ДОДАТОК А

Лістинг програми

```
# app.py - Flask сервер для інтеграції з ШІ-сервісом

from flask import Flask, request, jsonify
from flask_cors import CORS
import PyPDF2
import requests
import os
from datetime import datetime

app = Flask(__name__)
CORS(app)

# ЗАВАНТАЖЕННЯ ІНСТРУКЦІЙ

def load_instructions():
    """Завантаження інструкцій з PDF файлу"""
    instructions = {}
    try:
        with open('instructions.pdf', 'rb') as file:
            reader = PyPDF2.PdfReader(file)

            if len(reader.pages) >= 5:
                instructions['high_temperature'] = reader.pages[0].extract_text()
                instructions['high_pressure'] = reader.pages[1].extract_text()
                instructions['low_level'] = reader.pages[2].extract_text()
                instructions['leak'] = reader.pages[3].extract_text()
                instructions['high_flow'] = reader.pages[4].extract_text()
            else:
                print("PDF містить менше 5 сторінок")

    except FileNotFoundError:
        print("Файл instructions.pdf не знайдено")
    except Exception as e:
        print(f"Помилка завантаження PDF: {e}")

    return instructions

INSTRUCTIONS = load_instructions()
print(f"Завантажено інструкції для {len(INSTRUCTIONS)} типів ситуацій")

# ГЕНЕРАЦІЯ РЕКОМЕНДАЦІЙ
```

```
def generate_recommendation(situation_desc, instruction):
    """Генерація рекомендації через Groq API"""

    api_key = os.getenv('GROQ_API_KEY')
    if not api_key:
        return "ШІ-сервіс недоступний"

    # Обрізаємо інструкцію якщо довга
    if len(instruction) > 2000:
        instruction = instruction[:2000] + "..."

    try:
        response = requests.post(
            'https://api.groq.com/openai/v1/chat/completions',
            headers={
                'Authorization': f'Bearer {api_key}',
                'Content-Type': 'application/json'
            },
            json={
                'model': 'mixtral-8x7b-32768',
                'messages': [
                    {
                        'role': 'system',
                        'content': 'Ти асистент оператора. Надавай чіткі рекомендації
українською.'
                    },
                    {
                        'role': 'user',
                        'content':
f"{situation_desc}\n\nІнструкція:\n{instruction}\n\nНадай рекомендації."
                    }
                ],
                'temperature': 0.3,
                'max_tokens': 400
            },
            timeout=10
        )

        if response.status_code == 200:
            return response.json()['choices'][0]['message']['content']
        else:
            print(f"Groq API помилка: {response.status_code}")
            return "Помилка генерації рекомендації"

    except Exception as e:
```

```
        print(f"Помилка: {e}")
        return "ШІ-сервіс недоступний"

# REST API ENDPOINTS

@app.route('/analyze', methods=['POST'])
def analyze_situation():
    """Обробка аварійної ситуації"""

    try:
        data = request.json
        situation_type = data.get('type')
        params = data.get('params', {})
        message = data.get('message', '')

        print(f"Отримано запит: {situation_type}")

        # Формування опису ситуації
        situation_desc = f"""
        Аварійна ситуація: {message}

        Поточний стан:
        - Температура: {params.get('T', 'N/A')} °C
        - Тиск: {params.get('P', 'N/A')} бар
        - Рівень: {params.get('L', 'N/A')}%
        - Витрата: {params.get('F', 'N/A')}%
        """

        # Отримання інструкції
        instruction = INSTRUCTIONS.get(situation_type, "Інструкція не
знайдена")

        # Генерація рекомендації
        recommendation = generate_recommendation(situation_desc, instruction)

        return jsonify({
            'status': 'success',
            'recommendation': recommendation,
            'timestamp': datetime.now().isoformat()
        })

    except Exception as e:
        print(f"Помилка: {e}")
        return jsonify({
            'status': 'error',
```

```
            'recommendation': 'Помилка обробки запиту',
            'timestamp': datetime.now().isoformat()
        }), 500

@app.route('/health', methods=['GET'])
def health_check():
    """Перевірка стану сервера"""
    return jsonify({
        'status': 'online',
        'instructions_loaded': len(INSTRUCTIONS),
        'timestamp': datetime.now().isoformat()
    })

# ЗАПУСК СЕРВЕРА

if __name__ == '__main__':
    print("\n" + "="*60)
    print("Flask сервер запускається...")
    print("="*60)
    print(f"Інструкції: {len(INSTRUCTIONS)} типів")
    print(f"API ключ: {'Yes' if os.getenv('GROQ_API_KEY') else 'No'}")
    print("="*60 + "\n")

    app.run(host='0.0.0.0', port=5000, debug=True)
```

ДОДАТОК Б
Апробація наукових результатів досліджень

Міністерство освіти і науки України

NURE

Харківський національний університет радіоелектроніки

ЗБІРНИК

студентських наукових статей

«Автоматизація та приладобудування»

«Automation and Development of Electronic Devices»

ADED-2025

(Випуск 2)

[електронне видання]

KITAP

http://nure.ua/department/kafedra-komp-yuterno-integrovanih-tehnologiy-avtomatizatsiyi-ta-mehatroniki-kitam

iTeD

http://itez.zntu.edu.ua/

http://kafea.kdu.edu.ua

Харків 2025

УДК 004.896; 681.5

МІНІМІЗАЦІЯ ЛЮДСЬКОГО ФАКТОРУ В ПРОМИСЛОВІЙ АВТОМАТИЗАЦІЇ ЗАСОБАМИ ІНТЕЛЕКТУАЛЬНИХ СИСТЕМ ПІДТРИМКИ РІШЕНЬ

Д.А. Рябовол
Харківський національний університет радіоелектроніки
Україна, 61166, Харків, пр. Науки 14
E-mail: dmytro.riabovol@nure.ua

В роботі розглянуте актуальне питання інтелектуалізації систем оперативно-диспетчерського контролю промислових підприємств. Проведено аналіз існуючих SCADA-систем та технологій штучного інтелекту для промислових застосувань. Розроблено прототип системи інтелектуальної підтримки прийняття рішень на базі Node-RED та Python з інтеграцією великих мовних моделей. Система автоматично аналізує технологічні параметри, детектує аларми та надає операторам структуровані рекомендації на основі технічної документації.

Ключові слова: оперативно-диспетчерський контроль, штучний інтелект, SCADA, кібер-фізічні системи, Node-RED, RAG-система

MINIMIZATION OF THE HUMAN FACTOR IN INDUSTRIAL AUTOMATION BY MEANS OF INTELLECTUAL DECISION SUPPORT SYSTEMS

D. Riabovol
Kharkiv National University of Radioelectronics
Ukraine, 61166, Kharkiv, Nauky av., 14
E-mail: dmytro.riabovol@nure.ua

The topical issue of intellectualization of supervisory control systems at industrial enterprises is considered in the work. The analysis of existing SCADA systems and artificial intelligence technologies for industrial applications is carried out. A prototype of an intelligent decision support system based on Node-RED and Python with integration of large language models has been developed. The system automatically analyzes technological parameters, detects alarms and provides operators with structured recommendations based on technical documentation.

Key words: supervisory control, artificial intelligence, SCADA, cyber-physical systems, Node-RED, RAG-system

АКТУАЛЬНІСТЬ РОБОТИ. Сучасне промислове виробництво характеризується високим рівнем автоматизації та складністю технологічних процесів. Оператори диспетчерських пунктів щодня приймають критичні рішення, які безпосередньо впливають на безпеку, продуктивність та якість продукції. За статистикою промислових підприємств, до 70% позаштатних ситуацій пов'язані з помилками операторів при інтерпретації алармів або несвоєчасним реагуванням на відхилення технологічних параметрів [1]. Особливо критичною є ситуація під час навчання нових операторів або при виникненні рідкісних нештатних сценаріїв, коли досвіду персоналу може бути недостатньо.

Традиційні SCADA-системи (Supervisory Control and Data Acquisition) забезпечують візуалізацію параметрів та сигналізацію про відхилення, але не надають операторам конкретних рекомендацій щодо дій у кожній конкретній ситуації. Технічна документація, інструкції з експлуатації та стандартні операційні процедури зазвичай існують у вигляді великих PDF-файлів, пошук потрібної інформації в яких під час аварійної ситуації займає критично багато часу.

Розвиток технологій штучного інтелекту (ШІ), зокрема великих мовних моделей (Large Language Models, LLM) та RAG-систем (Retrieval-Augmented Generation), відкриває нові можливості для інтелектуалізації промислових систем керування [2, 3]. Саме тому актуальним є розроблення систем інтелектуальної підтримки, які можуть миттєво надавати операторам структуровані рекомендації на основі аналізу технічної документації підприємства.

ВСТУП. Кібер-фізичні системи (КФС) є основою сучасної концепції Industry 4.0, що передбачає інтеграцію фізичних виробничих процесів з цифровими технологіями [4]. Одним з ключових компонентів КФС є системи оперативно-диспетчерського контролю (ОДК), які забезпечують моніторинг параметрів обладнання в реальному часі та управління технологічними процесами.

Існуючі комерційні SCADA-системи, такі як Siemens WinCC, Schneider Electric EcoStruxure, Honeywell Experion PKS, забезпечують високу надійність та широкі можливості інтеграції з промисловими протоколами (OPC UA, Modbus, Profinet). Однак вони є дорогими та мають закриту архітектуру, що ускладнює їх адаптацію під специфічні потреби конкретного підприємства [5].

Альтернативою є open-source платформи, зокрема Node-RED – візуальний інструмент для створення потоків даних IoT, розроблений IBM. Node-RED підтримує понад 4000 готових вузлів для інтеграції з різними протоколами та сервісами, що робить його особливо ефективним для швидкого прототипування систем моніторингу [6].

Останнім часом активно розвиваються дослідження у сфері застосування ШІ для промислової автоматизації. Проте більшість існуючих рішень зосереджені на детекції аномалій або передбачувальному обслуговуванні, і жодна з проаналізованих систем не реалізує повноцінної інтеграції RAG-підходу для автоматичного аналізу технічної документації та формування контекстуальних рекомендацій у реальному часі [7].

Метою даної роботи є розроблення прототипу системи інтелектуальної підтримки операторів для оперативно-диспетчерського контролю виробництва на базі кібер-фізичних систем керування з використанням технологій штучного інтелекту.

МАТЕРІАЛ І РЕЗУЛЬТАТИ ДОСЛІДЖЕНЬ. Для досягнення поставленої мети була розроблена модульна архітектура програмного модуля, що складається з чотирьох основних рівнів (рис. 1):

1. Рівень збору даних – реалізує симуляцію промислових датчиків, які генерують значення технологічних параметрів (тиск, температура, потік). У реальному виробництві дані надходять через протоколи OPC UA або Modbus.

2. Рівень обробки та аналізу – реалізований на платформі Node-RED, виконує моніторинг параметрів, детекцію порогових значень та формування алармів при виявленні відхилень.

3. Рівень інтелектуальної підтримки – Python-сервер з REST API для інтеграції з сервісами штучного інтелекту та генерації рекомендацій на основі технічної документації.

4. Рівень візуалізації – веб-інтерфейс на базі Node-RED Dashboard для відображення параметрів в реальному часі та рекомендацій від системи ШІ.

Потік даних в системі організований наступним чином: симульовані датчики генерують значення технологічних параметрів з періодичністю 3 секунди. Ці дані надходять до блоку аналізу в Node-RED, де порівнюються з критичними пороговими значеннями. Для демонстрації функціональності використовуються три параметри:

– тиск: діапазон 100-180 bar, критичне значення > 150 bar;

– температура: діапазон 20-100°C, критичне значення > 85°C;

– потік: діапазон 50-150 л/хв, критичне значення < 60 л/хв.

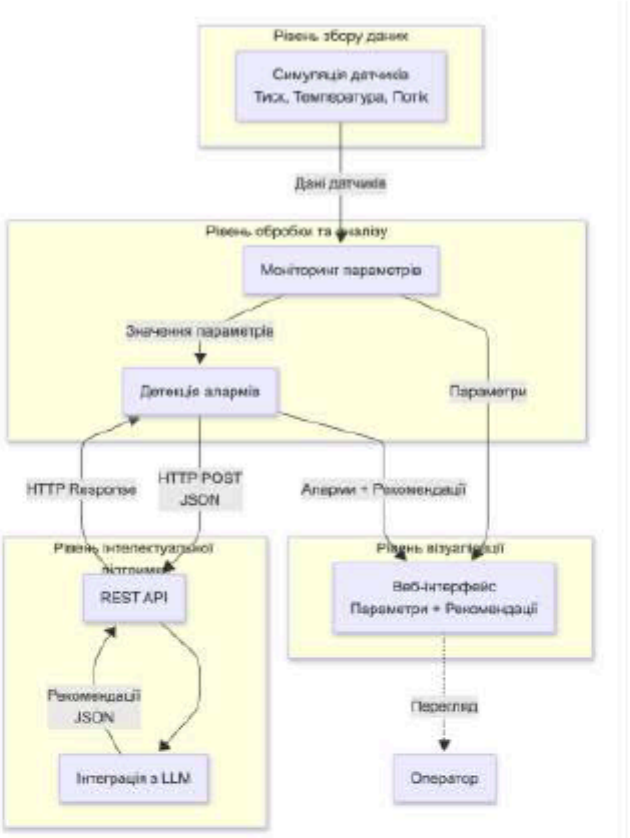

Рис. 1. Архітектура програмного модуля інтелектуальної підтримки операторів

При виявленні відхилення формується об'єкт аларму, що містить тип проблеми, поточне значення параметра, назву обладнання та мітку часу. Сформований аларм через HTTP-протокол передається до Python-сервера у форматі JSON.

Python-сервер на базі фреймворку Flask реалізує REST API для обробки алармів. При надходженні запиту сервер формує структурований prompt для великої мовної моделі, що містить:

– опис поточної ситуації (тип параметра, значення, поріг);
– контекст обладнання та технологічного процесу;
– вміст релевантних фрагментів з PDF-інструкцій;
– запит на конкретні рекомендації щодо дій оператора.

На поточному етапі прототипування створено тестовий набір PDF-інструкцій з процедурами реагування на типові аварійні ситуації. Вміст PDF-інструкцій передається як текстовий контекст безпосередньо в prompt до LLM. Це дозволяє моделі генерувати рекомендації на основі реальних виробничих процедур, а не лише загальних знань.

Відповідь від LLM обробляється та структурується у форматі JSON з полями: situation (оцінка ситуації), immediate_actions (негайні дії), possible_causes (можливі причини), next_steps (подальші кроки). Node-RED отримує відповідь та відображає її в інтерфейсі оператора разом з поточними параметрами обладнання (рис. 2).

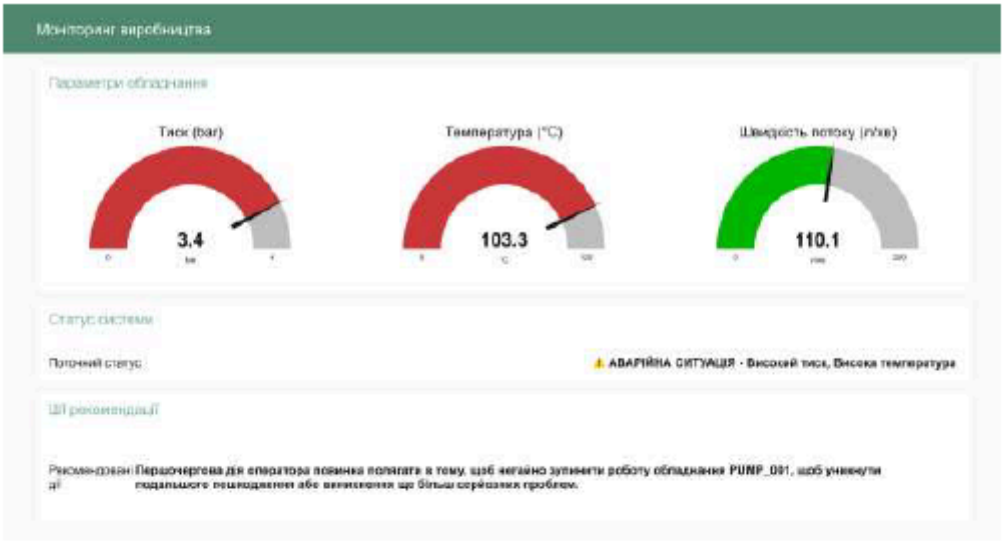

Рис. 2. Інтерфейс оператора при виявленні критичного аларму

Веб-інтерфейс Node-RED Dashboard організовано у три зони:

1. Зона моніторингу – gauge віджети для відображення поточних значень з кольоровим кодуванням (зелений – норма, жовтий – попередження, червоний – критично);

2. Зона трендів – віджети для візуалізації зміни параметрів за останні 5-10 хвилин;

3. Зона рекомендацій – віджет зі структурованими рекомендаціями від системи ШІ.

Потоки даних в Node-RED організовані за модульним принципом. Кожен параметр має окрему гілку обробки, що включає: inject node (генератор даних), function node (перевірка порогових значень), switch node (маршрутизація при алармі), HTTP request node (запит до ШІ-сервера), dashboard nodes (візуалізація). Така організація забезпечує легке налаштування системи та можливість додавання нових параметрів без зміни існуючих потоків.

Система здатна обробляти одночасне виникнення декількох алармів. Для кожного аларму формується окремий запит до ШІ-сервера, що дозволяє отримати специфічні рекомендації для кожної ситуації. Час від виявлення аларму до отримання рекомендацій становить 2-5 секунд, що є прийнятним для більшості промислових застосувань.

Технологічний стек розробленої системи включає:

– Node-RED v3.x – побудова потоків даних та веб-інтерфейсу;

– Python 3.10+ / Flask – серверна частина та REST API;

– LLM API – генерація рекомендацій (Groq/OpenAI);

– JSON – формат обміну даними між компонентами.

Переваги обраної архітектури полягають у модульній побудові, що дозволяє незалежно розвивати окремі компоненти системи. Використання стандартних протоколів (HTTP, JSON) забезпечує сумісність з різними промисловими системами. Веб-орієнтований підхід дозволяє операторам отримувати доступ до системи з будь-якого пристрою в локальній мережі.

Поточна архітектура розроблена з урахуванням майбутнього впровадження повноцінної RAG-системи. Для цього планується додати:

– модуль завантаження та обробки PDF-документації;

– векторну базу даних для зберігання embeddings (FAISS, Chroma);

– компонент семантичного пошуку релевантних фрагментів;

– інтеграцію знайденого контексту в prompt для LLM.

Така структура дозволить системі надавати рекомендації на основі всієї технічної документації підприємства з автоматичним пошуком найбільш релевантних фрагментів для кожної конкретної ситуації [8].

Експериментальна перевірка системи показала її ефективність у детекції алармів та формуванні структурованих рекомендацій. Типовий сценарій роботи оператора з системою:

1. Оператор спостерігає за параметрами в режимі реального часу;
2. При відхиленні параметра від норми змінює колір на жовтий;
3. При досягненні критичного значення стає червоним, генерується запит до ШІ;
4. Протягом 2-3 секунд з'являється структурована інструкція;
5. Оператор виконує рекомендовані дії;
6. Система продовжує моніторинг та відображає зміни параметрів.

Такий підхід мінімізує час реагування на аварійні ситуації та зменшує ймовірність помилкових дій оператора, особливо для працівників з невеликим досвідом роботи.

ВИСНОВКИ. У роботі розроблено прототип системи інтелектуальної підтримки операторів для оперативно-диспетчерського контролю виробництва. Система реалізує модульну чотирирівневу архітектуру на базі open-source технологій Node-RED та Python, що забезпечує гнучкість та можливість масштабування.

Запропонований підхід інтеграції великих мовних моделей з промисловими системами моніторингу дозволяє автоматично генерувати структуровані рекомендації для операторів на основі технічної документації. Час від виявлення аларму до отримання рекомендацій становить 2-5 секунд, що є прийнятним для практичного застосування.

Розроблена архітектура передбачає майбутнє розширення до повноцінної RAG-системи з автоматичною індексацією та семантичним пошуком у технічній документації. Це дозволить підвищити точність та релевантність рекомендацій, а також адаптувати систему під специфічні потреби конкретного підприємства.

Практична цінність роботи полягає у створенні працюючого прототипу, який демонструє можливість ефективної інтеграції технологій штучного інтелекту з промисловими системами для підвищення безпеки виробництва та зменшення впливу людського фактору.

Перспективи подальших досліджень включають експериментальну перевірку системи на реальних промислових об'єктах, розробку алгоритмів адаптивного навчання на основі зворотного зв'язку від операторів, а також впровадження повноцінної RAG-архітектури для роботи з великими обсягами технічної документації.

ЛІТЕРАТУРА

1. Wang S., Wan J., Zhang D., Li D., Zhang C. Towards smart factory for industry 4.0: a self-organized multi-agent system with big data based feedback and coordination. Computer Networks. 2016. Vol. 101. P. 158-168.

2. Brown T.B., Mann B., Ryder N., Subbiah M., Kaplan J., et al. Language Models are Few-Shot Learners. Advances in Neural Information Processing Systems. 2020. Vol. 33. P. 1877-1901.

3. Lewis P., Perez E., Piktus A., Petroni F., Karpukhin V., et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. Advances in Neural Information Processing Systems. 2020. Vol. 33. P. 9459-9474.

4. Lee J., Bagheri B., Kao H.-A. A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems. Manufacturing Letters. 2015. Vol. 3. P. 18-23.

5. Wollschlaeger M., Sauter T., Jasperneite J. The Future of Industrial Communication: Automation Networks in the Era of the Internet of Things and Industry 4.0. IEEE Industrial

Electronics Magazine. 2017. Vol. 11, Issue 1. P. 17-27.

6. Node-RED: Low-code programming for event-driven applications. URL: https://nodered.org/ (дата звернення: 15.11.2024).

7. Lewis P., Plumbley M., Mirmehdi M. The application of machine learning to SCADA systems for anomaly detection. Procedia Computer Science. 2020. Vol. 176. P. 2772-2781.

8. Gao Y., Xiong Y., Gao X., Jia K., Pan J., et al. Retrieval-Augmented Generation for Large Language Models: A Survey. arXiv preprint arXiv:2312.10997. 2023.

9. Невлюдов І.Ш. Основи наукових досліджень : підручник / І. Ш. Невлюдов, Ю. М. Олександров, А. О. Андрусевич, О. О. Чала. Prague : OKTAN PRINT, 2024. 468 с.

10. Методичні вказівки з підготовки та захисту кваліфікаційної роботи здобувачами другого (магістерського) рівня вищої освіти спеціальності 174 / Упоряд. І. Ш. Невлюдов, Р. В. Артюх, В. В. Безкоровайний, Н. П. Демська, В. В. Євсєєв, О. І. Филипенко, О. М. Цимбал. Харків: ХНУРЕ, 2024. 57 с.

11. Nevliudov, I., Yevsieiev, V., Baker, J. H., Ahmad, M. A., & Lyashenko, V. (2020). Development of a cyber design modeling declarative Language for cyber physical production systems. J. Math. Comput. Sci., 11(1), 520-542.

12. Mustafa, S.K., Yevsieiev, V., Nevliudov, I., & Lyashenko, V.. (2022). HMI Development Automation with GUI Elements for Object-Oriented Programming Languages Implementation. SSRG International Journal of Engineering Trends and Technology, 70(1), 139-145.

13. Lyashenko, V., Abu-Jassar, A. T., Yevsieiev, V., & Maksymova, S. (2023). Automated Monitoring and Visualization System in Production. International Research Journal of Multidisciplinary Technovation, 5(6), 9-18.

14. Al-Sharo, Y., Abu-Jassar, A., Lyashenko, V., Yevsieiev, V., & Maksymova, S. (2023). A Robo-hand prototype design gripping device within the framework of sustainable development. Indian Journal of Engineering, 20, e37ije1673.

15. Nevliudov, I., Yevsieiev, V., Maksymova, S., & Filippenko, I. (2020). Development of an architecturallogical model to automate the management of the process of creating complex cyberphysical industrial systems.Eastern-European Journal of Enterprise Technologies, 2020, 4(3-106), 44-52.

16. Abu-Jassar, A. T., Attar, H., Amer, A., Lyashenko, V., Yevsieiev, V., & Solyman, A. (2025). Development and investigation of vision system for a small-sized mobile humanoid robot in a smart environment. International Journal of Crowd Science, 9(1), 29-43. https://doi.org/10.26599/IJCS.2023.9100018

17. Nevliudov, I., Yevsieiev, V., Maksymova, S. & Chala, O. (2025). A Small-Sized Robot Prototype Development Using 3D Printing. Acta Mechanica et Automatica, 19(1), 2025. 77-81. https://doi.org/10.2478/ama-2025-0010

18. Bortnikova V, Yevsieiev V, Beskorovainyi V, Nevliudov I, Botsman I, Maksymova S. Structural parameters influence on a soft robotic manipulator finger bend angle simulation. In 2019 IEEE 15th International Conference on the Experience of Designing and Application of CAD Systems (CADSM), Polyana, Ukraine 2019; 35-38. doi: 10.1109/CADS M.2019.8779300

19. I. Nevliudov, V. Yevsieiev, S. Maksymova, N. Demska, K. Kolesnyk, Olha Miliutina, "Mobile Robot Navigation System Based on Ultrasonic Sensors", 2023 IEEE XXVIII International Seminar/Workshop on Direct and Inverse Problems of Electromagnetic and Acoustic Wave Theory (DIPED), vol.1, pp.247-251, 2023.

Науковий керівник: *Демська Наталія Павлівна, к.т.н., доцент кафедри КІТАР Харківського національного університету радіоелектроніки*

ДОДАТОК В

Зміст інструкції

СТОРІНКА 1: КРИТИЧНЕ ПЕРЕВИЩЕННЯ ТЕМПЕРАТУРИ

Причини виникнення:

Критичне перевищення температури може бути спричинене порушенням циркуляції робочого середовища, несправністю системи охолодження, перевантаженням обладнання або блокуванням теплообмінних поверхонь.

Послідовність дій оператора:

НЕГАЙНО зменшити навантаження системи на 30-40% шляхом зниження витрати робочого середовища або зменшення потужності нагрівальних елементів.

Перевірити роботу циркуляційних насосів та відкрити додаткові контури охолодження якщо вони доступні.

Моніторити температуру кожні 30 с. Якщо через 2 хвилини температура перевищує 106 °C - виконати аварійну зупинку системи.

Перевірити справність датчиків температури для виключення хибного спрацювання.

Викликати технічний персонал для діагностики системи охолодження.

Критичні пороги:

Попередження при T > 100 °C, критичний рівень при T > 105 °C, аварійна зупинка при T > 110 °C.

СТОРІНКА 2: АВАРІЙНЕ ПІДВИЩЕННЯ ТИСКУ

Причини виникнення:

Аварійне підвищення тиску може бути спричинене блокуванням випускних трубопроводів, несправністю запобіжної арматури, інтенсивним нагріванням робочого середовища в замкненому обсязі або несправністю

системи регулювання тиску.

Послідовність дій оператора:

НЕГАЙНО перевірити стан запобіжних клапанів та переконатись у їх справності.

Зменшити подачу енергії до системи на 40-50% для уповільнення процесу підвищення тиску.

Перевірити відсутність блокування випускних ліній та відкрити додаткові лінії скидання тиску.

При тиску понад 9 бар виконати аварійний скид через запобіжні клапани.

Евакуювати персонал з небезпечної зони при неможливості швидкого зниження тиску.

Задокументувати інцидент та виклик аварійної служби.

Критичні пороги:

Попередження при P > 7 бар, критичний рівень при P > 8 бар, аварійний скид при P > 9 бар.

СТОРІНКА 3: КРИТИЧНЕ ЗНИЖЕННЯ РІВНЯ

Причини виникнення:

Зниження рівня може відбутися через витік робочого середовища, несправність системи живлення, підвищене споживання або випаровування. Недостатній рівень створює ризик порушення циркуляції та пошкодження обладнання.

Послідовність дій оператора:

Перевірити систему живлення та переконатись у справності живильних насосів.

Провести візуальний огляд системи на наявність видимих витоків робочого середовища.

Моніторити швидкість зниження рівня для оцінки масштабу проблеми.

Запустити резервні насоси живлення для підтримання рівня.

При рівні нижче 15% виконати аварійну зупинку для запобігання пошкодженню обладнання.

Локалізувати джерело проблеми та усунути причину зниження рівня.

Критичні пороги:

Попередження при L < 30%, критичний рівень при L < 20%, аварійна зупинка при L < 15%.

СТОРІНКА 4: ВИЯВЛЕННЯ ВИТОКУ РОБОЧОГО СЕРЕДОВИЩА

Ознаки витоку:

Витік характеризується одночасним зниженням тиску та рівня заповнення. Швидкість зміни цих параметрів залежить від масштабу витоку та місця розгерметизації системи.

Послідовність дій оператора:

НЕГАЙНО локалізувати місце витоку за допомогою візуального огляду та приладів контролю.

Оцінити масштаб витоку та потенційну небезпеку для персоналу.

Ізолювати пошкоджену ділянку за допомогою запірної арматури якщо це можливо.

Зупинити подачу робочого середовища до пошкодженої зони.

Викликати аварійну службу для усунення витоку.

Евакуювати персонал з небезпечної зони якщо необхідно.

Задокументувати інцидент з зазначенням часу виявлення, місця витоку та вжитих заходів.

Запобіжні заходи:

Регулярний контроль герметичності з'єднань, моніторинг стану ущільнень та своєчасна заміна зношених елементів.

СТОРІНКА 5: НАДМІРНА ВИТРАТА РОБОЧОГО СЕРЕДОВИЩА

Причини виникнення:

Підвищена витрата може свідчити про витік через нещільності з'єднань, несправність регулюючої арматури, неправильне налаштування системи автоматичного керування або приховані дефекти обладнання.

Послідовність дій оператора:

Перевірити налаштування системи автоматичного регулювання та відкоригувати параметри якщо необхідно.

Провести діагностику регулюючої арматури для виявлення можливих несправностей.

Перевірити наявність прихованих витоків шляхом огляду всіх з'єднань та ущільнень.

Скоригувати режим роботи системи для зниження витрати до номінального значення.

Моніторити витрату протягом 10-15 хвилин для підтвердження ефективності вжитих заходів.

Викликати технічний персонал для детального налаштування системи якщо проблема не вирішена.

Економічні наслідки:

Надмірна витрата призводить до збільшення експлуатаційних витрат та може вказувати на приховані несправності обладнання. Своєчасне виявлення та усунення причин дозволяє уникнути значних економічних втрат.

ДОДАТОК Г

Демонстраційний матеріал

№ докумен-та	Позначення	Найменування	Додаткові відомості
		Текстові документи	
1	ГЮІК 464430.011 ВД	Пояснювальна записка	А4, 64 с.
		Додаткові матеріали	
2		Лістинг програми	А4, 4 с.
3		Апробація наукових	А4, 7 с.
		результатів	
		досліджень	
4		Зміст інструкції	А4, 4 с.
5		Демонстраційний матеріал	А4, 12 с.

Змін.	Арк.	Номер докум.	Підпис	Дата	ГЮІК 464430.011 ВД			
Розроб.		Рябовол Д.А.			Розроблення програмного модуля оперативно-диспетчерського контролю виробництва на базі кібер-фізичних систем керування	Літера	Аркуш	Аркушів
Перевір.		Демська Н.П.				Н	1	1
						Кафедра КІТАР ХНУРЕ		
Н.контр.		Стародубцев М.Г.						
Затв.		Невлюдов І.Ш.						