтні параметри прикладу. Спочатку він false. Потім, якщо симуляція ще не дійшла до мінімального часу для якісного позиціонування minStepsForPosEst, то вмикаємо дефолт. Рис.4.80.

```
565
566         if rx.rxcntr <= minStepsForPosEst
567             useDefaultSatPos = true;
568         end
569
```

Рис.4.80. Прапорець useDefaultSatPos

Так само, якщо tow порожній або deccfg1 порожній (тобто нема що використовувати для реальної геометрії супутників), теж вмикаємо дефолт. Рис.4.81.

```
569
570         if isempty(tow) || isempty(deccfg1)
571             useDefaultSatPos = true;
572         end
573
```

Рис.4.81. Дефолт для порожніх tow або deccfg1

Якщо useDefaultSatPos == true, то завантажуємо receiverPositionProperties, де лежить дефолтна інформація, зокрема типові позиції супутників для прикладу, а також ставимо isFullDataDecoded як true для всіх псевдовідстаней. Далі задаємо еталонні ім'я RINEX і координати приймача, і якщо вони не збігаються з запуском, видаємо попередження, що оцінка може відрізнятися, бо модель не мала повних даних або декодування було неповним. Рис. 4.82.

```
574         if useDefaultSatPos
575             % The parameters that are loaded here are valid for the default configuration
576             load receiverPositionProperties;
577
578             isFullDataDecoded = true(length(rho),1);
579
580             defaultRINEXFileName = "GODS00USA_R_20211750000_01D_GN.rnx";
581             defaultRxPos = [39.021, -76.827, 19];
582             exampleRxPos = [rxlat, rxlon, rxalt];
583
584             if ~(strcmp(rinexFileName,defaultRINEXFileName) && isequal(defaultRxPos,exampleRxPos))
585                 warning("Estimated receiver position may be different as simulation didn't run full data duration or decoding failed.");
586             end
```

Рис.4.82. Алгоритм обробки вхідних даних та перевірки цілісності RINEX

Якщо ж useDefaultSatPos == false, тоді намагаємося побудувати координати супутників із декодованих ефермерид. Спочатку формуємо список обов'язкових полів reqCEIFields, які мають бути присутні у deccfg для коректного розрахунку орбіти. Рис.4.83.

```
588             % Повний розрахунок satpos із декодованими ефемеридами
589             reqCEIFields = ["WeekNumber","ReferenceTimeOfEphemeris","ReferenceTimeOfClock", ...
590                 "SemiMajorAxisLength","MeanMotionDifference","MeanAnomaly","Eccentricity", ...
591                 "ArgumentOfPerigee","Inclination","InclinationRate","HarmonicCorrectionTerms", ...
592                 "RateOfRightAscension","LongitudeOfAscendingNode"];
593
```

Рис.4.83. Побудова координат супутників із декодованих ефемерид

Далі беремо L = length(tow) як кількість супутників, для яких маємо синхронізацію, і створюємо набори масивів параметрів орбіти нульовими, використовуючи deal(zeros(L,1)), щоб одразу виділити пам'ять під усі параметри. Рис.4.84.

```
L = length(tow);

[timeofweek, SatelliteID, Delta_n, M0, Eccentricity, sqrtA, Toe, Toc, ...
    Cis, Cic, Crs, Crc, Cus, Cuc, OMEGA0, i0, omega, OMEGA_DOT, IDOT, GPSWeek] = ...
    deal(zeros(L,1));
```

Рис.4.84. Набори масивів параметрів орбіти

Після цього ставимо isFullDataDecoded = true(L,1) як припущення, що все ок, але в циклі далі будемо відмічати false там, де дані неповні.

Далі у циклі по isat = 1:L записуємо timeofweek(isat) = tow(isat), а потім перевіряємо, чи в deccfg1{isat} є PRNID і всі необхідні поля з reqCEIFields. Якщо чогось нема, то isFullDataDecoded(isat) стає false і параметри для цього супутника не заповнюються. Рис.4.85.

```
for isat = 1:L
    timeofweek(isat) = tow(isat);

    isFullDataDecoded(isat) = (isfield(deccfg1{isat},"PRNID")) && ...
        (all(isfield(deccfg1{isat}, reqCEIFields)));
```

Рис.4.85. Схема перевірки готовності даних супутників до розрахунків

Якщо дані повні, тоді переносимо параметри ефермерид у локальні масиви, причому частина величин переводиться в радіани, а sqrtA береться як корінь із SemiMajorAxisLength. Окремо зчитуються гармонічні поправки орбіти і розкладаються на Cis/Cic/Crs/Crc/Cus/Cuc. Рис.4.86.

```
    if isFullDataDecoded(isat)
        SatelliteID(isat) = deccfg1{isat}.PRNID;
        GPSWeek(isat) = deccfg1{isat}.WeekNumber;
        Toe(isat) = deccfg1{isat}.ReferenceTimeOfEphemeris;
        Toc(isat) = deccfg1{isat}.ReferenceTimeOfClock;
        sqrtA(isat) = sqrt(deccfg1{isat}.SemiMajorAxisLength);
        Delta_n(isat) = deccfg1{isat}.MeanMotionDifference*pi;
        M0(isat) = deccfg1{isat}.MeanAnomaly*pi;
        Eccentricity(isat) = deccfg1{isat}.Eccentricity;
        omega(isat) = deccfg1{isat}.ArgumentOfPerigee*pi;
        i0(isat) = deccfg1{isat}.Inclination*pi;
        IDOT(isat) = deccfg1{isat}.InclinationRate*pi;

        hterms = num2cell(deccfg1{isat}.HarmonicCorrectionTerms);
        [Cis(isat), Cic(isat), Crs(isat), Crc(isat), Cus(isat), Cuc(isat)] = deal(hterms{:});

        OMEGA_DOT(isat) = deccfg1{isat}.RateOfRightAscension*pi;
        OMEGA0(isat) = deccfg1{isat}.LongitudeOfAscendingNode*pi;
    end
end
```

Рис.4.86. Перенесення параметрів ефемерид у локальні масиви

Після циклу перевіряємо, чи хоч для одного супутника дані повні. Якщо ні, то виводимо попередження і знову падаємо у receiverPositionProperties, виставивши isFullDataDecoded для всіх псевдовідстаней як true. Рис.4.87.

```
628
629             if ~any(isFullDataDecoded)
630                 warning("Ephemeris incomplete for this scenario; falling back to default satpos.");
631                 load receiverPositionProperties;
632                 isFullDataDecoded = true(length(rho),1);
```

Рис.4.87. Перевірка даних супутника

Якщо ж є хоча б один супутник з повними ефермерами, тоді формуємо час передачі transmissionTime через HelperGPSConvertTime для супутників з повними даними, а також refTime як часову прив'язку, потім збираємо timetable з усіх параметрів і викликаємо gnssconstellation щоб отримати satpos – координати супутників у момент передачі. Рис.4.88.

```
633             else
634                 transmissionTime = HelperGPSConvertTime(GPSWeek(isFullDataDecoded), timeofweek(isFullDataDecoded));
635                 refTime = HelperGPSConvertTime(GPSWeek(:), Toc(:));
636
637                 rxtimetable = timetable(refTime, SatelliteID, Delta_n, M0, Eccentricity, sqrtA, ...
638                     Toe, Cis, Cic, Crs, Crc, Cus, Cuc, OMEGA0, i0, omega, ...
639                     OMEGA_DOT, IDOT, GPSWeek);
640
641                 if isempty(transmissionTime)
642                     warning("Empty transmissionTime; falling back to default satpos.");
643                     load receiverPositionProperties;
644                     isFullDataDecoded = true(length(rho),1);
645                 else
646                     [satpos, ~] = gnssconstellation(transmissionTime(1), rxtimetable);
647                 end
648             end
649         end
```

Рис.4.88. Координати супутника у момент передачі

Після цього є додатковий захист: якщо змінна satpos так і не була створена, тоді завантажуємо дефолтні параметри. Рис.4.89.

```
650
651         if exist("satpos","var") ~= 1
652             load receiverPositionProperties;
653         end
```

Рис.4.89. Завантаження дефолтних параметрів

Далі узгоджуємо розмірності: псевдовідстаней може бути Npr, а супутників у satpos може бути Ns, тому беремо N = min(Npr, Ns) і обрізаємо масиви, щоб вони відповідали один одному. Рис.4.90.

```
654
655         Npr = length(rho);
656         Ns  = size(satpos,1);
657         N   = min(Npr, Ns);
658
659         rho = rho(1:N);
660
```

Рис.4.90. Узгодження розмірностей

Після цього, якщо isFullDataDecoded раптом має іншу довжину, ми підміняємо його на true(N,1), і так само обрізаємо satpos. Рис.4.91.

```
661         if length(isFullDataDecoded) ~= N
662             isFullDataDecoded = true(N,1);
663         end
664         satpos = satpos(1:N,:);
665
```

Рис.4.91. Процедура синхронізації параметрів за кількістю супутників

Тепер безпосередньо обчислюємо оцінену позицію приймача rxposest через receiverposition(...), використовуючи тільки ті супутники, для яких isFullDataDecoded true. Рис.4.92.

```
666         % Обчислення позиції
667         rxposest = receiverposition(rho(isFullDataDecoded), satpos(isFullDataDecoded,:));
668
```

Рис.4.92. Обчислення позиції

Далі перетворюємо оцінену позицію у локальні NED-відхилення відносно істинної [rxlat, rxlon, rxalt] і беремо норму як метрику помилки. Рис.4.93.

```
669         estRxPosNED = lla2ned(rxposest, [rxlat, rxlon, rxalt], 'ellipsoid');
670         distanceError = vecnorm(estRxPosNED);
671     end
```

Рис.4.93. Розрахунок метрики похибки у системі NED

Далі допоміжна функція generateNarrowbandBlock створює вузькосмугову модульовану заваду. Спочатку вона визначає символьну швидкість symRate як приблизно bw/2, але не менше 1, щоб уникнути нульових значень. Рис. 4.94.

```
673    function nb = generateNarrowbandBlock(Ns, fs, bw, amp)
674        symRate = max(1, round(bw/2));
```

Рис.4.94. Створення вузькосмугової модульованої завади

Потім обчислюємо, скільки семплів припадає на символ, і скільки символів потрібно, щоб покрити Ns семплів блоку. Рис.4.95.

```
675        samplesPerSym = max(1, round(fs / symRate));
676        numSym = ceil(Ns / samplesPerSym);
677
```

Рис.4.95. Обчислення семплів

Далі генеруємо випадкові символи з 4-фазної маніпуляції (QPSK-подібно), використовуючи randi([0 3]) і множення на $\pi/2$, після чого бере комплексну експоненту. Рис. 4.96.

```
677
678        sym = exp(1j * (randi([0 3], numSym, 1) * pi/2));
679
```

Рис.4.96. Генерація випадкових символів з 4-фазної маніпуляції

Після цього створюємо апсемплений вектор up довжини Ns, де символи записуються з кроком samplesPerSym, а між ними нулі. Рис.4.97.

```
680        up = zeros(Ns,1);
681        idx = 1:samplesPerSym:Ns;
682        up(idx(1:length(sym))) = sym;
683
```

Рис.4.96. Створення апсемпленого вектора

Далі застосовується простий усереднювальний фільтр довжини samplesPerSym, який згладжує імпульсні вставки символів і робить сигнал вузькосмуговішим. Рис.4.97.

```
683
684         b = ones(samplesPerSym,1)/samplesPerSym;
685         nb = filter(b, 1, up);
686
```

Рис.4.97. Запис усередненого фільтру довжини

Наприкінці масштабуються амплітуда завади amp, і результат повертається. Далі steeringVectorFromAzEl будує керуючий вектор (steering vector) для плоскої хвилі, яка приходить з азимута az та кута місця el. Спочатку переводимо напрям у координатах ENU у вектор u, де використовуються cos(el) та sin(el) для горизонтальної/вертикальної складової, а азимут розкладається на компоненти через sin(az) і cos(az):

Потім обчислюємо хвильове число, яке визначає, як фаза змінюється в просторі на частоті fc. Рис.4.98.

```
690     function sv = steeringVectorFromAzEl(az, el, arrayCoords, fc, c)
691         % ULA у локальній системі масиву:
692         % модель плоскої хвилі
693         u = [cos(el)*sin(az), cos(el)*cos(az), sin(el)]; % ENU-напрямок
694         k = 2*pi*fc / c;
695
```

Рис.4.98. Розрахунок керуючого вектора для плоскої хвилі

Далі рахуємо фазовий набіг на кожному елементі масиву як скалярний добуток координат елементів arrayCoords на напрям u, помножений на -k, і формуємо комплексні експоненти як steering-вектор. Рис.4.99.

```
696         % Для простоти трактуємо arrayCoords як локальні
697         phase = -k * (arrayCoords * u.');
698         sv = exp(1j*phase).';  % 1 x M
699     end
```

Рис.4.99. Обчислення фазового профілю керуючого вектору

Остання функція mvdrWeights рахує ваги MVDR. Спочатку перетворюємо вхідний steering-вектор у стовпчик sv, беремо розмір решітки M з матриці коваріації R. Рис.4.100.

```
function w = mvdrWeights(R, sv_row)
    % sv_row: 1 x M
    sv = sv_row(:);
    M = size(R,1);

```

Рис.4.100. Обрахунок ваги MVDR

Потім додаємо діагональне навантаження пропорційно середньому значенню trace(R)/M, щоб зробити інверсію стабільнішою, і додаємо це до діагоналі. Рис.4.101.

```
    % Diagonal loading
    dl = 1e-3 * trace(R)/M;
    R = R + dl * eye(M);

```

Рис.4.101. Діагональне навантаження

Далі нормалізуємо sv, щоб масштаб steering-вектора не впливав на чисельну стійкість. Рис.4.102.

```
    % Normalize sv
    if norm(sv) > 0
        sv = sv / norm(sv);
    end

```

Рис.4.102. Нормалізація sv

Після цього обчислюємо псевдообернену Rinv = pinv(R) і знаменник MVDR. Рис.4.103.

```
    Rinv = pinv(R);
    denom = (sv' * Rinv * sv);
```

Рис.4.103. Обчислення псевдооберненої матриці та параметрів MVDR

Якщо знаменник майже нульовий, то для безпеки беремо простий рівномірний розподіл ваг, застосовуємо класичну формулу MVDR. Рис.4.104.

```
714
715         Rinv = pinv(R);
716         denom = (sv' * Rinv * sv);
717         if abs(denom) < eps
718             w = sv / M;
719         else
720             w = (Rinv * sv) / denom;
721         end
722     end
723
724
```

Рис.4.104. Формування вагового вектору з механізмом стабілізації

Отримані результати

У результаті моделювання для супутника PRN1 побудовано графік модуля інтегрованого виходу трекінгу для трьох режимів: clean (зелений), jammed (червоний) і jammed+MVDR (синій). До приблизно 18с, усі криві мають близькі значення, бо завади ще не вмикаються.

Після 18с у режимі jammed видно різкий перехідний стрибок і далі стабільно вищий середній рівень та більші коливання, що означає сильний вплив сумарної завади (гармонічна + вузькосмугова модульована + гаусівський шум). У режимі jammed+MVDR такого різкого зростання немає: рівень помітно нижчий і ближчий до clean, тобто MVDR на антенної решітці пригнічує заваду перед обробкою приймачем. Рис.4.105.

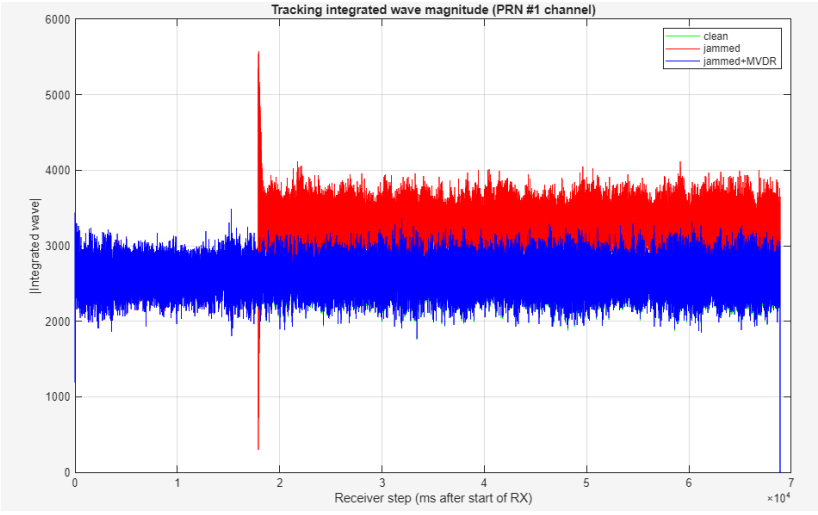

Рис.4.105. Оцінка впливу MVDR-фільтрації на рівень вихідного сигналу трекінгу в умовах завад

На Рис.4.106. показано порівняння помилки визначення координат для трьох сценаріїв (clean, jammed, jammed+MVDR). По осі Y – Position error (m), тобто норма вектора помилки в локальній системі NED, яку рахує функція estimatePositionFromRxState як distanceError = vecnorm(lla2ned(rxposest, [rxlat, rxlon, rxalt], ...)). Для clean отримано приблизно 25.01 м, а для jammed+MVDR – приблизно 24.83 м, тобто після MVDR помилка трохи менша (приблизно на 0.18 м).

Стовпчика для jammed фактично не видно, що означає, що позиціонування в jammed-сценарії зірвалось через провал синхронізації/декодування, тоді як застосування антенної решітки + MVDR дозволило зберегти працездатний розрахунок координат.

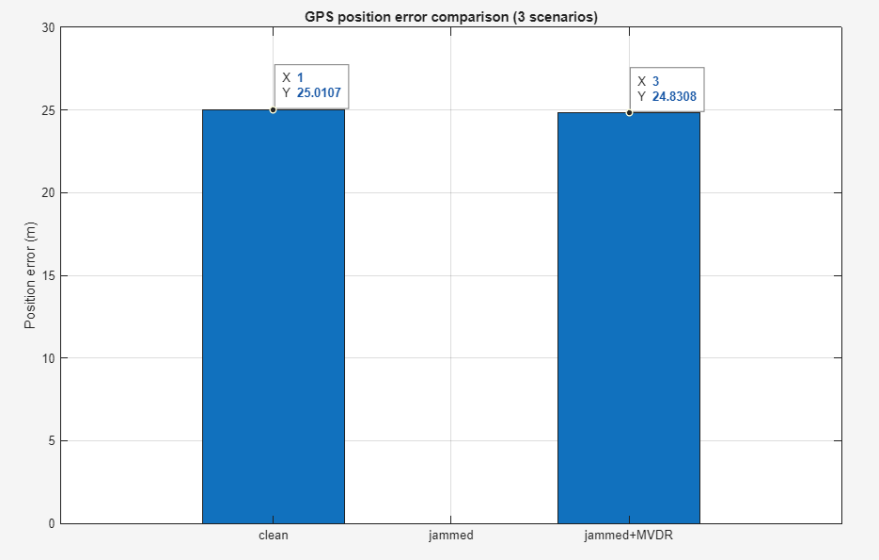

Рис.4.106. Порівняння середньоквадратичної похибки позиціонування у сценаріях з активними завадами

Повний end-to-end ланцюжок симуляції виконувався 3223.41 с (≈ 53.7 хв). Після цього були обчислені похибки позиціонування: для clean помилка 25.0107 м, для jammed позицію не вдалося оцінити через зрив синхронізації/недостатньо валідних псевдовідстаней, а для jammed + array MVDR помилка 24.8308 м. Рис.4.107.

```
End-to-End chain ran for 3223.4086 seconds.

Distance error (NED norm):
A - clean:                25.0107 m
B - jammed:               NaN m
C - jammed + array MVDR:  24.8308 m
>>
```

Рис.4.107. Вивід консолі

Висновок

У цьому розділі було розглянуто та описано імітаційні GPS моделі у середовищі MATLAB. Зокрема було описано основи GPS та генерацію сигналів GPS у середовищі MATLAB. Потім було описано захоплення та відстеження GPS-приймача та наскрізний GPS-навігаційний приймач старого покоління з використанням C/A-коду. Після цього було розглянуто захоплення та відстеження GPS-приймача за допомогою C/A-коду. Потім наступним підрозділом була робота в середовищі MATLAB. Було проведено MATLAB-моделювання GPS-приймача з адаптивною антенною решіткою. Було зроблено наступне: створено модель GPS-приймача в середовищі MATLAB, яка приймає задані вхідні дані та формує радіосигнали на вхід приймача. Приймач обробляє сигнал і визначає координати. Далі до вхідного сигналу було додано завади (гармонічні, модульовані вузькосмугові та гаусівський шум), що призвело до порушення роботи приймача. Після цього реалізовано багатоканальну адаптивну антену з алгоритмом адаптації, завдяки чому робота приймача відновилася. Було продемонстровану успішну реалізацію задуманого.

РОЗДІЛ 5
РОЗРОБЛЕННЯ СТАРТАП-ПРОЄКТУ

5.1. Опис ідеї проєкту

Розроблено імітаційну модель роботи GPS приймача з подальшим впровадженням в канал різного типу перешкод та реалізацією приймача з адаптивною антенною решіткою для протидії перешкодам, в середовищі MATLAB.

Таблиця 5.1.

Опис ідеї стартап-проєкту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Створення адаптивного GNSS-приймача з антенною решіткою, здатного автоматично пригнічувати радіоперешкоди та забезпечувати високу точність позиціонування в умовах завад.	1. Автомобільна навігація в умовах міських перешкод і джамерів	- Стійке позиціонування навіть при дії джамерів, спуферів та інтерференції
	2. Дрони та БПЛА для стабільного позиціонування в складних умовах	- Підвищена точність координат завдяки адаптивним алгоритмам придушення завад
	3. Військова та оборонна техніка для навігації у високозавадних середовищах	- Зменшення ризику втрати сигналу та аварій критичних систем
	4. Маршрутизація та моніторинг транспорту: логістика, доставка.	- Гнучка інтеграція з існуючими GNSS-системами.
	5. Геодезія та точні вимірювання	- Можливість тестування різних сценаріїв у MATLAB перед впровадженням
	6. Сільське господарство, точне землеробство.	

У наступній таблиці наведено порівняння з альтернативними рішеннями, а також надано оцінку сильних і слабких сторін, досягнутих завдяки методології

проєктування, яка гарантує високу швидкодію та оптимальну продуктивність у кожній конкретній ситуації.

Таблиця 5.2.

Визначення сильних, слабких та нейтральних характеристик ідеї проєкту

№ п/п	Техніко-економічні характеристики ідеї	(потенційні) товари/концепції конкурентів				W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
		Мій проєкт	NovAtel GAJT	Raytheon Landshie	BAE DIGAR			
1.	Наявність готового комерційного продукту	Ні	Так	Так	Так	Так		
2.	Рівень перешкодостійкості	Помір-ний	Високий	Високий	Високий		Так	
3.	Габарити, маса, споживання	Варіати-вний	Середні й	Середні й	Великий		Так	
4.	Доступність для тестування, відкритість алгоритмів	Висока	Низька	Низька	Низька			Так
5.	Ціна та доступність	Низька	Висока	Дуже виска	Дуже висока			Так
6.	Масштабованість кількості антенних елементів	Повна	Середня	Середня	Висока			Так
7.	Швидкість впровадження інновацій	Дуже висока	Низька	Середня	Середня			Так
8.	Рівень технологічної зрілості (TRL)	4-5	8-9	9	9	Так		

5.2. Технологічний аудит ідеї проекту

Таблиця 5.3.

Технологічна здійсненність ідеї проєкту

№ п/п	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
1.	Проєктування адаптивної антени для GPS-приймача з моделюванням перешкод	Моделювання GNSS-сигналів та адаптивних антен у MATLAB з використанням Phased Array System Toolbox та Signal Processing Toolbox	Так, технологія існує як комерційне ПЗ з готовими інструментами для симуляції сигналів та алгоритмів	Так, доступна авторам через університетські ліцензії або пробні версії MATLAB
2.		Моделювання в Python з бібліотеками NumPy, SciPy та scikit-signal для імітації перешкод та адаптивних фільтрів	Так, технологія існує як open-source інструменти, але вимагає розробки конкретних скриптів для GNSS-моделі	Так, доступна авторам безкоштовно через відкритий код та онлайн-ресурси
3.		Апаратна реалізація на базі SDR-платформ, наприклад, USRP з GNU Radio для тестування адаптивної антени	Так, технологія існує, але потребує доопрацювання для конкретних GNSS-сигналів та перешкод	Обмежено доступна, якщо автори мають обладнання або лабораторії; інакше вимагає інвестицій

Обрана технологія реалізації ідеї проєкту: Моделювання в MATLAB, оскільки вона наявна, доступна авторам, не вимагає значних доопрацювань базових інструментів та дозволяє ефективно реалізувати імітаційну модель GPS-приймача з адаптивною антеною для протидії перешкодам. Проект технологічно здійсненний з фокусом на програмну симуляцію.

5.3. Аналіз ринкових можливостей запуску стартап-проекту

Таблиця 5.4.

Попередня характеристика потенційного ринку стартап-проєкту

№ п/п	Показники стану ринку (найменування)	Характеристика
1.	Кількість головних гравців, од	Близько 50-100 глобальних компаній, в Україні – від 5-10
2.	Загальний обсяг продаж, грн/ум.од	Глобальний ринок GNSS – близько 12 300 млрд грн; Ринок анти-джемінгу – близько 225 млрд грн; В Україні – оціночно 1-2 млрд грн з фокусом на військові потреби.
3.	Динаміка ринку (якісна оцінка)	Зростаюча через військові конфлікти, автономні транспортні засоби та точне землеробство.
4.	Наявність обмежень для входу (вказати характер обмежень)	Високі бар'єри: технологічні – патенти, експертиза в сигналовій обробці, капітальні – розробка та тестування, регуляторні – сертифікація та ринкові – конкуренція від великих гравців.
5.	Специфічні вимоги до стандартизації та сертифікації	Вимагає відповідності стандартам ICAO, FAA (AC 20-138), EASA (ETSO-C129A), військовим сертифікаціям (MIL-STD), тестування на чутливість та стійкість до перешкод.
6.	Середня норма рентабельності в галузі (або по ринку), %	5-10%

За результатами аналізу, ринок є привабливим для входження: попит існує та зростає (обсяг GNSS ринку >300 млрд USD з CAGR 11%), динаміка позитивна через геополітичні фактори та технологічний прогрес. Середня рентабельність (5-10%) близька до банківських депозитних ставок в Україні (близько 10% у 2025 р.), але потенціал для вищої прибутковості в ніші анти-джемінгу робить проект перспективним, особливо з фокусом на військові та цивільні застосування. Рекомендується інвестувати з урахуванням подолання бар'єрів через партнерства та гранти.

Таблиця 5.5.

Характеристика потенційних клієнтів стартап-проєкту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1.	Стійкість до електронної боротьби та перешкод у критичних умовах, що забезпечує безперервну навігацію в зонах конфліктів та військових операціях	Військовий сектор: Збройні Сили України, оборонні підприємства, виробники БПЛА та розвідувального обладнання	Висока лояльність до постачальників з сертифікацією, пріоритет безпеки та конфіденційності, низька чутливість до ціни за рахунок бюджетного фінансування, довгострокові контракти з сервісом	Висока стійкість до перешкод, компактність для мобільних платформ, сертифікація MIL-STD-461, інтеграція з існуючими GNSS-системами, ціна до 50 000 грн/одиниця з гарантією 5 років.
2.	Точна навігація в умовах урбанізованого середовища та перешкод для підвищення продуктивності, з фокусом на економію ресурсів.	Цивільний сектор: сільськогосподарські підприємства, виробники автономних транспортних засобів, геодезичні фірми	Орієнтація на швидке впровадження, чутливість до ціни та простоти інтеграції, активне використання дилерських мереж, фокус на демонстраціях, висока чутливість до оновлень ПЗ	Сумісність з комерційними стандартами GPS/GLONASS/ Galileo, низьке енергоспоживання, точність позиціонування, модульна конструкція для легкої інтеграції, ціна 10 000-20 000 грн.

Таблиця 5.6.

Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1.	Висока конкуренція на ринку GNSS-обладнання	Існують сильні гравці (u-blox, NovAtel, Trimble), що пропонують стійкі до завад приймачі.	Фокус на нішах: БПЛА, військові/спецпризначення, високоточна навігація.
2.	Недовіра ринку до нових технологій	Клієнти можуть обирати класичні антени через простоту, невисоку ціну та перевірену надійність.	Демонстрація переваг через тестові стенди; відкриті польові випробування; надання порівняльних звітів продуктивності.

Продовження таблиці 5.6.

3.	Залежність від постачальників електронних компонентів	Перебої в поставках мікросхем або зростання їх вартості ускладнюють виробництво приймачів із ААР.	Диверсифікація постачальників; використання взаємозамінних компонентів; локалізація критичних виробництв.

Таблиця 5.7.

Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1.	Поширення БПЛА та автономних систем	Дрони, автопілоти, роботизовані системи вимагають високої точності позиціонування та захисту від перешкод.	Створення компактної, енергоефективної ААР; оптимізація MATLAB-алгоритмів для роботи на вбудованих платформах.
2.	Зростання попиту на стійку до завад навігацію	Критична інфраструктура, БПЛА, військові та промислові системи потребують захищених приймачів.	Розробка й комерціалізація ААР-модуля як окремого продукту; вихід у сегмент оборонних та професійних рішень.
3.	Державні та оборонні програми з розвитку навігаційних технології	Країни інвестують у безпеку навігації, зокрема Україна у випадку екзистенційної загрози існуванню.	Участь у державних проєктах; подання на гранти; партнерство з інститутами та оборонними підприємствами.

Таблиця 5.8.

Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
4. Тип конкуренції: олігополія з елементами монополістичної конкуренції	Ринок ГНСС-приймачів контролюється кількома великими виробниками (u-blox, Trimble, NovAtel). Водночас дрібні інноваційні компанії пропонують унікальні функціональні рішення.	Позиціонування продукту через інноваційність алгоритмів MATLAB та ААР; патентування власних рішень; фокус на специфічних сегментах.
2. За рівнем конкурентної боротьби: міжнародний (глобальний)	Продукція конкурує на світовому ринку, де активно працюють США, Канада, ЄС, Китай. Усі постачальники змагаються за якість, точність, стійкість до завад.	Розробка продукту, сумісного з мультисистемними GNSS; відповідність міжнародним стандартам.

Продовження таблиці 5.8.

5. За галузевою ознакою: внутрішньогалузева конкуренція	Конкуренти працюють у межах однієї галузі – виробництво ГНСС-приймачів та антенних систем. Основна боротьба ведеться між компаніями, які створюють технології підвищення стійкості до завад.	Впровадження сучасних алгоритмів адаптивної фільтрації, створення ААР як додаткового модуля до існуючих приймачів; співпраця з виробниками антен.
4. Конкуренція за видами товарів: товарно-видова	Конкуренція відбувається між різними типами приймачів одного призначення: класичні антени, антени із захистом, адаптивні антени, SDR-рішення.	Створення компактного, енергоефективного ААР-модуля; демонстрація переваг над класичними антенами через симуляції MATLAB; гнучкість інтеграції у різні платформи.
5. За характером конкурентних переваг: нецінова конкуренція	Основні критерії конкуренції – точність, стійкість до завад, якість фільтрації, швидкість захоплення сигналу, підтримка багаточастотності. Ціна є вторинною.	Фокус на технічних характеристиках; створення унікальних алгоритмів просторової обробки; проведення тестів у симуляційному середовищі; підвищення надійності.
6. За інтенсивністю: марочна конкуренція (брендова)	На ринку присутні відомі бренди з сильною репутацією, довгим циклом довіри та наявністю сертифікацій.	Формування власного бренду через якісну документацію, демонстрацію результатів MATLAB-моделювання, участь у виставках та конференціях; створення демонстраційних прототипів.

Таблиця 5.9.

Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
	NovAtel GAJT, Raytheon Landshield, BAE DIGAR	Основні бар’єри входження: потреба у сертифікації; тривала побудова довіри бренду у галузі навігації.	Фактори сили постачальників: високі ціни на високотехнологічні елементи; можливі політичні та логістичні ризики.	Фактори сили споживачів: високі вимоги до якості й точності навігації; низька толерантність до помилок в оборонному сегменті;	Пасивні антени, приймачі з покращеними DSP-функціями без ААР.

Продовження таблиці 5.9.

Висно-вки	Інтенсив-ність боротьби: дуже висока – ринок високо-техноло-гічний і насичений сильними гравцями.	Можливість входу: можлива лише через інновацію. Потенційні конкуренти: малі стартапи SDR-напрямку або оборонні підприємства. Строки появи нових гравців: 2–5 років, з огляду на швидкий розвиток.	Постачальники диктують умови, бо ринок радіочастотних компонентів концентрований у США/ЄС/Азії. А саме: мінімальні партії закупівлі, висока ціна.	Споживачі диктують умови: сертифікаційні вимоги, стабільність, дослідження стійкості, тривалий супровід продукту.	Дешевші альтерна-тиви можуть витісняти ААР у масовому сегменті;

Отже, аналіз конкурентної ситуації показує, що можливість виходу на ринок є високою за умови зайняття технологічної ніші. Цей висновок ґрунтується на необхідності пропонування унікальних характеристик, зокрема, стійкості до радіоперешкод, які забезпечуються розробленою адаптивною антеною. Саме ця технологічна перевага дозволяє подолати високі бар'єри входу та вигідно диференціювати продукт на тлі конкурентів. Здатність адаптивної антени підтримувати стабільний зв'язок формує чітку цінність пропозиції для потенційних клієнтів у секторах безпеки, оборони та промислової автоматизації.

Таблиця 5.10.

Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
1	Динаміка галузі	Ринок постійно потребує вдосконалення інтерфейсу та простоти інтеграції новітніх технологій у існуючі приймачі.
2	Концепція товару	Система, що включає адаптивну антену, дозволяє утримувати споживачів, пропонуючи їм гарантовану працездатність в умовах високого рівня перешкод, що є потужним нецiновим маркетинговим фактором.
3	Обслуговування після продажу	Це включає надання інструкцій з налаштування алгоритмів адаптивної обробки, оновлення прошивки та консультації щодо інтеграції приймача в комплексні

		навігаційні системи.

Таблиця.5.11.

Порівняльний аналіз сильних та слабких сторін імітаційної моделі роботи GPS приймача з адаптивною антенною решіткою для протидії перешкодам, в середовищі MATLAB

№ п/п	Фактори конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні з власною системою						
			-3	-3	-1	0	1	2	3
1.	Динаміка в галузі	20							+
2.	Концепція товару	18						+	
3.	Обслуговування після продажу	12					+		

Таблиця 5.12.

SWOT- аналіз стартап-проєкту

Сильні сторони: інноваційний підхід, практична цінність, використання потужного інструментарію.	Слабкі сторони: обмеженість симуляційним середовищем, високі вимоги до стандартизації
Можливості: подальший розвиток, застосування в інших сферах	Загрози: швидкий розвиток технологій, обмеження апаратного забезпечення

Таблиця 5.13.

Альтернативи ринкового впровадження стартап-проєкту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1.	Стратегія вузької спеціалізації	Середня – потрібні високовартісні радіочастотні компоненти, але можна почати з прототипу на SDR; можливість грантів/держзамовлень для оборонних проєктів.	1-2 роки, час залежить від можливості швидко побудувати демо-зразок.
2.	Стратегія партнерства з виробником приймачів або оборонним підприємством	Висока – партнери можуть взяти на себе витрати на сертифікацію, закупівлю компонентів та виробництво	2-3 роки – час на укладення угод, інтеграцію в апаратну платформу, перші польові тести
3.	Стратегія технологічного прориву через створення	Низька – Середня – потребує оптимізації витрат, труднощі отримання	3-5 років –повільніший вихід через необхідність серійного виробництва,

	малобюджетної ААР для БПЛА та комерційного сегмента	дешевих компонентів, конкуренція з китайськими недорогими рішеннями.	налаштування ланцюжків постачання

Оптимальною стратегією для виведення стартапу на ринок є стратегія вузької спеціалізації, оскільки вона потребує найменших ресурсів і може бути реалізована у найкоротші строки завдяки можливості швидкого створення прототипу на основі вже розробленої MATLAB-моделі. Саме ця альтернатива забезпечує найвищу ймовірність отримання ресурсів та найшвидше входження у конкурентний високотехнологічний ринок. Крім того, фокусування на вузькій ніші дозволяє сконцентрувати обмежені фінансові та людські ресурси стартапу на вирішенні конкретної проблеми цільового сегменту, що підвищує шанси на створення дійсно затребуваного продукту та формування лояльної клієнтської бази на ранніх етапах розвитку. Такий підхід також мінімізує ризики розпорошення зусиль та створює можливість для поступового масштабування.

5.4. Розроблення ринкової стратегії проєкту

Таблиця 5.14.

Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1.	Оборонні підприємства, інтегратори систем навігації.	Висока. Через потребу у завадо-захищених антенних системах.	Середній. Сегмент стабільний і високовартісний.	Дуже висока Домінують NovAtel, Raytheon, BAE Systems.	Низька. Високі вимоги безпеки, довгий цикл перевірок.
2.	Наукові лабораторії, університети, що займаються GNSS, SDR та завадостійкими технологіями.	Висока. Високий інтерес до нових технологій та відкритих архітектур.	Низький. Попит обмежений дослідницькими проєктами.	Низька. Невелика кількість прямих конкурентів.	Висока. Найпростіший сегмент для входу, достатньо прототипу та документації.
3.	Виробники БПЛА, безпілотних платформ та	Середня. Залежить від вимог до стійкості	Високий. Зростаюча кількість дронів потребує	Висока. Активна конкуренція між	Середня. Можливий доступ через демонстрацію

	роботизованих систем.	навігації та бюджету платформ.	захисту від перешкод.	стартапами та великими виробниками.	прототипу та польових тестів.

На основі проведеного аналізу сегментів найбільш доцільною цільовою групою для виведення продукту є виробники БПЛА, безпілотних платформ та роботизованих систем. Цей сегмент має високий потенційний попит, проявляє реальну зацікавленість у завадостійких GNSS-системах і характеризується відносно простішим входом порівняно з оборонними гігантами чи критичною інфраструктурою. Відповідно, компанія обирає стратегію концентрованого маркетингу, фокусуючись саме на цьому сегменті як найбільш перспективному та доступному для стартапу.

Таблиця 5.15.

Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проєкту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
1.	Орієнтована на використання адаптивної антенної решітки як інноваційного модуля, що захищає від завад.	Концентрований маркетинг (орієнтація на один найбільш перспективний сегмент)	Висока завадостійкість завдяки адаптивній антенній решітці та MATLAB-алгоритмам; можливість швидкого створення робочого прототипу;	Стратегія спеціалізації, з елементами диференціації – унікальність технології та функціоналу.

Таблиця 5.16.

Визначення базової стратегії конкурентної поведінки

№ п/п	Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
1.	Ні, ринок вже сформований та зайнятий лідерами (NovAtel GAJT, Raytheon Landshield, BAE DIGAR)	Компанія переважно забиратиме частину існуючих клієнтів лідерів за рахунок нижчої ціни та швидшої адаптації під	Так, буде частково наслідувати критично важливі характеристики лідерів: захист від спрямованих перешкод, адаптивну обробку, але зосередиться на їх	Стратегія наслідування лідеру

		нішеві потреби.	удосконаленні в межах вузького сегмента.	

Таблиця 5.17.

Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкуренто-спроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
1.	Висока стійкість до перешкод і електронної боротьби, компактність, сертифікація, сумісність із системами GNSS, точність у складних умовах, можливість інтеграції у мобільні та безпілотні платформи	Стратегія спеціалізації з елементами диференціації – фокус на унікальному модулі адаптивної антенної решітки	Завадостійкість завдяки адаптивній антенній решітці та MATLAB-алгоритмам; швидке створення прототипу; нижча ціна порівняно з лідерами ринку; можливість кастомізації під нішеві потреби	- Захищена навігація; - Адаптивний інтелект сигналу; - Технологія військового рівня для компактних систем.

5.5. Розроблення маркетингової програми стартап-проекту

Таблиця 5.18.

Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити
1.	Стійкість до електронної боротьби та перешкод у критичних умовах.	Безперервна та надійна навігація у зонах конфліктів та військових операцій	Використання адаптивної антенної решітки; MATLAB-алгоритми обробки сигналу; швидке створення робочого прототипу; можливість кастомізації під конкретні потреби
2.	Точна навігація в умовах урбанізованого середовища та перешкод	Підвищення продуктивності та точності позиціонування, економія ресурсів	Модульна конструкція для легкої інтеграції; сумісність із GPS/GLONASS/Galileo
3.	Потреба у швидкому впровадженні та інтеграції продукту	Швидке впровадження в існуючі системи та економія часу на інтеграцію	Гнучкість під конкретні нішеві завдання; можливість програмного налаштування алгоритмів у MATLAB;

Таблиця 5.19.

Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові		
I. Товар за задумом	Адаптивна антена здатна забезпечити цілісність та безперервність високоточної GNSS-навігації. Основна функціональна вигода: Пригнічення перешкод		
II. Товар у реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх /Тл/Е/Ор
	1. Протидія перешкодам: Пригнічення 3-х одночасних джерел перешкод	М	Тх, Нд
	2. Фізичні параметри: 6-ти канальна APP	М	Тх, Тр, Е
	Якість: Якість функціонування в симуляції, що підтверджується мінімізацією шуму позиціонування та швидкості. Стійкість корпусу до зовнішніх умов експлуатації.		
	Пакування: Компактний, захищений корпус. Інструкції щодо обов'язкового монтажу на заземлювальну пластину		
	Марка: назва організації-розробника + назва товару: (Назва ВНЗ) + Програмно-апаратний комплекс «Адаптивна Антена APP-GNSS v1.0».		
III. Товар із підкріпленням	До продажу: Консультації, навчання персоналу, детальна технічна документація та верифікація моделі MATLAB		
	Після продажу: Технічна підтримка, гарантія (на алгоритм), оновлення ПЗ для протидії новим загрозам. Щорічна плата за обслуговування ліцензії MATLAB		
За рахунок чого потенційний товар буде захищено від копіювання: за рахунок захисту алгоритмічного ядра. Патентування оригінального методу формування діаграми спрямованості та захист ПЗ: Ліцензування та розповсюдження скомпільованого, закритого MATLAB-коду.			

Таблиця 5.20.

Визначення меж встановлення ціни

№ п/п	Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
1.	Дешевші рішення без адаптивної антенної решітки, 5 000 – 15 000 грн	Товари конкурентів з частковим захистом від перешкод, 40 000 – 60 000 грн	Військовий та оборонний сектор: високий бюджет, готові до інвестицій у безпеку; Цивільний сектор: середній дохід, чутливість до ціни.	Нижня межа: 10 000 грн. Верхня межа: 50 000 грн для військового/оборонного сектору з гарантією 5 років і сертифікацією.

Таблиця 5.21.

Формування системи збуту

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1.	Військовий сектор: довгострокові контракти, високі вимоги до сертифікації, пріоритет безпеки та конфіденційності.	Консультації з інтеграції у військові системи, забезпечення сертифікації, технічна підтримка, навчання персоналу, гарантійне обслуговування, супровід оновлень ПЗ.	Мала кількість проміжних ланок, прямий доступ до ключових клієнтів.	Прямий продаж компаніям/підрозділам ЗСУ та оборонним підприємствам, персоналізоване постачання та супровід.
2.	Цивільний сектор: швидка інтеграція, потреба в демонстраціях та тестуваннях, швидке впровадження	Консультації з інтеграції у комерційні системи, навчальні демонстрації, технічна підтримка, оновлення ПЗ, гарантійне обслуговування.	Канал може включати дилерів, дистриб'юторів та технічні демонстраційні центри.	Комбінований канал продажу: прямий B2B для великих клієнтів, дилери та демонстраційні точки для малих і середніх клієнтів.

Таблиця 5.22.

Концепція маркетингових комунікацій

№ п/п	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1.	Військовий сектор. Довгострокові контракти, високі вимоги до сертифікації, пріоритет безпеки та конфіденційності	Прямі контакти з оборонними підприємствами, спеціалізовані виставки, військові конференції, професійні журнали, онлайн-зустрічі	Висока завадостійкість, сертифікація, надійність.	Донести надійність, безпеку, стійкість до перешкод, гарантію безперервної навігації.	«Надійна навігація в критичних умовах»
2.	Цивільний сектор. Швидка інтеграція, потреба у демонстраціях.	Дилерські мережі, онлайн-платформи, галузеві виставки, демонстраційні центри, соціальні мережі та вебінари.	Компактність, сумісність, точність, модульність.	Показати зручність інтеграції, точність та економію ресурсів, доступну ціну.	«Точна навігація для ефективної роботи»

Висновки

1. Ринкова комерціалізація можлива: є попит, ринок динамічний, рентабельність реальна у нішевому сегменті.
2. Перспективи впровадження позитивні: зацікавлені цільові групи, бар’єри високі, але інноваційність забезпечує конкурентоспроможність.
3. Оптимальна альтернатива: спеціалізація з елементами диференціації на концентрованому сегменті.
4. Подальша імплементація доцільна завдяки унікальності та відповідності потребам клієнт

ЗАГАЛЬНІ ВИСНОВКИ ПО РОБОТІ

1. У магістерській дисертації вирішено важливу науково-практичну задачу проєктування адаптивної антени для приймача систем глобальної супутникової радіонавігації, що забезпечує стійку роботу навігаційного обладнання в умовах інтенсивних радіоперешкод, проведено комплексний аналіз принципів функціонування фазованих антенних решіток. Визначено критичні параметри антенних решіток, що становить теоретичний фундамент для подальшого проєктування адаптивних систем.

2. Розроблено та досліджено методи адаптивної багатоканальної обробки сигналів для антенних решіток з подвійною поляризацією. Реалізовано та порівняно ефективність алгоритмів адаптації сімейства LMS, які демонструють швидку збіжність та низьку обчислювальну складність, придатну для реалізації на FPGA.

3. Проведено систематичний аналіз типів завад у ГНСС-системах, що дозволило класифікувати загрози за механізмом впливу та розробити специфічні методи протидії. Запропоновано методику паралельного кореляційного аналізу, що використовує апаратні можливості GPU для одночасної обробки множинних каналів антенної решітки.

4. Створено комплексну MATLAB-модель GPS-приймача з адаптивною антенною решіткою, що включає генерацію реалістичних GPS-сигналів, моделювання різних типів завад та імітацію роботи адаптивних алгоритмів придушення перешкод. Експериментально підтверджено працездатність розробленої системи у випадку гармонічних завад, модульованих вузькосмугових перешкод та гаусівського шуму, при цьому застосування багатоканальної адаптивної решітки забезпечило відновлення навігаційної точності до рівня, близького до роботи в умовах відсутності завад.

5. Достовірність отриманих результатів підтверджується використанням фундаментальних принципів теорії антен та цифрової обробки сигналів, застосуванням перевірених математичних моделей розповсюдження радіохвиль та апробованих адаптивних алгоритмів, а також узгодженістю результатів комп'ютерного моделювання з теоретичними очікуваннями. Реалізація алгоритмів у середовищі MATLAB з використанням стандартних бібліотек обробки сигналів забезпечує відтворюваність результатів та можливість їх верифікації.

6. Наукова новизна роботи полягає у комплексному підході до проєктування адаптивної антенної системи, що поєднує оптимізацію геометричних параметрів фазованої решітки, розробку модифікованих адаптивних алгоритмів обробки сигналів та архітектуру програмно-конфігурованого приймача з апаратним прискоренням на GPU. Практична цінність дослідження визначається можливістю безпосереднього впровадження розроблених методів та алгоритмів у проєктування комерційних ГНСС-приймачів нового покоління з підвищеною стійкістю до завад.

7. Результати дисертаційного дослідження рекомендується використовувати при розробці перспективних навігаційних систем для критичної інфраструктури, модернізації існуючого обладнання ГНСС у секторах, що потребують підвищеної надійності навігаційних вимірювань, а також у навчальному процесі закладів вищої освіти при викладанні дисциплін з антенної техніки та цифрової обробки сигналів. Подальші дослідження доцільно спрямувати на оптимізацію енергоспоживання адаптивних алгоритмів для мобільних застосувань, розробку методів захисту від складних багатопроменевих завад та інтеграцію адаптивних антенних решіток з іншими сенсорами у гібридних навігаційних системах.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Диаграммы направленности фазированной антенной решетки. Часть 1. Характеристики луча и множитель линейной решетки/ Peter DELOS, Bob BROUGHTON, Jon KRAFT: стаття, 2020 рік.

2. Диаграммы направленности фазированной антенной решетки. Часть 2. Лепестки решетки и смещение луча/ Peter DELOS, Bob BROUGHTON, Jon KRAFT: стаття, 2020 рік.

3. Null-steering LMS Dual-Polarised Adaptive Antenna Arrays for GPS [Електронний ресурс] – Режим доступу до ресурсу: https://www.scirp.org/html/320.html

4. Overview of Jamming Technology for Satellite Navigation/ Xiangjun Li, Lei Chen, Zukun Lu, FeixueWang, Wenxiang Liu, Wei Xiao and Peiguo Liu – Changsha: College of Electronic Science, National University of Defense Technology: Review, 2023 рік

5. A Real-Time Capable Software-Defined Receiver Using GPU for Adaptive Anti-Jam GPS Sensors [Електронний ресурс] – Режим доступу до ресурсу: https://www.mdpi.com/1424-8220/11/9/8966

6. GPS and How It Works [Електронний ресурс] – Режим доступу до ресурсу: https://www.mathworks.com/help/satcom/gs/gps-and-how-it-works.html

7. GPS Waveform Generation [Електронний ресурс] – Режим доступу до ресурсу: https://www.mathworks.com/help/satcom/ug/gps-waveform-generation.html

8. GPS Receiver Acquisition and Tracking [Електронний ресурс] – Режим доступу до ресурсу: https://www.mathworks.com/help/satcom/ug/gps_receiver_acquisition_and_tracking.html

9. End-to-End GPS Legacy Navigation Receiver Using C/A-Code [Електронний ресурс] – Режим доступу до ресурсу: https://www.mathworks.com/help/satcom/ug/end-to-end-gps-legacy-navigation-receiver-using-cacode.html

10. Simulate GPS Sensor Noise [Електронний ресурс] – Режим доступу до ресурсу: https://www.mathworks.com/help/nav/ug/simulate-gps-sensor-noise.html
11. GPS Receiver Acquisition and Tracking Using C/A-Code [Електронний ресурс] – Режим доступу до ресурсу: https://www.mathworks.com/help/satcom/ug/gps-receiver-acquisition-and-tracking-using-ca-code.html