МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Сумський державний університет

Факультет електроніки та інформаційних технологій

Кафедра комп'ютерних наук

«До захисту допущено»

В.о. завідувача кафедри

Ігор ШЕЛЕХОВ

_____

(підпис)

травня 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня бакалавр

зі спеціальності 122 - Комп'ютерних наук,

освітньо-наукової програми «Інформатика»

на тему: «Інформаційна система оперативного контролю стану виконуючого механізму»

здобувача групи ІНз – 01с Застави Д.С.

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

Дмитро ЗАСТАВА

(підпис)

Керівник,
кандидат технічних наук,
доцент

Віктор АВРАМЕНКО _____

(підпис)

Суми – 2024

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«Затверджую»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавра

зі спеціальності 122 - Комп'ютерних наук, освітньо-наукової програми «Інформатика»
здобувача групи ІНз – 01с Застави Д.С.

1. Тема роботи: «Інформаційна система оперативного контролю стану виконуючого механізму»
затверджую наказом по СумДУ від _____
2. Термін здачі здобувачем кваліфікаційної роботи_____
3. Вхідні дані до кваліфікаційної роботи _____
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити) 1) Аналіз проблеми предметної області, постановка й формування завдань дослідження. 2) Огляд архітектури та методів реалізації 3) Практична реалізація 4) Висновок.
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)_____
6. Консультанти до проекту (роботи), із зазначенням розділів проекту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «____» _____ 20 ___ р.

Завдання прийняв до виконання _____ Керівник _____
(підпис) (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1	Аналіз проблеми предметної області, постановка й формування завдань дослідження		
2	Огляд архітектури та методів реалізації		
3	Практична реалізація		
4	Аналіз отриманих результатів		
5	Оформлення пояснювальної записки до кваліфікаційної роботи		

Здобувач вищої освіти _____ Керівник _____
(підпис) (підпис)

АНОТАЦІЯ

Записка: 41 стор., 12 рис., 1 додаток, 20 літературних джерел.

Об’єкт дослідження — інформаційна система оперативного контролю виконуючого механізму системи автоматичного регулювання

Мета роботи — розробка інформаційної системи оперативного контролю виконуючого механізму системи автоматичного регулювання, орієнтованої на веб-сервіс.

Результати — проведено дослідження теоретичних аспектів систем автоматичного регулювання, проаналізовано вже існуючі системи, спроектовано та розроблено інформаційну систему оперативного контролю виконуючого механізму системи автоматичного регулювання.

СИСТЕМИ АВТОМАТИЧНОГО РЕГУЛЮВАННЯ, ОПЕРАТИВНИЙ КОНТРОЛЬ ВИКОНУЮЧОГО МЕХАНІЗМУ, C++

ЗМІСТ

ВСТУП

У наш час технології розвиваються з блискавичною швидкістю, і для того, щоб залишатися конкурентоспроможними та ефективними, важливо йти в ногу з прогресом. Особливо це стосується сфери автоматизації, де точність, надійність та швидкість реакції на зміни є ключовими факторами успіху. Автоматизовані системи контролю вже довели свою ефективність у підвищенні продуктивності та безпеки у різних галузях, від виробництва до повсякденного життя. І серцем цих систем є саме інформаційні системи оперативного контролю.

Метою кваліфікаційної роботи стало глибоке вивчення цих систем, розуміння принципів їх роботи та знаходження шляхів їх вдосконалення. Планом роботи було розібратися, як різні компоненти, такі, як: датчики, обчислювальні блоки та засоби передачі даних, взаємодіють між собою та впливають на загальну ефективність системи. Особливу увагу було приділено здатності цих систем швидко та адекватно реагувати на зміни у робочому середовищі, адже саме це є запорукою надійності та безпеки автоматизованих процесів.

Крім того, було досліджено, як інформаційні системи оперативного контролю інтегруються в загальну структуру автоматичного регулювання та взаємодіють з іншими її елементами. Це дозволило отримати краще розуміння їх ролі та потенціалу у контексті цілісної системи.

Також було зосередження на технічних та програмних аспектах цих систем. Розглядались питання безпеки, надійності та стабільності їх роботи. Особливу увагу приділено тому, як сучасні технології, такі як машинне навчання та штучний інтелект, можуть бути застосовані для вдосконалення інформаційних систем оперативного контролю. Ці технології відкривають нові можливості для аналізу даних, прогнозування та прийняття рішень в реальному

часі, що може суттєво підвищити ефективність та адаптивність автоматизованих систем.

Однак мета - не лише теоретичне дослідження, а й розробка практичних рекомендацій та рішень. Будуть запропоновані конкретні шляхи оптимізації існуючих систем та впровадження інноваційних підходів. Наші висновки та пропозиції будуть ґрунтуватися на глибокому аналізі реальних кейсів та врахуванні специфіки різних галузей застосування.

1 ТЕОРЕТИЧНА ЧАСТИНА

1.1 Дослідження актуальності проблеми

Розглянемо докладніше конкретні приклади та характеристики інформаційних систем оперативного моніторингу виконавчого механізму систем автоматичного управління (САУ):

1. Типологія та використання САУ:

- САУ можна класифікувати за сферою застосування (промисловість, енергетика, транспорт тощо), ступенем автоматизації (низький, середній, високий), методом управління (централізовані, децентралізовані), типом керованого об'єкта (технологічні процеси, енергетичні об'єкти, транспортні засоби) [1].

2. Засади функціонування та складові САУ:

- Системи базуються на принципі зворотного зв'язку, де сенсори використовуються для збирання інформації про стан об'єкта [2], яка порівнюється з заданими параметрами, і на основі відхилень ухвалюються рішення для коригування стану об'єкта [3].
- Ключові компоненти включають вимірювальні прилади, регулятор, виконавчі пристрої, об'єкт керування та системи зворотного зв'язку.

3. Режими функціонування САУ:

- САУ мають декілька режимів роботи: пуск, робочий, налаштування, "планова зупинка", аварійний. Кожен режим має свої особливості та цілі. Наприклад, у режимі запуску здійснюється діагностика технічного стану та приймаються рішення на основі аналізу даних, тоді як у робочому режимі забезпечується контроль якості продукції та функціонування допоміжних систем.

4. Переваги та недоліки:

- САУ забезпечують автоматизацію моніторингу та управління, підвищуючи ефективність і точність, проте мають певні недоліки, такі як складність проєктування, високі витрати на обладнання та ймовірність збоїв у роботі.

Рисунок 1.1 – Структурна схема системи автоматичного контролю

Ці системи активно використовуються в різноманітних сферах і відіграють вирішальну роль у зростанні продуктивності, забезпеченні безпеки та автоматизації процесів у сучасній індустрії.

Крім того, вони створюються на основі теорії автоматичного керування. Теорія автоматичного керування є фундаментальною складовою у проєктуванні та дослідженні систем моніторингу та управління [4].

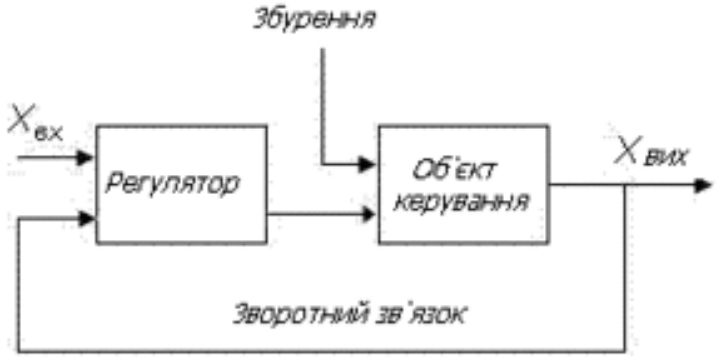

Рисунок 1.2 – схематичне представлення теорії автоматичного управління

Ось деякі з її основних концепцій і принципів:

1. Фундаментальні концепції та засади:

· Система: Це сукупність взаємопов'язаних компонентів, які взаємодіють для досягнення спільної цілі. Системи бувають фізичними, біологічними або інформаційними.

· Зворотний зв'язок: Процес, при якому вихідні дані системи використовуються для коригування вхідних даних, що дозволяє системі адаптуватися до змін зовнішніх умов.

· Керування: Процес ухвалення рішень і виконання дій для досягнення бажаного стану або поведінки системи.

· Передавальна функція: Математичний опис, що характеризує співвідношення між вхідними та вихідними сигналами системи.

· Стабільність: Здатність системи підтримувати свій стан або поведінку в межах прийнятних границь при впливі зовнішніх чинників.

· Оптимальне керування: Визначення такого управління, яке мінімізує заданий критерій ефективності.

2. Типологія систем керування:

· За структурою: Системи можуть бути простими (один елемент), складовими (кілька взаємодіючих елементів) або комплексними (багато взаємопов'язаних підсистем і компонентів).

3. Види систем керування:

· Розімкнені: Не отримують зворотної інформації про стан системи, генерують управління на основі заданих критеріїв або часових програм.

· Замкнені: Використовують зворотний зв'язок для коригування управління на основі актуального стану системи.

4. Сучасні методи керування:

· Нелінійне управління, теорія катастроф, адаптивне керування, робастні регулятори, ігрові методи, інтелектуальне управління.

5. Оптимізація керування:

· Пошук оптимальних параметрів налаштування регулятора для мінімізації відхилень регульованої величини від заданого значення.

Ця теорія має широке застосування в різноманітних сферах, зокрема в промисловості, на транспорті, в енергетиці та багатьох інших, де важливі автоматизація та точне управління [5].

1.2 Постановка задачі

Будь-яка система автоматичного регулювання (САР) режимних параметрів при протіканні технологічних процесів включає виконавчий механізм. Він виконує команди, що надходять від автоматичного регулятора. Зазвичай це колонка дистанційного управління (КДУ) з електричним, гідравлічним або пневматичним приводом, який переміщає виконавчий орган в задане положення від 0 до 100%. Наприклад, таким виконавчим органом може бути клапан для подачі дистильованої води в пароперегрівач для підтримки заданої температури пари перед турбіною. Щоб знизити температуру пари, збільшують подачу дистиляту в пароперегрівач. Якщо температуру потрібно підняти, клапан прикривають. Зазвичай статична характеристика виконавчого механізму є пропорційною. При відкритті клапана від 0 до 100% подача дистильованої води повинна збільшуватися пропорційно від 0 до 16000 кг/год. Практично для всіх приводів ця пропорційна залежність може порушуватися з деяких причин. Одна з них – ослаблення кріплення клапана, поява люфту у приводі. В результаті з'являється так званий "мертвий хід", коли привід переміщає шток

кріплення до клапана, але клапан якийсь час залишається нерухомим. Це порушує роботу САР. Тому ставиться наступна задача.

Розробити алгоритм і комп'ютерну програму для оперативного виявлення появи люфту в кріпленні штока до клапану подачі дистиляту в пароперегрівач. Передбачити комп'ютерне моделювання роботи системи.

1.3 Математична постановка задачі

На рисунку 1.3 приведена статична характеристика виконуючого механізму.

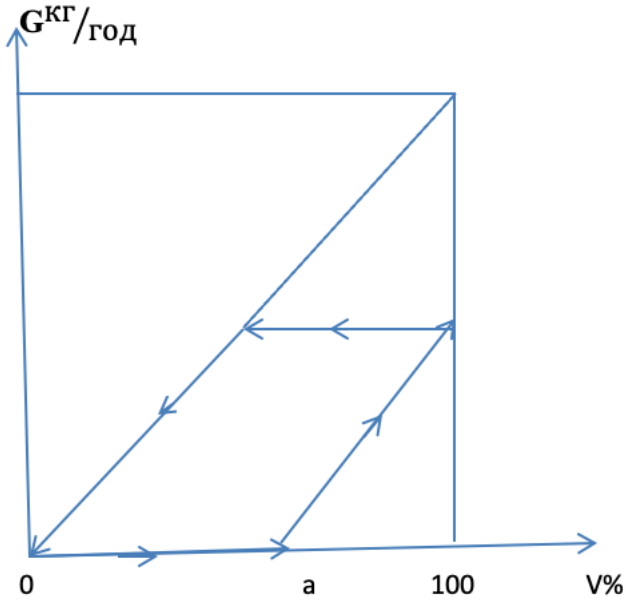

Рисунок 1.3– статична характеристика виконуючого механізму.

На рисунку 1.3 показано, що при переміщенні V% від 0 до a% електродвигуном штоку по команді від регулятора температури, внаслідок наявності люфту клапан не рухається. І тільки після a% він починає відкриватися і подавати дистилят. При досягнення V=100% клапан залишається не повністю відкритим. Подача дистиляту менша ніж 16тон/год. Тепер, якщо треба зменшити подачу дистиляту, знову певний час електродвигун рухає шток клапана (V зменшується), але клапан не рухається,

бо немає щеплення із штоком. І тільки, коли люфт буде «вибрано» і з'явиться щеплення між клапаном і його штоком, реально почнеться зменшення подачі дистиляту.

Позначимо:

t – час;

V – переміщення штоку в %;

Vmax - максимальне переміщення штоку;

G –подача дистиляту в пароперегрівач в кг/год;

Gmax – максимальна подача дистиляту кг/год;

a – ширина зони люфту в %;

dv – перша похідна переміщення штоку;

k – крутизна зміни подачі дистиляту в залежності від переміщення штоку

Залежність G від V описується наступним виразом:

$$G=\begin{cases} 0 \text{ при } 0<=V<a,\ dv>0 \\ (V-a)*k \text{ при } a<=V<Vmax,\ dv>0 \\ (Vmax-a)*k \text{ при } v>Vmax - a,\ dv<0 \\ V*k \text{ при } 0<=V<=Vmax-a,\ dv<0 \end{cases} \quad (1)$$

Ставиться задача по поточним значенням V(t) і G(t) виявляти подію, коли ширина зони люфту a>0.

2 ОГЛЯД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТА ВИБІР МЕТОДУ РОЗВ'ЯЗАННЯ ЗАДАЧІ

2.1 Огляд програмного забезпечення

SCADA (Supervisory Control and Data Acquisition) є широко використовуваним типом програмного забезпечення, призначеного для промислової автоматизації, моніторингу та керування промисловими процесами [10]. Ось декілька прикладів SCADA систем:

1. **AVEVA Intouch**
    - **Опис**: Програмне забезпечення для великих та малих промислових підприємств, яке дозволяє покращити надійність обладнання та оперативну ефективність.
    - **Переваги**: Підтримка реального часу, можливість керувати складними операціями.
    - **Недоліки**: Може бути складним у налаштуванні та вимагати спеціалізованих знань.
    - **Застосування**: Хімічна, морська, енергетична промисловість, виробництво, харчові продукти та напої, енергетика та комунальні послуги.

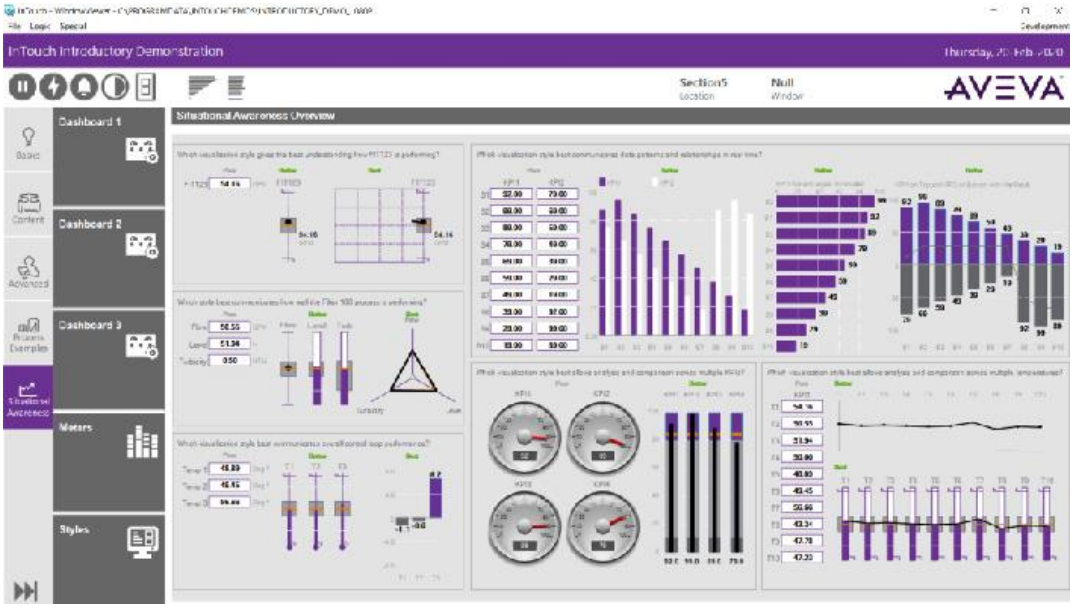

Рисунок 2.1 – графічний інтерфейс програми AVEVA

2. **FactoryTalk View**

- **Опис**: SCADA-система для розширених промислових застосувань, розроблена Rockwell Automation.
- **Переваги**: Підтримка декількох користувачів та серверів, інструменти для створення додатків моніторингу процесів.
- **Недоліки**: Може вимагати додаткового обладнання для підключення до різних типів пристроїв.
- **Застосування**: Від процесних до пакетних та дискретних застосувань у промисловості.

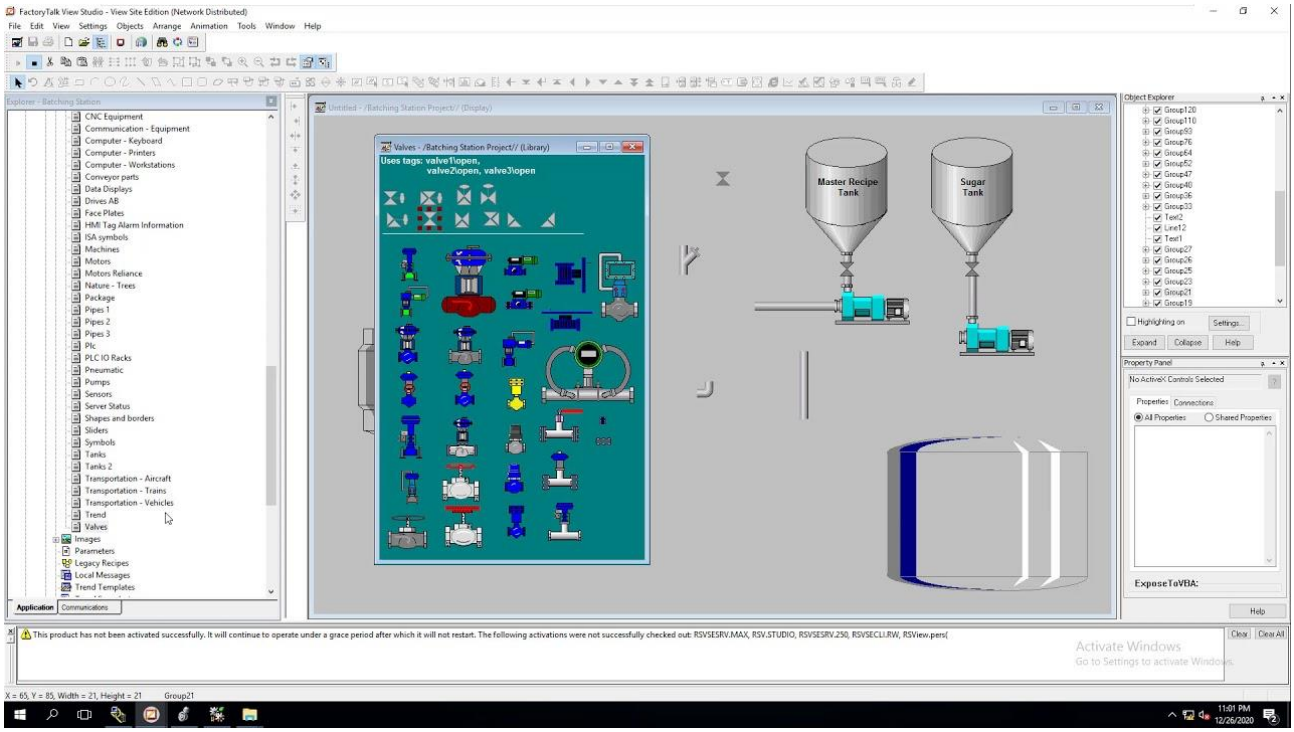

Рисунок 2.2 – приклад використання FactoryTalk View Site Edition

3. **CIMPLICITY SCADA**

- **Опис**: Система для великих виробничих підприємств з можливістю дистанційного керування процесами.
- **Переваги**: Швидка відповідь, зниження витрат, висока прибутковість.
- **Недоліки**: Може вимагати високого рівня технічної підтримки.
- **Застосування**: Автомобільна промисловість, авіація, хімічна промисловість, електроенергетика, виробництво, нафта та газ.

Рисунок 2.3 – використання системи CIMPLICITY SCADA для відслідковування руху та наявності палива

4. **SIMATIC WinCC V7**

- **Опис**: SCADA-система для моніторингу промислових процесів з високопродуктивними функціями.
- **Переваги**: Підходить для будь-якого застосування, високопродуктивне архівування даних.
- **Недоліки**: Може вимагати складної настройки та спеціалізованих знань.
- **Застосування**: Моніторинг автоматизованих процесів.

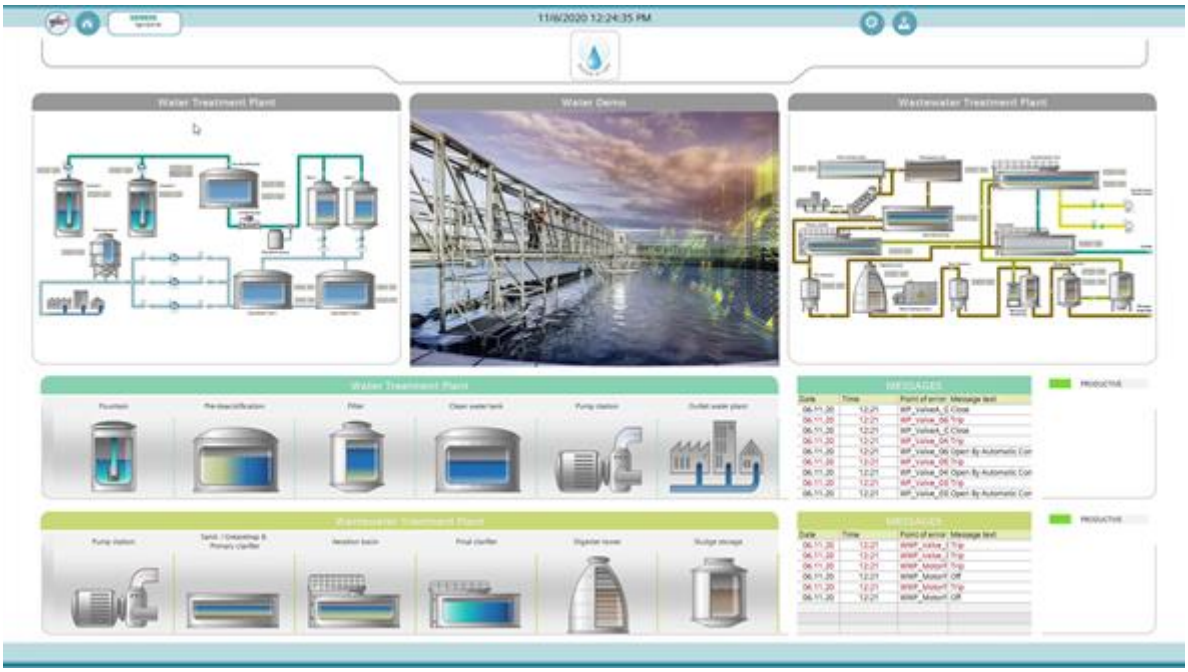

Рисунок 2.4 – приклад використання Simatic WinCC v7

5. **Ignition SCADA**

- **Опис**: SCADA-система з можливістю підключення до SQL баз даних, мобільним доступом та функціями тривог.
- **Переваги**: Легкість доступу до даних процесів, підключення до IIOT пристроїв.
- **Недоліки**: Може вимагати розширених налаштувань для підключення до різних типів обладнання.
- **Застосування**: Контроль процесів з використанням потужних HMI-інтерфейсів.

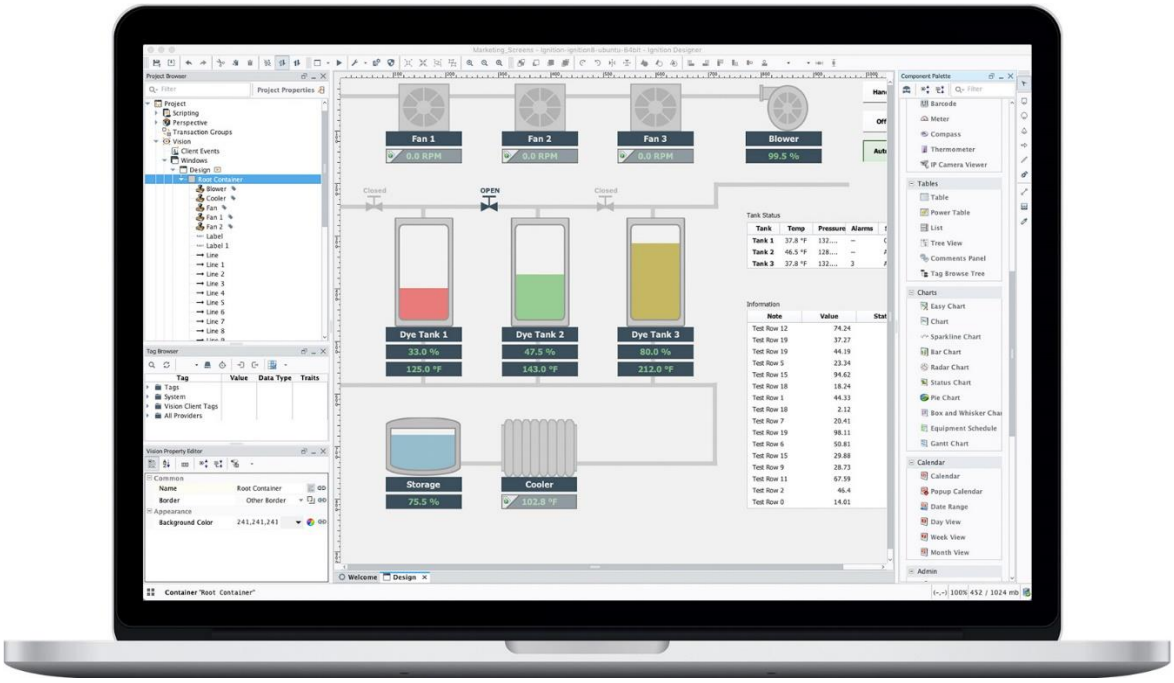

Рисунок 2.5 – використання Ignition Scada на одному з цехів фабрики

Кожна з цих систем має свої специфічні особливості та може бути застосована в різних сферах, залежно від потреб та вимог користувачів [6]. Важливо враховувати ці переваги та недоліки при створенні майбутнього аналогічного продукту.

2.2 Вибір методу розв’язання задачі

Добуток дистиляту через клапан залежить не тільки від того на скільки він відкритий, але також від тиску, який створюють насоси. На практиці цей тиск випадково змінюється. Як результат, випадковим чином змінюється крутизна, тобто відношення G(t) до V(t).

Тому простим діленням G(t) на V(t) задачу розв’язати не можна. Потрібно кількісно оцінювати відхилення статичної характеристики виконуючого механізму від пропорціонального виду. Таким вимогам відповідає метод, який базується на функціях непропорційності.

При обчисленні непропорційності по похідній першого порядку необхідно мати перші похідні режимних параметрів. Для цього застосовується чисельний метод обчислення похідних по трьом точкам.

2.3 Хід розв'язання задачі

Згідно із [1-3] непропорційність z(t) по похідній першого порядку подачі дистиляту G(t) по V(t) має вид:

$$z(t) = @d^{(1)}_{v(t)}G(t) = \frac{G(t)}{V(t)} - \frac{dG/dt}{dV/dt} \quad (2)$$

При зростанні V (dv>0) доки V<a , G=0. Тому непропорційність z(t)=0.

У випадку, коли dv> і a<=V<=Vmax,

G(t)=(V(t)-a)*k.

Відповідно непропорційність z(t) обчислюється по формулі:

$$z(t)= \frac{(V(t)-a)*k}{V(t)} - \frac{((V(t)-a)*k)'}{V'(t)} = -\frac{a}{V} \quad (3)$$

При зменшенні V (dv<0) і V(t)>Vmax-a G=(Vmax-a)*k. В цьому випадку непропорційність

$$z(t)=-\frac{(Vmax-a)*k}{V} \quad (4)$$

Нарешті, коли dv<0 і 0<=V(t)<=Vmax-a, G(t) = k*V(t). Між ними існує пропорціональний зв'язок. Відповідна непропорціональність

$$Z(t)=\frac{k*V(t)}{V(t)} - \frac{k*V'(t)}{V'(t)} =0 \quad (5)$$

Таким чином, появу люфту в виконуючому механізму САР можна виявити оперативно по поточним значенням G(t) і V(t) шляхом обчислення непропорційності z(t) (2) по похідній першого порядку.

2.4 Алгоритм розв'язання задачі

1. Ініціалізація:
    - Встановити початкові значення для V(t), G(t), a, Vmax, і k.

- Визначити початковий час t=0 і початковий крок часу Δt.

2. Обчислення похідних:
   - Для кожного часу t, обчислити похідні $\frac{dV}{dt}$ і $\frac{dG}{dt}$.
3. Обчислення значення G(t):
   - Якщо $\frac{dV}{dt} > 0$, то:
     1. Якщо V < a, тоді G(t)=0.
     2. Якщо a ≤ V ≤ Vmax, тоді G(t)=(V(t)−a)·k.
   - Якщо $\frac{dV}{dt} > 0$, то:
     1. Якщо V > Vmax − a, тоді G(t)=(Vmax−a)·k.
     2. Якщо 0≤V≤Vmax−a, тоді G(t)=k·V(t).
4. Обчислення непропорційності z(t):
   - Якщо $\frac{dV}{dt} > 0$ і V < a, тоді z(t)=0.
   - Якщо $\frac{dV}{dt} > 0$ і a ≤ V ≤ Vmax, тоді:

$$z(t)= \frac{(V(t)-a)\cdot k}{V(t)} - \frac{((V(t)-a)\cdot k)'}{V'(t)} = -\frac{a}{V}$$

   - Якщо $\frac{dV}{dt} < 0$ і V > Vmax – a, тоді:

$$z(t)= -\frac{(Vmax-a)\cdot k}{V}$$

   - Якщо $\frac{dV}{dt} < 0$ і 0 ≤ V ≤ Vmax – a, тоді:

$$z(t)= \frac{k\cdot V(t)}{V(t)} - \frac{k\cdot V'(t)}{V'(t)} = 0$$

5. Перевірка на люфт:

- Якщо z(t)≠0, виявити наявність люфту в виконуючому механізмі САР.

6. Оновлення часу:

- Збільшити час на крок Δt і повторити з кроку 2.

Псевдокод алгоритму:

```
initialize(V, G, a, V_max, k, t = 0, delta_t)
while (condition):
    dV_dt = compute_derivative(V, t)
    dG_dt = compute_derivative(G, t)

    if dV_dt > 0:
        if V < a:
            G = 0
        elif a <= V <= V_max:
            G = (V - a) * k
    elif dV_dt < 0:
        if V > V_max - a:
            G = (V_max - a) * k
        elif 0 <= V <= V_max - a:
            G = k * V
    if dV_dt > 0 and V < a:
        z = 0
    elif dV_dt > 0 and a <= V <= V_max:
        z = ((V - a) * k) / V - derivative((V - a) *
k, t) / derivative(V, t)
    elif dV_dt < 0 and V > V_max - a:
        z = -((V_max - a) * k) / V
    elif dV_dt < 0 and 0 <= V <= V_max - a:
       z = 0
    if z != 0:
        detect_luft()
    t = t + delta_t
```

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Програмна реалізація

Для вирішення даної задачі за допомогою алгоритму імплементуємо програмний код. В цьому випадку оптимальним буде використання мови програмування C++. Дане рішення обумовлено такими перевагами:

1. Висока продуктивність. C++ являє собою мову програмування, що компілюється в машинний код. За рахунок цього забезпечується висока швидкість виконання програм, а відповідно є вищою швидкість числових розрахунків.

2. Контроль над пам’яттю. В даній мові програмування є можливість низькорівневого управління пам’яттю, що забезпечує оптимізацію використання ресурсів, а відповідно і високу ефективність обчислень.

3. Набір бібліотек та інструментів. Завдяки вбудованим бібліотекам iostream/fstream для введення/виведення та cmath для математичних функцій, є можливість без пошуку сторонніх інструментів втілити у вигляді коду алгоритм обчислення.

C++ є оптимальним вибором для задач, де критично важлива висока продуктивність, ефективне використання пам'яті та доступ до низькорівневих операцій. У випадку з алгоритмом, який обчислює числові похідні та обробляє великі масиви даних, C++ забезпечує необхідну швидкість виконання та ефективність ресурсів, що робить його ідеальним вибором для таких задач.

Перейдемо безпосередньо до програмної реалізації. Почнемо з імпортування потрібних бібліотек. Цей розділ виглядатиме наступним чином:

```
#include <iostream>
#include <fstream>
#include <math.h>
#include <stdio.h>
```

```
#define N  200
```

Розберемо кожен імпорт по порядку:

- `#include <iostream>`: підключає бібліотеку для введення-виведення.
- `#include <fstream>`: підключає бібліотеку для роботи з файлами.
- `#include <math.h>`: підключає математичні функції.
- `#include <stdio.h>`: підключає стандартні бібліотеки введення-виведення на мові C.
- `#define N 200`: визначає макрос N зі значенням 200. Це кількість елементів у масивах.

Також нам треба визначити глобальні змінні, вони виглядатимуть наступним чином:

```
using namespace std;
double a = 0, k, h = 1, Gmax = 16000, Vmax = 100;
```

Першим рядком визначаємо простір імен, що містить у собі стандартні методи, такі як cout (вивід), cin (ввід), vector та string. У другому рядку визначаються глобальні змінні. Кожна з них відповідає за наступне:

1. a - ширина зони люфту, яка за замовчуванням дорівнює 0.
2. k – крутизна зміни подачі дистиляту в залежності від переміщення штоку.
3. h – крок, що використовується при обчисленні похідних.
4. Gmax – максимальна подача дистиляту кг/год.
5. Vmax – максимальне переміщення штоку.

Перейдемо до функції для обчислення похідної. Виглядатиме вона наступним чином:

```
void proizv(double y[], double dy[], int n)
{
    for (int i = 1; i < n - 1; i++)
    {
        dy[i] = (y[i + 1] - y[i - 1]) / (2 * h);
    }
```

```
    return;
}
```

Розберемо цю функцію по порядку. Наша функція для обчислення похідної називається proizv, яка приймає наступні аргументи:

- `y[]`: Вхідний масив.
- `dy[]`: Масив для зберігання похідних.
- `n`: Розмір масиву.

Цикл for у ній проходить через всі елементи масиву (крім першого та останнього) і обчислює центральну різницю (dy) за формулою $(y[i+1] - y[i-1])/(2 * h)$.

Також нам потрібна функція для обчислення подачі дистиляту у кг/год. Дана функція виглядатиме наступним чином:

```
    double f(double v, double dv)
{
    double r = 0;

    if (v >= 0 && v <= a && dv > 0)
    {
        r = 0;
    }
    else if (a < v && v <= Vmax && dv > 0)
    {
        r = (v - a) * k;
    }
    else if (v > Vmax - a && v <= Vmax && dv < 0)
    {
        r = (Vmax - a) * k;
    }
    else if (v == Vmax)
    {
        r = (Vmax - a) * k;
    }
    else
    {
        r = v * k;
    }
    return r;
}
```

Дана функція приймає в якості аргументів значення швидкості (v) та похідну швидкості (dv). Сама функція робить різні обчислення при різних умовах, а саме:

1. Якщо v більше або дорівнює 0 чи менше або дорівнює a, похідна швидкості є позитивною, отже G = 0.
2. Якщо v більше ніж a, менше або дорівнює максимальному переміщенню штоку, похідна швидкості є позитивною, то G обчислюється за формулою (v – a) * k.
3. Якщо v більше ніж різниця Vmax та a, менше або дорівнює Vmax, похідна швидкості є негативною, то G обчислюється за формулою (Vmax – a) * k.
4. Якщо v дорівнює Vmax, отже G також розраховується за формулою (Vmax – a) * k.
5. В будь-якому іншому випадку, G = v * k.

Ось таким чином виглядатиме головна функція програми:

```
int main()
{
    double G[N+1], V[N+1], dV[N+1], dG[N+1], Disp[N+1];
    ofstream fout("zastava.txt");
    cout << "Enter a=" << '\n';
    cin >> a;
    cout << "Enter Gmax=" << '\n';
    cin >> Gmax;
    k = Gmax / Vmax;

    for (int i = 0; i < N; i++)
    {
        if (i < N / 2)
            V[i] = i; // зростання від 0 до максимуму
100%
        else
            V[i] = N - i; // зменшення від максимуму 100%
до 0%
```

```
    }

    proizv(V, dV, N); // обчислює похідну для V

    for (int ii = 0; ii < N + 1; ii++)
    {
        G[ii] = f(V[ii], dV[ii]); // обчислює значення G
    }

    proizv(G, dG, N); // обчислює похідну для G

    // Обчислює дисперсію G по V
    for (int jj = 1; jj < N; jj++)
    {
        if (V[jj] != 0 && dV[jj] != 0)
        {
            Disp[jj] = G[jj] / V[jj] - dG[jj] / dV[jj];
        }
        else
        {
            Disp[jj] = 1000; // будь-яке велике число
замість нескінченності
        }
        cout << "jj=" << jj << "  V=" << V[jj] << "  G="
<< G[jj] << "  Disp=" << Disp[jj] << '\n';
        fout << "jj=" << jj << "  V=" << V[jj] << "  G="
<< G[jj] << "  Disp=" << Disp[jj] << '\n';
    }

    fout.close();
    cout << "FINISH" << '\n';
}
```

Розберемо цю функцію, що за що відповідає та що отримаємо у кінці.

1. Оголошуються такі масиви, як G, V, dV, dG, Disp з розмірністю N+1.
2. Відкривається текстовий файл для запису результатів. В нашому випадку це буде файл “zastava.txt”.
3. У користувача запитуються значення a (ширина зони люфту) та Gmax (максимальна подача дистиляту у кг/год). На основі введених даних,

обчислюється k за формулою Gmax / Vmax, де Vmax це максимальне переміщення штоку.

4. Ініцалізується масив V, в якому:

   4.1. Перша половина (від 0 до N/2) заповнюється зростаючими значеннями у межах, про які вже згадано в дужках.

   4.2. Друга половина (від N/2 до N) заповнюється спадаючими значеннями від N/2 до 0.

5. Викликається функція proizv для обчислення похідної швидкості (dV) за даними масиву V.

6. Відбувається обчислення масиву G за допомогою функції f для кожного значення V та dV.

7. Знову викликається функція proizv, але вже для обчислення похідної подачі дистиляту (dG) за даними масиву G.

8. Потім відбувається обчислення дисперсії G по значенням V з такими умовами:

   8.1. Якщо V[jj] та dV[jj] не дорівнюють 0, то дисперсія обчислюється за формулою $\frac{G[jj]}{V[jj]} - \frac{dG[jj]}{dV[jj]}$.

   8.2. В будь-якому іншому випадку, дисперсія задається зі значенням 1000.

9. Результати виводяться у консоль, а також записуються у заздалегідь створений текстовий файл. Після цього файл закривається та виводиться повідомлення про завершення роботи.

3.2 Контрольні приклади

Перейдемо до тестування алгоритму, який втілений у вигляді програмного коду з використанням C++. Для цього використовуватимемо різнідані, щоб

розглянути чи коректно працює алгоритм при різних умовах. В кожному з тестових сценаріїв вводитимемо два значення – одне для ширини зони люфту (a), інше для максимальної подачі дистиляту кг/год (GMax).

В першому випадку використаємо значення a=10, GMax = 16000. Очікувані результати наступні:

1. Для V від 0 до 10, G дорівнюватиме 0.
2. Для V від 10 до 100, G пропорційно збільшуватиметься з кроком 160.

На рисунку 3.1 представлено фрагмент коду та консольне відображення скомпільованої програми.

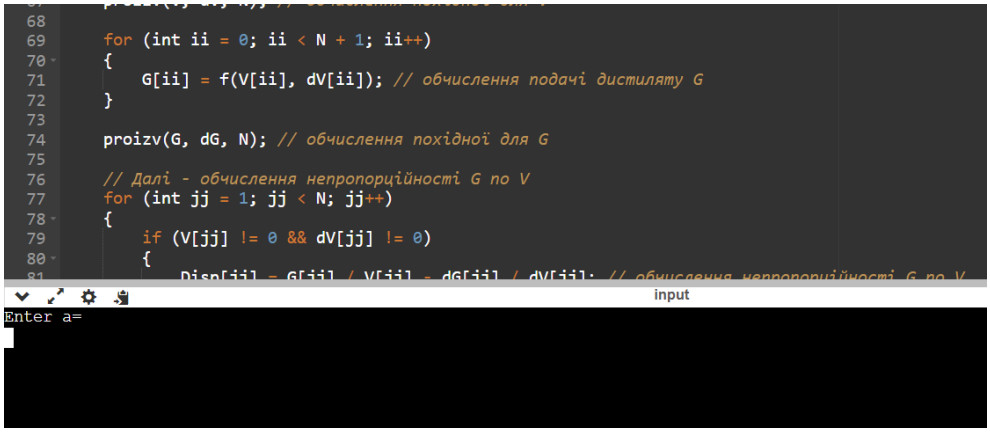

Рисунок 3.1 – фрагмент коду та консольне відображення скомпільованої програми

Як бачимо, програма компілюється коректно і можемо приступати до розрахунків. На рисунку 3.2 зображено розрахунки, що записані у файл, а також відображені в консолі.

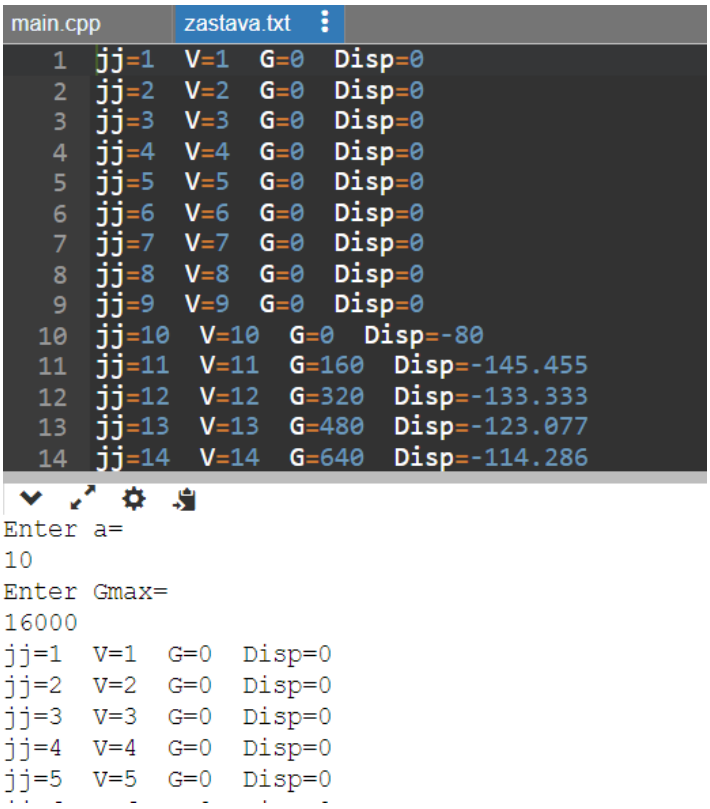

Рисунок 3.2 – відображення результатів в текстовому файлі та консолі

Аналізуючи роботу програми, можна зробити наступні висновки:

1. Розрахунки відображаються в консолі, а також записуються у текстовий файл. В нашому випадку, назва текстового файлу – zastava.txt.
2. Очікування, які були напередодні введення значень, були підтверджені.

   На рисунку 3.3 представлено фрагмент результатів розрахунків з параметрами a=10, GMax=16000.

```
jj=1  V=1  G=0  Disp=0
jj=2  V=2  G=0  Disp=0
jj=3  V=3  G=0  Disp=0
jj=4  V=4  G=0  Disp=0
jj=5  V=5  G=0  Disp=0
jj=6  V=6  G=0  Disp=0
jj=7  V=7  G=0  Disp=0
jj=8  V=8  G=0  Disp=0
jj=9  V=9  G=0  Disp=0
jj=10  V=10  G=0  Disp=-80
jj=11  V=11  G=160  Disp=-145.455
jj=12  V=12  G=320  Disp=-133.333
jj=13  V=13  G=480  Disp=-123.077
jj=14  V=14  G=640  Disp=-114.286
jj=15  V=15  G=800  Disp=-106.667
jj=16  V=16  G=960  Disp=-100
```

Рисунок 3.3 – результати розрахунків

Тепер спробуємо інші параметри, а саме a=20, Gmax=20000. Очікуваними є наступні результати:

1. G дорівнюватиме 0, коли V знаходиться в рамках від 0 до 20.
2. Якщо V знаходитиметься в рамках від 20 до 100, відбуватиметься збільшення G з кроком 200.

На рисунку 3.4 зображено результати при вказаних значеннях:

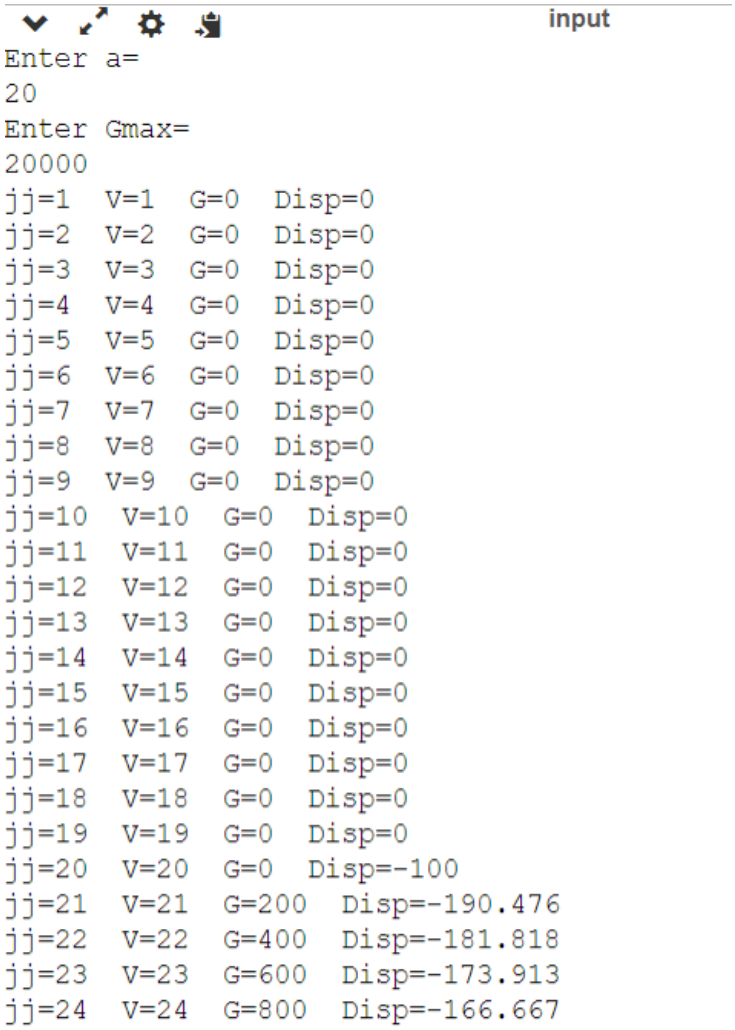

Рисунок 3.4 – результати при значеннях іншого прикладу

Як можемо побачити, фрагмент результатів також відповідає очікуванням.

Тепер спробуємо з іншими значеннями, а саме a=5, Gmax=15000. При таких значеннях очікуємо наступне:

1. G буде дорівнювати 0 при V від 0 до 5.
2. G буде збільшуватися з кроком 150 при V від 5 до 150.

На рисунку 3.5 представлено фрагмент результатів, які вказують на те, що вони відповідають очікуванням.

```
Enter a=
5
Enter Gmax=
15000
jj=1  V=1  G=0  Disp=0
jj=2  V=2  G=0  Disp=0
jj=3  V=3  G=0  Disp=0
jj=4  V=4  G=0  Disp=0
jj=5  V=5  G=0  Disp=-75
jj=6  V=6  G=150  Disp=-125
jj=7  V=7  G=300  Disp=-107.143
jj=8  V=8  G=450  Disp=-93.75
jj=9  V=9  G=600  Disp=-83.3333
jj=10  V=10  G=750  Disp=-75
jj=11  V=11  G=900  Disp=-68.1818
jj=12  V=12  G=1050  Disp=-62.5
jj=13  V=13  G=1200  Disp=-57.6923
jj=14  V=14  G=1350  Disp=-53.5714
jj=15  V=15  G=1500  Disp=-50
jj=16  V=16  G=1650  Disp=-46.875
jj=17  V=17  G=1800  Disp=-44.1176
jj=18  V=18  G=1950  Disp=-41.6667
jj=19  V=19  G=2100  Disp=-39.4737
```

Рисунок 3.5 – результати обчислень зі значеннями третього прикладу.

Спробуємо ще один приклад, в якому a=0, Gmax=16000. З такими значеннями очікуємо на те, що G збільшуватиметься для всіх значень V від 0 до 100 з кроком 160.

На рисунку 3.6 зображено результати обчислень з вищевказаними значеннями, що відповідають очікуванням.

```
Enter a=
0
Enter Gmax=
16000
jj=1  V=1  G=160  Disp=0
jj=2  V=2  G=320  Disp=0
jj=3  V=3  G=480  Disp=0
jj=4  V=4  G=640  Disp=0
jj=5  V=5  G=800  Disp=0
jj=6  V=6  G=960  Disp=0
jj=7  V=7  G=1120  Disp=0
jj=8  V=8  G=1280  Disp=0
jj=9  V=9  G=1440  Disp=0
jj=10  V=10  G=1600  Disp=0
jj=11  V=11  G=1760  Disp=0
jj=12  V=12  G=1920  Disp=0
jj=13  V=13  G=2080  Disp=0
jj=14  V=14  G=2240  Disp=0
jj=15  V=15  G=2400  Disp=0
jj=16  V=16  G=2560  Disp=0
jj=17  V=17  G=2720  Disp=0
jj=18  V=18  G=2880  Disp=0
jj=19  V=19  G=3040  Disp=0
jj=20  V=20  G=3200  Disp=0
jj=21  V=21  G=3360  Disp=0
jj=22  V=22  G=3520  Disp=0
```

Рисунок 3.6 – результати з використанням значень четвертого прикладу

В передостанньому прикладі використаємо значення a=50, Gmax=10000. Очікуваними будуть наступні результати:

1. При V від 0 до 50, G дорівнюватиме 0.

2. При V від 50 до 100, G пропорційно збільшуватиметься з кроком 100.

На рисунку 3.7 представлено результати, що відповідають очікуваним.

```
jj=41  V=41  G=0  Disp=0
jj=42  V=42  G=0  Disp=0
jj=43  V=43  G=0  Disp=0
jj=44  V=44  G=0  Disp=0
jj=45  V=45  G=0  Disp=0
jj=46  V=46  G=0  Disp=0
jj=47  V=47  G=0  Disp=0
jj=48  V=48  G=0  Disp=0
jj=49  V=49  G=0  Disp=0
jj=50  V=50  G=0  Disp=-50
jj=51  V=51  G=100  Disp=-98.0392
jj=52  V=52  G=200  Disp=-96.1538
jj=53  V=53  G=300  Disp=-94.3396
jj=54  V=54  G=400  Disp=-92.5926
jj=55  V=55  G=500  Disp=-90.9091
jj=56  V=56  G=600  Disp=-89.2857
jj=57  V=57  G=700  Disp=-87.7193
jj=58  V=58  G=800  Disp=-86.2069
jj=59  V=59  G=900  Disp=-84.7458
jj=60  V=60  G=1000  Disp=-83.3333
jj=61  V=61  G=1100  Disp=-81.9672
jj=62  V=62  G=1200  Disp=-80.6452
jj=63  V=63  G=1300  Disp=-79.3651
jj=64  V=64  G=1400  Disp=-78.125
jj=65  V=65  G=1500  Disp=-76.9231
jj=66  V=66  G=1600  Disp=-75.7576
jj=67  V=67  G=1700  Disp=-74.6269
```

Рисунок 3.7 – результати зі значеннями п’ятого прикладу

В останньому прикладі спробуємо вказати у вигляді значення рядок. Не дивлячись на те, що C++ має строгу типізацію, тобто очікувані типи даних вказуються при компіляції, все ж не варто виключати людський фактор, що спробує в якості значення передати рядок. На рисунку 3.8 представлено поведінку алгоритму, якщо він отримує на вхід який-небудь рядок.

```
Enter a=
loremipsum
Enter Gmax=
jj=1  V=1  G=160  Disp=0
jj=2  V=2  G=320  Disp=0
jj=3  V=3  G=480  Disp=0
jj=4  V=4  G=640  Disp=0
jj=5  V=5  G=800  Disp=0
jj=6  V=6  G=960  Disp=0
jj=7  V=7  G=1120  Disp=0
jj=8  V=8  G=1280  Disp=0
jj=9  V=9  G=1440  Disp=0
jj=10  V=10  G=1600  Disp=0
jj=11  V=11  G=1760  Disp=0
jj=12  V=12  G=1920  Disp=0
jj=13  V=13  G=2080  Disp=0
jj=14  V=14  G=2240  Disp=0
jj=15  V=15  G=2400  Disp=0
jj=16  V=16  G=2560  Disp=0
jj=17  V=17  G=2720  Disp=0
jj=18  V=18  G=2880  Disp=0
jj=19  V=19  G=3040  Disp=0
jj=20  V=20  G=3200  Disp=0
jj=21  V=21  G=3360  Disp=0
jj=22  V=22  G=3520  Disp=0
jj=23  V=23  G=3680  Disp=0
jj=24  V=24  G=3840  Disp=0
jj=25  V=25  G=4000  Disp=0
jj=26  V=26  G=4160  Disp=0
jj=27  V=27  G=4320  Disp=0
```

Рисунок 3.8 – поведінка алгоритму при отриманні рядкового значення

Результати ідентичні тим, що були отримані при значеннях з четвертого прикладу. Що, власне, відбулось? Повернемось до одного з перших рядків нашої програми, а саме:

```
double a = 0, k, h = 1, Gmax = 16000, Vmax = 100;
```

Цей рядок задає глобальні змінні. Як можемо помітити, змінні a та Gmax мають вказані стандартні значення. Тобто, програма просто проігнорувала рядок та продовжила своє виконання зі значеннями, які попередньо було вказано як стандартні. Це є нормальною поведінкою при такому сценарії.

Окрім цього, підтвердились також ще деякі загальні очікування, а саме:

1. В кожному прикладі розмір масиву дорівнював 200.

2. Disp відображав значення непропорційності там, де це варто було очікувати.

Отже, згідно тестування застосунку та контрольних прикладів при різних сценаріях, можемо зробити висновок що алгоритм працює справно.

ВИСНОВКИ

У цьому дослідженні було здійснено комплексний аналіз інформаційної системи оперативного моніторингу, яка відіграє вирішальну роль в ефективному управлінні виконавчими механізмами систем автоматичного регулювання.

Вивчення теоретичних основ предметної області дозволило з'ясувати ключові принципи та механізми роботи таких систем, а також значення їх використання в сучасних промислових та виробничих процесах.

Потім було проаналізовано та обрано мови програмування і інструменти для створення власної інформаційної системи. Вибір зупинився на мові C++ , адже вона є найефективнішою та найшвидшою для таких складних розрахунків.

Було визначено, за яким алгоритмом вирішуватиметься задача, а також було надано псевдокод алгоритму.

Після програмної реалізації, було проведено тестування програмного коду в різноманітних сценаріях. Очікування, які були напередодні перевірки сценаріїв, було підтверджено, програмний код працює коректно.

СПИСОК ЛІТЕРАТУРИ

1. Advanced Control Engineering" [Електронний ресурс] - режим доступу: https://www.elsevier.com/books/advanced-control-engineering/9780750633727.
2. "Digital Control Systems" [Електронний ресурс] - режим доступу: https://www.wiley.com/en-us/Digital+Control+Systems-p-9781119262202.
3. "Principles of Control Systems" [Електронний ресурс] - режим доступу: https://www.springer.com/gp/book/9783030300150.
4. "Control Systems for Electrical Engineering" [Електронний ресурс] - режим доступу: https://www.elsevier.com/books/control-systems-for-electrical-engineering/9780128175878.
5. "Mechatronics: Principles and Applications" [Електронний ресурс] - режим доступу: https://www.springer.com/gp/book/9783030333226.
6. "Application of Artificial Intelligence in Industrial Automation Control" [Електронний ресурс] - режим доступу: https://www.researchgate.net/publication/341874437_Application_of_Artificial_Intelligence_in_Industrial_Automation_Control.
7. Авраменко В.В.,ХАРАКТЕРИСТИКИ НЕПРОПОРЦІЙНОСТІ ЧИСЛОВИХ ФУНКЦІЙ, РДАСНТІ 27.23.15, Суми - 1997
8. Авраменко, В.В. Характеристики непропорциональности числовых функций и их применение при решении задач диагностики [Текст] / В.В. Авраменко // Вісник СумДУ. – 2000. № 16. – С.12 – 20.
9. Standard C++ [Електронний ресурс] - режим доступу: https://isocpp.org/
10. Control Systems for Precision Machines" [Електронний ресурс] - режим доступу: https://link.springer.com/chapter/10.1007/978-981-15-2449-0_2.
11. "Operational Technology and Information Technology in Industrial Control Systems" [Електронний ресурс] - режим доступу: https://link.springer.com/chapter/10.1007/978-3-030-63206-0_6.
12. "What is an Actuator? Types Principles and Applications" [Електронний ресурс] - режим доступу: https://www.wevolver.com/article/what-is-an-actuator-types-principles-and-applications.

13."Automatic Control System - an overview" [Електронний ресурс] - режим доступу: https://www.sciencedirect.com/topics/engineering/automatic-control-system.

14."Actuators explained: Types of actuators" [Електронний ресурс] - режим доступу: https://www.controleng.com/articles/actuators-explained-types-of-actuators/.

15."What is an automatic control system?" [Електронний ресурс] - режим доступу: https://automationforum.co/what-is-an-automatic-control-system/.

16."Electronic Control System of Automatic Transmission" [Електронний ресурс] - режим доступу: https://link.springer.com/chapter/10.1007/978-981-15-0445-9_4.

17."Automatic control systems: optimization and sensitivity study" [Електронний ресурс] - режим доступу: https://www.researchgate.net/publication/336858123_Automatic_control_systems_optimization_and_sensitivity_study.

18."A brief history of automatic control" [Електронний ресурс] - режим доступу: https://ieeexplore.ieee.org/document/932727.

19."Modern Control Engineering" [Електронний ресурс] - режим доступу: https://www.pearson.com/store/p/modern-control-engineering/P100000709541.

20."Industrial Automation and Control" [Електронний ресурс] - режим доступу: https://www.elsevier.com/books/industrial-automation-and-control/9780128115590.

21."Feedback Control of Dynamic Systems" [Електронний ресурс] - режим доступу: https://www.pearson.com/store/p/feedback-control-of-dynamic-systems/P100000711781.

22."Control Systems Engineering" [Електронний ресурс] - режим доступу: https://www.wiley.com/en-us/Control+Systems+Engineering-p-9781119493149.

23."Modeling and Control of Complex Systems" [Електронний ресурс] - режим доступу: https://www.springer.com/gp/book/9783319734504.

ДОДАТКИ

Додаток А. Вихідний код продукту

```
#include <iostream>
#include<fstream>
#include<math.h>
#include<stdio.h>
#define N  200
using namespace std;

double a = 0, k, h = 1, Gmax = 16000, Vmax = 100;

void proizv(double y[], double dy[], int n)
{
    for (int i = 1; i < n - 1; i++)
    {
        dy[i] = (y[i + 1] - y[i - 1]) / (2 * h);
    }
    return;
}

double f(double v, double dv)
{
    double r = 0;

    if (v >= 0 && v <= a && dv > 0)
    {
        r = 0;
    }
    else if (a < v && v <= Vmax && dv > 0)
    {
        r = (v - a) * k;
    }
    else if (v > Vmax - a && v <= Vmax && dv < 0)
    {
        r = (Vmax - a) * k;
    }
    else if (v == Vmax)
    {
        r = (Vmax - a) * k;
```

```
    }
    else
    {
        r = v * k;
    }
    return r;
}

int main()
{
    double  G[N + 1], V[N + 1], dV[N + 1], dG[N + 1],
Disp[N + 1];
    ofstream fout("zastava.txt");

    cout << "Enter a=" << '\n';
    cin >> a;
    cout << "Enter Gmax=" << '\n';
    cin >> Gmax;
    k = Gmax / Vmax;

    for (int i = 0; i < N; i++)
    {
        if (i < N / 2)
            V[i] = i; // шток рухається від 0 по напрямку
до 100%
        else
            V[i] = N - i; // шток рухається від 100% по
напрямку до 0%
    }

    proizv(V, dV, N); // обчислення похідної для V

    for (int ii = 0; ii < N + 1; ii++)
    {
        G[ii] = f(V[ii], dV[ii]); // обчислення подачі
дистиляту G
    }

    proizv(G, dG, N); // обчислення похідної для G

    // Далі - обчислення непропорційності G по V
    for (int jj = 1; jj < N; jj++)
    {
        if (V[jj] != 0 && dV[jj] != 0)
```

```
        {
            Disp[jj] = G[jj] / V[jj] - dG[jj] / dV[jj];
// обчислення непропорційності G по V
        }
        else
            Disp[jj] = 1000; // Any big number instead of
infinity

        cout << "jj=" << jj << "  V=" << V[jj] << "  G="
<< G[jj] << "  Disp=" << Disp[jj] << '\n';
        fout << "jj=" << jj << "  V=" << V[jj] << "  G="
<< G[jj] << "  Disp=" << Disp[jj] << '\n';
    }

    fout.close();
    cout << "FINISH" << '\n';
}
```