МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: **«РОЗРОБКА АВТОМАТИЗОВАНОЇ СИСТЕМИ КЕРУВАННЯ ВЕРСТАТАМИ»**

Виконала: студентка 4 курсу, групи 6.1210-2пі
спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)
освітньої програми програмна інженерія
(назва освітньої програми)
Д.Д. Мірошник
(ініціали та прізвище)

Керівник доцент кафедри програмної інженерії, доцент, к.ф.-м.н. Кудін О.В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент доцент кафедри фундаментальної та прикладної математики, доцент, к.ф.-м.н. Панасенко Є.В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра програмної інженерії

Рівень вищої освіти бакалавр

Спеціальність 121 інженерія програмного забезпечення
(шифр і назва)

Освітня програма програмна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедри програмної інженерії, к.ф.-м.н., доцент

_____ Лісняк А.О.
(підпис)

“ ____ ” _____ 2023 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТЦІ

Мірошник Діані Денисівні
(прізвище, ім’я та по-батькові)

1. Тема роботи Розробка автоматизованої системи керування верстатами

керівник роботи Кудін Олексій Володимирович, к.ф.-м.н., доцент
(прізвище, ім’я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 21 » грудня 2023 року № 2180-с

2. Строк подання студентом роботи 03.06.2024 р.

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Постановка задачі.
2. Основні теоретичні відомості.
3. Проєктування і розробка системи керування верстатами.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)
презентація за темою доповіді

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 25.12.2023 р.

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	09.01.2024	
2.	Збір вихідних даних.	22.01.2024	
3.	Обробка методичних та теоретичних	19.02.2024	
	джерел.		
4.	Розробка першого та другого розділу.	15.04.2024	
5.	Розробка третього розділу.	20.05.2024	
6.	Оформлення та нормоконтроль	27.05.2024	
	кваліфікаційної роботи бакалавра.		
7.	Захист кваліфікаційної роботи.	20.06.2024	

Студент _____ Д.Д. Мірошник
(підпис) (ініціали та прізвище)

Керівник роботи _____ О.В. Кудін
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ А.В. Столярова
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота бакалавра «Розробка автоматизованої системи керування верстатами»: 43 с., 18 рис., 23 джерела.

3D-ДРУК, 3D-ПРИНТЕР, ВЕРСТАТ, ЛАЗЕРНИЙ ВЕРСТАТ, МІКРОКОНТРОЛЕР, СИСТЕМА КЕРУВАННЯ ВЕРСТАТАМИ, ЧИСЛОВЕ ПРОГРАМНЕ КЕРУВАННЯ, G-CODE, STL.

Об’єкт дослідження – процес керування верстатами з числовим програмним керуванням (ЧПК).

Предмет дослідження – верстати з числовим програмним керуванням, мова програмування G-Code, 3D-друк.

Мета роботи: спроєктувати та розробити застосунок для генерації керуючого G-коду для 3D-друку з використанням фреймворків Streamlit та STL to G-Code Slicer API.

Метод дослідження – методи збору та аналізу вимог до програмного забезпечення, методи моделювання, проєктування та консультування програмного забезпечення.

При розробці застосунку було проведено аналіз актуальності предметної області, огляд основних принципів роботи верстатів з ЧПК, а також порівняльний аналіз сучасних систем керування верстатами. Розроблені основні функціональні та нефункціональні вимоги до системи та проілюстровано діаграмою прецедентів.

Реалізовано вебзастосунок за допомогою фреймворків Streamlit та STL to G-Code Slicer API. Розроблені фізичні моделі для тестування роботи застосунку.

SUMMARY

Bachelor’s qualifying paper “Development of an Automated Machine Control System”: 43 pages, 18 figures, 23 references.

3D PRINTING, 3D PRINTER, MACHINE TOOL, LASER MACHINE, MICROCONTROLLER, MACHINE CONTROL SYSTEM, COMPUTER NUMERICAL CONTROL, G-CODE, STL.

The object of the study is the process of controlling numerically controlled machines (CNC).

The subjects of the study are CNC machines, G-code programming language and 3D printing.

The aim of the study is to design and develop an application for generating G-code for 3D printing using the Streamlit and STL to G-code Slicer API frameworks.

The methods of research are methods of collecting and analyzing software requirements, modeling methods, software design and consulting.

An analysis of the subject’s area relevance, a review of the basic principles of CNC machine operation and a comparative analysis of modern machine control systems were conducted during the application’s development. Basic functional and non-functional requirements for the system were developed and illustrated with the use case diagram.

A web application was implemented using the Streamlit and STL to G-Code Slicer API frameworks. Physical models were developed to test the application`s functionality.

ЗМІСТ

ВСТУП

Сучасне промислове виробництво неможливе без застосування високотехнологічного обладнання, яке дає змогу виконувати складні завдання з високою точністю та ефективністю. Верстати з числовим програмним керуванням (ЧПК) дозволяють автоматизувати процеси обробки матеріалів та підвищити якість роботи, водночас зменшуючи витрати часу та ресурсів. Обробка таких матеріалів, як, наприклад, дерево, скло, метал, пластмаса, папір значно полегшується з використанням верстатів з ЧПК.

Основна перевага у впровадженні верстатів з ЧПК у виробництво полягає у збільшенні можливостей виконання складних операцій з високою точністю та повторюваністю. Особливо важливий високий рівень точності в умовах серійного виробництва, оскільки дозволяє виготовляти деталі з мінімальним відхиленням від заданих або стандартних параметрів. Крім цього, автоматизація виробництва знижує вплив людського чинника, мінімізує кількість помилок, таким чином підвищуючи ефективність використання виробничих потужностей.

Однак, незважаючи на переваги, які надають верстати з ЧПК, існують складнощі, пов’язані з їхнім керуванням. Основна проблема полягає в складності налаштування та програмування верстатів. Процес підготовки та оптимізації керівної програми вимагає значних витрат часу та людських ресурсів. До того ж існує проблема в підготовці кваліфікованих спеціалістів, здатних налаштовувати, обслуговувати та програмувати верстати з ЧПК, оскільки це вимагає спеціалізованих знань та навичок.

Інша важлива проблема – інтеграція верстатів в загальну виробничу систему на підприємстві. Під інтеграцією в загальну систему мається на увазі можливість взаємодії з різними видами верстатів, інтеграція з системами планування та управління виробництвом, системами контролю якості та іншими інформаційними системами. Ефективна інтеграція може скоротити

час простоїв обладнання та зменшити витрати матеріалів, і у такий спосіб підвищити продуктивність виробництва.

Метою даної кваліфікаційної роботи є проєктування та розробка автоматизованої середи керування верстатами з ЧПК.

Завдання дослідження:

– аналіз сучасних систем керування верстатами з ЧПК;
– визначення вимог до програмного забезпечення;
– розробка архітектури системи;
– програмування та тестування прототипу системи;
– оцінювання ефективності роботи програмного забезпечення.

Об’єктом дослідження є процес керування верстатами з ЧПК на виробництві.

Предметом дослідження є автоматизована середа керування, що використовується для управління верстатами з ЧПК.

Структурно робота складається з вступу, трьох розділів, висновку та додатку. В першому розділі наведений огляд предметної області та аналіз наявних систем. В другому розділі визначені основні функціональні та нефункціональні вимоги, розроблено діаграму прецедентів та наведено детальний опис прецедентів, наведено основи мови програмування G-code. В третьому розділі наведено опис розробленого застосунку та його тестування.

1 ОСНОВНІ ТЕОРЕТИЧНІ ВІДОМОСТІ

1.1 Загальний огляд верстатів з ЧПК

Верстати з числовим програмним керуванням (ЧПК) – обладнання, яке використовує програмне забезпечення для управління інструментами [1]. Це дозволяє виконувати складні операції з високою точністю та повторюваністю. Верстати з ЧПК застосовуються в багатьох галузях промисловості для обробки різних матеріалів, таких як метал, дерево, пластик, акрил та багато інших.

Основний принцип роботи верстатів з ЧПК полягає в автоматизованому виконанні заздалегідь написаних програм, які задають послідовність дій для обробки матеріалів. Ці програми пишуться мовою G-коду та містять інструкції щодо подачі інструменту, швидкості переміщення й обертання інструменту та інших параметрів. Завдяки цьому верстати з ЧПК можуть виконувати операції, які важко або навіть неможливо реалізувати використовуючи ручні інструменти [2].

Верстати з ЧПК мають широке застосування в сучасній промисловості. Вони використовуються в машинобудуванні для виготовлення деталей для автомобілів, літаків, залізничного транспорту та іншої техніки. У медицині вони застосовуються для виготовлення протезів, імплантатів та медичних інструментів. В аерокосмічній промисловості верстати використовуються для виробництва складних деталей з високими вимогами до точності і якості. Крім того, верстати з ЧПК широко використовуються в електронній промисловості для виготовлення друкованих плат, корпусів та інших компонентів. У меблевій промисловості використовуються для різання дерев’яних фігурних деталей. У ювелірній справі – для виготовлення складних виробів з металу та дорогоцінного каміння. Також варто зазначити, що використання верстатів з ЧПК у виробництві має велике значення для малого бізнесу з декількох причин:

– підвищення продуктивності;
– збільшення «гнучкості» виробництва;
– зниження витрат;
– покращення якості роботи;
– конкурентоспроможність.

Нижче розглянемо зазначене докладніше.

Одна з найбільш вагомих причин – це підвищення продуктивності та завдяки цьому збільшення обсягів виробництва без потреби у збільшенні робітничої сили, оскільки обслуговуванням та налаштуванням верстатів може займатись один спеціаліст, що особливо важливо для малих виробництв.

Малий бізнес часто стикається з виробництвом невеликих партій продукції, або виготовлення продукції на замовлення, коли кількість може вимірюватись одним виробом. Використання верстатів з ЧПК у виробництві може дозволити малим підприємствам швидко та ефективно реагувати на зміни в попиті та швидко перелаштовувати обладнання для виготовлення нових видів продукції.

Автоматизація обладнання допомагає знизити виробничі витрати шляхом зменшення відходів матеріалів та помилок у виробництві, які можуть виникати при ручній обробці.

Використання верстатів з ЧПК у виготовленні складних, фігурних або багатокомпонентних деталей з високими вимогами до точності забезпечує високий рівень якості обробки та водночас короткі терміни виготовлення продукції, що також позитивно впливає на конкурентоспроможність малого підприємства.

Верстати з ЧПК класифікуються за кількома ознаками, включно з типом обробки, кількістю осей та конструктивними особливостями. За типом обробки верстати поділяються на фрезерні, токарні, свердлильні, шліфувальні та інші. За кількістю осей вони можуть бути трьох-, п’яти- або багато координатними. Кількість керованих осей визначає, скільки ступенів свободи має інструмент під час обробки. Стандартні верстати з ЧПК мають три осі (X,

Y, Z), але існують також верстати з п’ятьма і більше осями, що дозволяє виконувати більш складні операції. За конструктивними особливостями верстати можуть бути вертикальними та горизонтальними, з фіксованою або рухомою робочою платформою. Розмір робочої зони визначає максимальні габарити деталі, яку можна обробити на верстаті.

Основні характеристики верстатів з ЧПК включають точність позиціювання, швидкість обробки, потужність шпинделя, кількість керованих осей та розмір робочої зони. Точність позиціювання визначає, наскільки точно верстат може переміщати інструмент по заданих координатах. Швидкість обробки залежить від потужності шпинделя та швидкості подачі інструменту [3].

Отже, верстати з ЧПК є невіддільною частиною сучасного виробництва, забезпечуючи високу точність, продуктивність та автоматизацію процесів обробки матеріалів. Вони широко застосовуються в різних галузях промисловості, маючи значні переваги та деякі недоліки. Завдяки різноманітності характеристик і можливостей, верстати з ЧПК дозволяють обробляти широкий спектр матеріалів, що робить їх універсальними інструментами для виготовлення деталей і виробів.

1.2 Принципи роботи верстатів з ЧПК

Процес роботи верстатів з числовим програмним керуванням базується на використанні мікропроцесорних систем для автоматизованого управління механізмом обробки матеріалів. Основний принцип роботи полягає у взаємодії між керуючою програмою та безпосередньо виконавчим механізмом верстата.

Керуюча програма, як правило написана мовою G-code, містить в собі інструкції для верстата, які визначають траєкторію переміщення інструменту по робочій зоні для виготовлення необхідної деталі, швидкість переміщення інструменту та інші параметри.

Процес роботи над обробкою матеріалу для виготовлення необхідної деталі чи виробу починається із завантаження керуючої програми в систему ЧПК.

Програма розбивається на окремі компоненти, які обробляє мікроконтролер та надсилає на виконання верстату.

Мікроконтролер – центральний елемент системи та головний обчислювальний пристрій. Відповідає за обробку вхідних даних, тобто приймає команди з керуючої програми, перетворення команд в електричні сигнали для двигунів і інших виконавчих механізмів та координацію роботи підсистеми, тобто синхронізацію рухів різних осей верстата для забезпечення точності обробки.

Мікроконтролер у системах ЧПК забезпечує:

– обробку команд G-коду, тобто перетворює команди в електричні сигнали, які керують рухом двигунів;
– координацію рухів осей, тобто синхронізує переміщення осей верстата для точного позиціювання інструменту;
– контроль за параметрами обробки: швидкість руху, температура охолоджувальної рідини, подачу повітря, та інші параметри;
– обробку зворотного зв’язку, тобто обробка сигналів з датчиків для забезпечення стабільної роботи [4].

Одним із популярних мікроконтролерів, які використовуються у верстатах з ЧПК є Arduino UNO. Arduino UNO – це мікроконтролер на базі ATmega328P. Він має 14 цифрових входів/виходів, 6 з яких можуть використовуватись як ШІМ-виходи (виходи широтно-імпульсної модуляції), та 6 аналогових входів. Також має 16 МГц кварцовий генератор, ICSP-роз’єм для програмування, USB-з’єднання для програмування та обміну даними, роз’єм живлення (від батареї або адаптера змінного струму) та кнопка перезавантаження [5].

Зовнішній вигляд мікроконтролера Arduino Uno можна переглянути на рисунку 1.1.

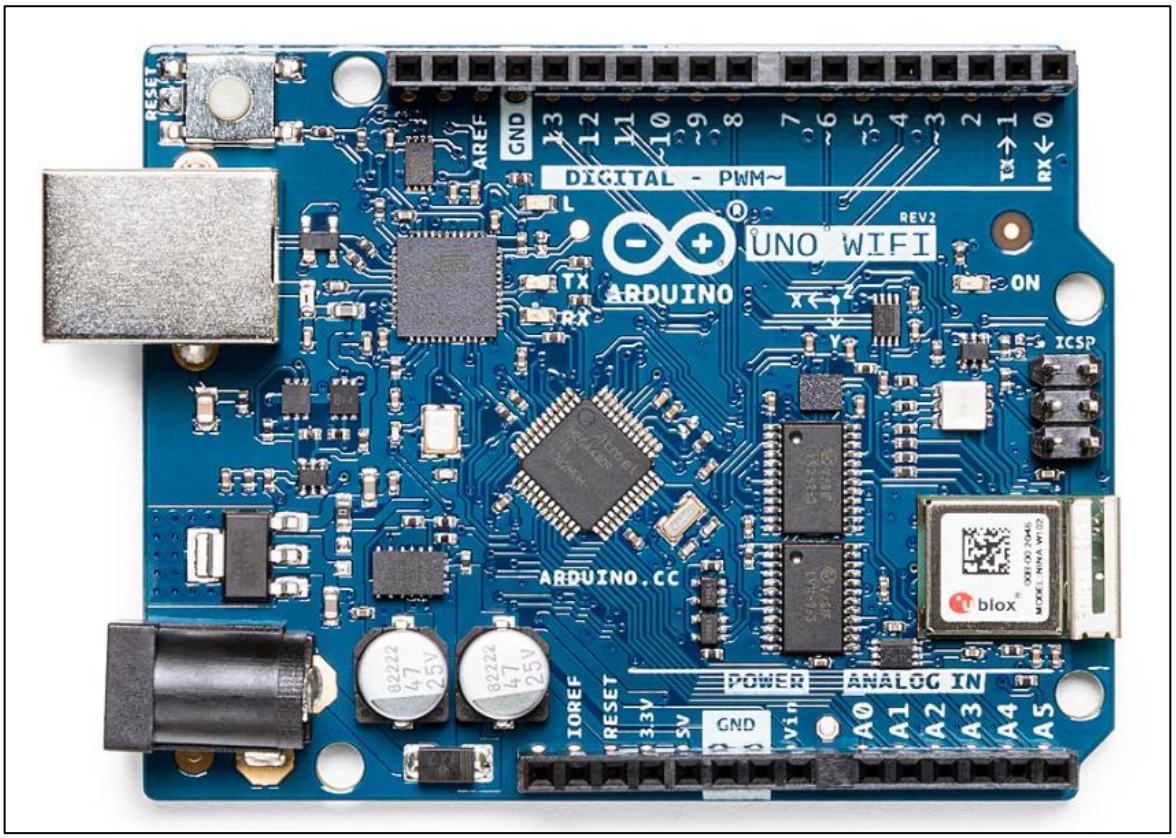

Рисунок 1.1 – Плата Arduino Uno WiFi REV2 [6]

Для використання Arduino Uno у верстатах з ЧПК часто застосовується плата розширення CNC Shield.

CNC Shield дає змогу підключення драйверів кінцевих вимикачів, крокових двигунів та інших периферійних пристроїв, що забезпечують коректну роботу верстата. За допомогою плати розширення відбувається одночасне керування декількома кроковими двигунами.

CNC Shield дає можливість реалізувати наступні функції:

– керування кроковими двигунами, що забезпечує точне позиціювання інструменту;
– підключення кінцевих вимикачів, які значно підвищують рівень безпеки роботи верстата завдяки контролю кінцевих положень осей;
– регулювання швидкості та прискорення (налаштування параметрів руху інструменту для різних видів обробки).

На рисунку 1.2 зображена плата розширення CNC Shield, встановлена на мікроконтролер Arduino Uno з підключеними кроковими двигунами.

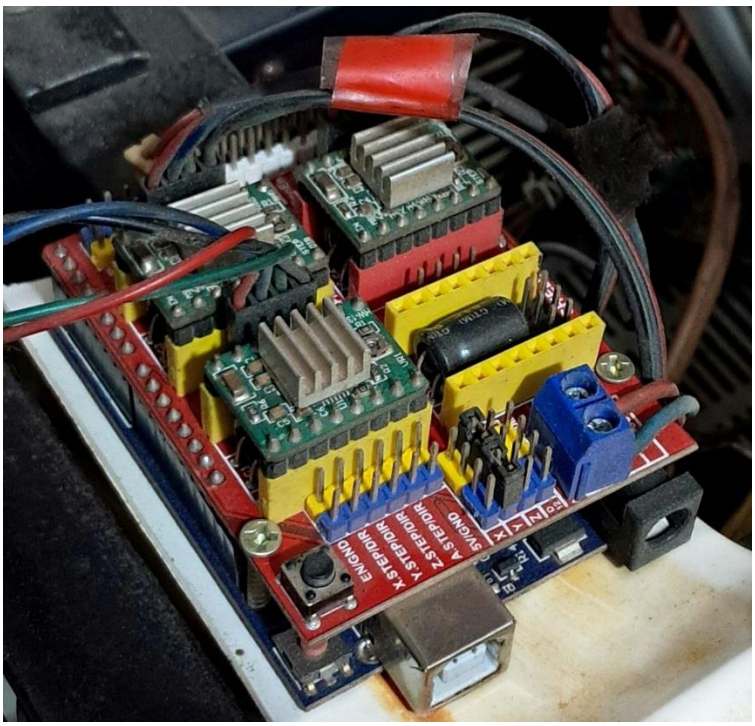

Рисунок 1.2 – Плата розширення CNC Shield

Розглянемо докладніше принцип роботи на прикладі лазерного CO_2 верстата «L-Laser L1490».

Лазерний CO_2 верстат є прикладом сучасного верстата з ЧПК, який використовує лазерний промінь для обробки матеріалів. Основний принцип роботи полягає у генерації та фокусуванні лазерного променя, який спрямовується на оброблюваний матеріал для його різання, гравіювання або маркування. Лазерний промінь генерується в газовій лазерній трубці, що заповнена сумішшю вуглекислого газу, азоту та гелію. Електричний розряд у трубці викликає збудження молекул газу, що призводить до емісії лазерного випромінювання. Це випромінювання концентрується за допомогою системи дзеркал та лінз, формуючи висококонцентрований промінь, який спрямовується на оброблюваний матеріал.

Управління рухом лазерного променя здійснюється за допомогою системи ЧПК. Керуюча програма визначає траєкторію руху каретки, на якій встановлена лазерна головка, регулює потужність променя та швидкість руху лазерної головки залежно від властивостей оброблюваного матеріалу та вимог до вихідної продукції. Наприклад, при різанні матеріалів з великою товщиною

в керуючій програмі встановлюються такі параметри обробки, як висока потужність та низька швидкість руху по траєкторії, а при гравіюванні тонких деталей навпаки, потрібна низька потужність лазерного променя та висока швидкість руху.

Лазерний CO_2 верстат «L-Laser L1490» зображено на рисунку 1.3.

Рисунок 1.3 – L-Laser L1490 [7]

Далі розглянемо основні етапи роботи лазерного CO_2 верстата.

Першим кроком є підготовка керуючої програми, яка створюється з використанням G-коду та CAD/CAM програмного забезпечення. Програма містить налаштування потужності лазерної трубки та швидкості переміщення робочої головки та деякі додаткові параметри, як, наприклад, інтервал переміщення по осі Y для гравіювання, подачу повітря та інше.

Готова програма завантажується в пам’ять мікроконтролера, встановленого на верстаті, наприклад, Arduino Uno, характеристики та функції якого було розглянуто вище.

Після запуску програми мікроконтролер подає сигнали на керування кроковими двигунами, які переміщують лазерну головку по осях X, Y, та, за наявності, осі Z.

Лазерний промінь генерується та фокусується на поверхні матеріалу. Потужність променя регулюється мікроконтролером відповідно до налаштувань, заданих у керуючій програмі, у такий спосіб забезпечуючи різання чи гравіювання матеріалу [8].

Під час роботи мікроконтролер безперервно моніторить стан системи, використовуючи підключені датчики. У разі виникнення помилок, аварійного стану верстата чи порушення виконання програми мікроконтролер зупиняє роботу верстата для запобігання критичних ситуацій та видає оператору повідомлення про зупинку та помилку в роботі.

1.3 Огляд подібних систем

Одним із найважливіших аспектів у роботі верстата з ЧПК є програмне забезпечення. У сучасному виробництві важлива ефективність керування, тобто середа керування повинна відповідати таким основним вимогам, як швидкодія, простота у налаштуванні і водночас багатофункціональність, інтуїтивно зрозумілий інтерфейс для зниження витрат часу і ресурсів на навчання операторів та стабільність роботи ПЗ. На сьогодні існує доволі багато програмного забезпечення для керування верстатами з ЧПК, кожне з яких має свої характеристики, переваги та недоліки. У цьому розділі проведемо порівняльний аналіз шести популярних програмних продуктів: Mach3, VCarve Pro, Siemens NX, LinuxCNC, LightBurn Software та Cura.

Розглянемо кожен з продуктів докладно.

VCarve Pro, розробка компанії Vectric [9] – потужний інструмент для керування верстатами з ЧПК, орієнтованих на різання, гравіювання та створення 3D моделей.

Перевагами є простота у використанні, потужні інструменти для створення 3D моделей з різними можливостями налаштування та широка вбудована бібліотека шаблонів і інструментів, яка значно полегшує роботу.

Однак є декілька недоліків – це висока вартість програмного забезпечення, що може бути критичним для малих виробництв та обмежена сумісність (програма підтримує обмежену кількість контролерів та верстатів).

Інтерфейс програмного забезпечення VCarve Pro зображено на рисунку 1.4.

Рисунок 1.4 – Інтерфейс VCarvePro [10]

Siemens NX [11] – один з найпотужніших програмних пакетів для проєктування і керування виробничими процесами. Це комплексне рішення для CAD, CAM і CAE. Основні переваги:

– висока функціональність (можливості моделювання, аналізу та керування верстатами в одному пакеті);
– інтеграція з іншими системами (Siemens NX підтримує легку інтеграцію з ERP, PLM та іншими системами підприємства);
– потужна підтримка від компанії.

Серед недоліків програмного забезпечення можна виділити дуже високу вартість, яка може стати перешкодою для багатьох підприємств та складність у використанні, внаслідок цього підвищені вимоги до навчання персоналу. Інтерфейс Siemens NX зображено на рисунку 1.5.

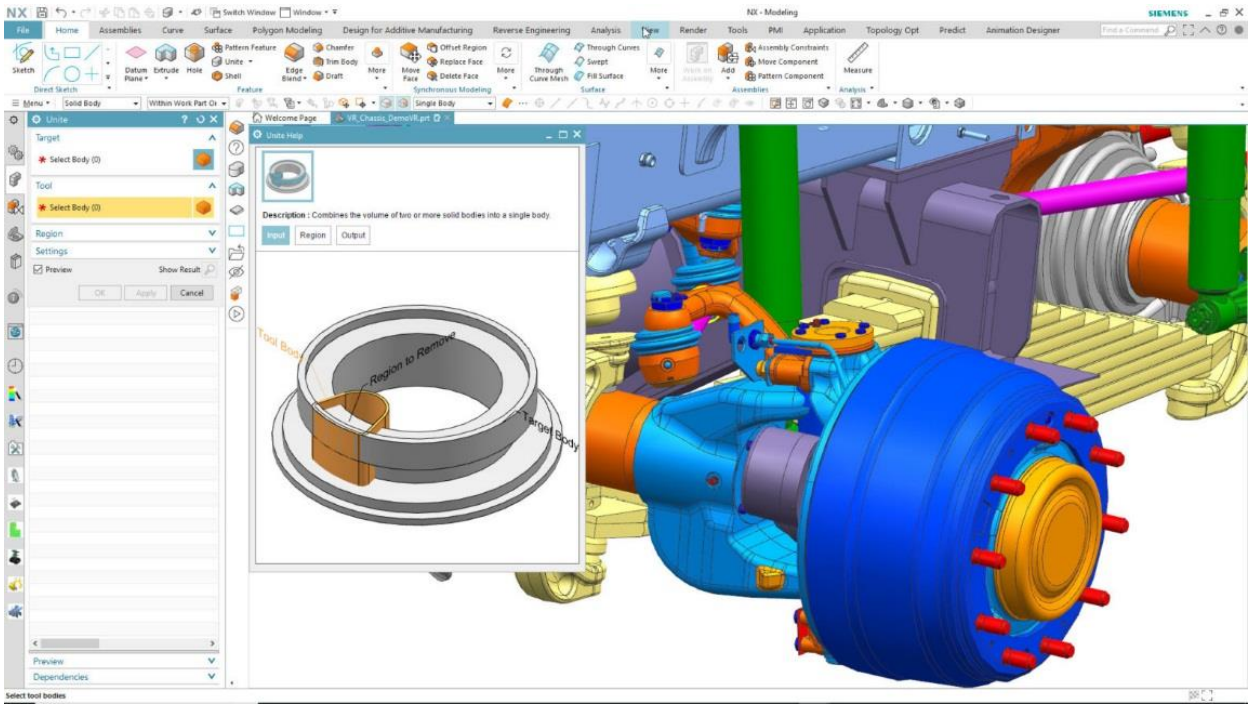

Рисунок 1.5 – Інтерфейс Siemens NX [12]

LinuxCNC [13] – програмне забезпечення з відкритими кодом для керування ЧПК верстатами, яке працює на операційній системі Linux. Це надійне і гнучке рішення для широкого спектра застосувань.

Основні переваги Linux CNC – це гнучкість і масштабованість, можливість налаштування для будь-яких типів верстатів, відсутність плати за ліцензію та користування (програмне забезпечення безкоштовне та має відкритий вихідний код) та велика спільнота користувачів і розробників, що забезпечує підтримку і регулярні оновлення.

Недоліками є складність налаштування та обмежена підтримка. LinuxCNC вимагає значних технічних знань для налаштування і використання. Також варто зазначити, що відсутність офіційної підтримки може бути проблемою для користувачів, які потребують професійної допомоги.

Mach3 [14] – один з найпоширеніших продуктів для керування верстатами з ЧПК. Розроблене компанією Newfangled Solutions, це програмне забезпечення надає високу гнучкість і функціональність для різних типів верстатів.

Інтерфейс програми зображено на рисунку 1.6.

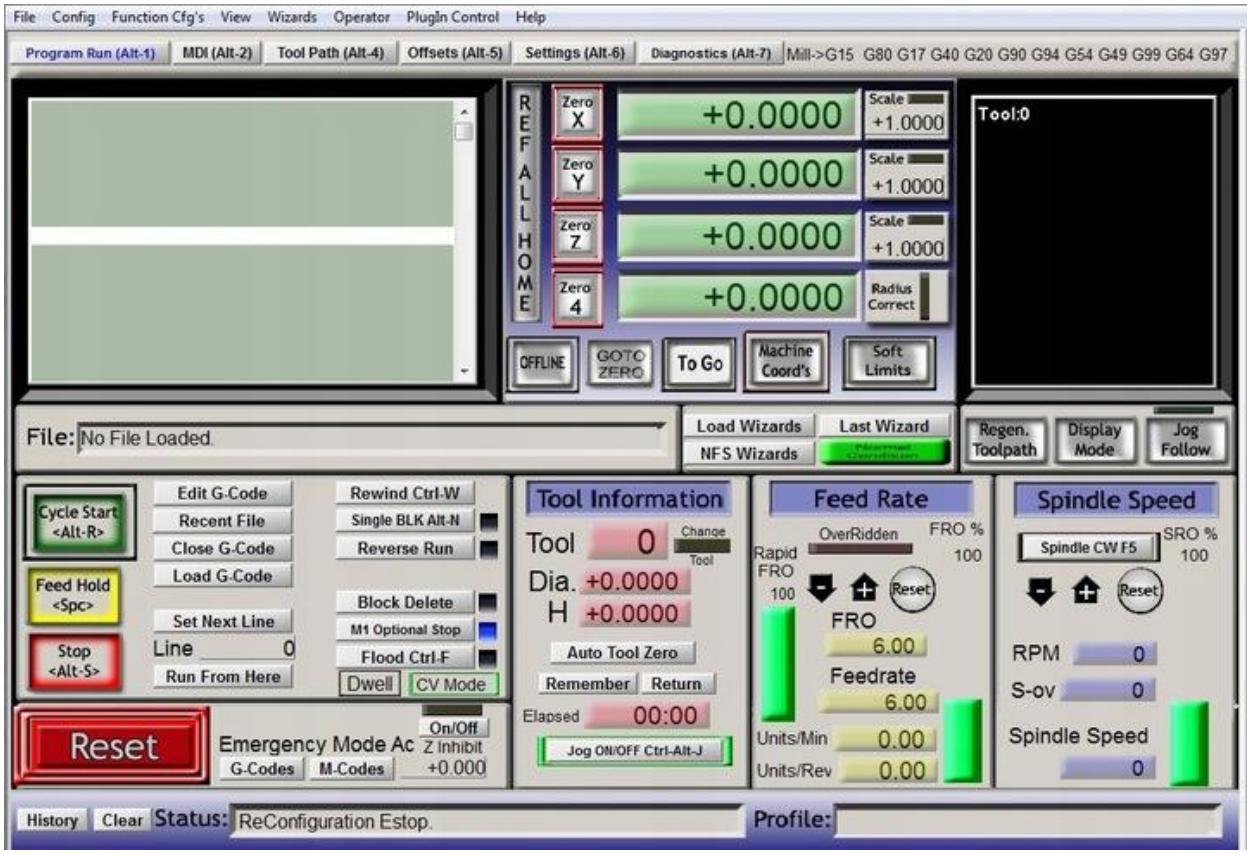

Рисунок 1.6 – Інтерфейс Mach3 [15]

Серед основних переваг можна виділити легкість налаштування та широку сумісність з різними видами обладнання та контролерами. Інтуїтивно зрозумілий інтерфейс дозволяє користувачам швидко налаштовувати систему під свої потреби, а підтримка макросів та скриптів дозволяє користувачам налаштовувати програму для специфічних завдань.

Однак, незважаючи на простоту розуміння інтерфейсу, налаштування може бути складним для тих, хто не має досвіду роботи з ЧПК. Крім цього, програмне забезпечення є платним, що може бути перешкодою для малих підприємств.

LightBurn Software [16] – програмне забезпечення, розроблене для управління лазерними ЧПК-верстатами. LightBurn забезпечує повний цикл від проєктування до обробки завдання. Серед основних функцій можна виокремити наявність інструментів для створення та редагування векторної графіки, підтримку різних форматів файлів, можливість налаштування параметрів лазерної різки та гравіювання, інтеграцію з популярними моделями лазерних верстатів, попередній перегляд обробки завдання.

Інтуїтивний інтерфейс, легкість у використанні, спеціалізовані інструменти для лазерної обробки та доступна ціна – основні переваги LightBurn. Однак, обмежена підтримка для інших видів верстатів з ЧПК та доволі обмежений функціонал в порівнянні з іншими CAM-системами є недоліками даного програмного забезпечення.

Інтерфейс програмного забезпечення наведений на рисунку 1.7.

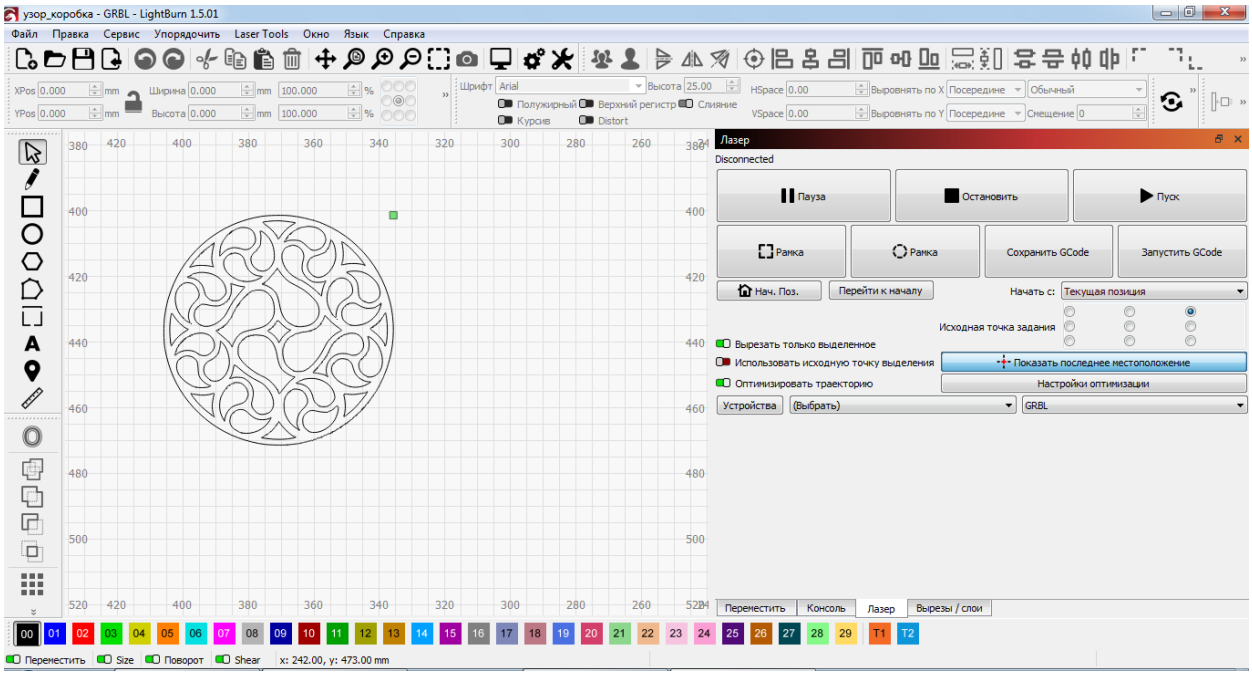

Рисунок 1.7 – Інтерфейс Lightburn Software

Cura, розробка компанії Ultimaker [17] – програмне забезпечення для перетворення 3D-моделей у форматі STL, OBJ або 3MF у G-код, який може бути виконаний 3D-принтером. В Cura можна налаштувати різні параметри друку, такі як товщина шару, швидкість друку, температура екструдера,

підтримка структури, заповнення тощо. Основними перевагами є зручний та інтуїтивно зрозумілий інтерфейс, кросплатформність, підтримка багатьох моделей 3D-принтерів, можливість збереження та експорту профілів друку для майбутнього користування, можливість встановлення додаткових плагінів (наприклад, плагін інтеграції з SolidWorks, Siemens NX, Autodesk Inventor та інші) та відсутність плати за використання програмного забезпечення.

Інтерфейс програмного забезпечення UltiMaker Cura можна переглянути на рисунку 1.8.

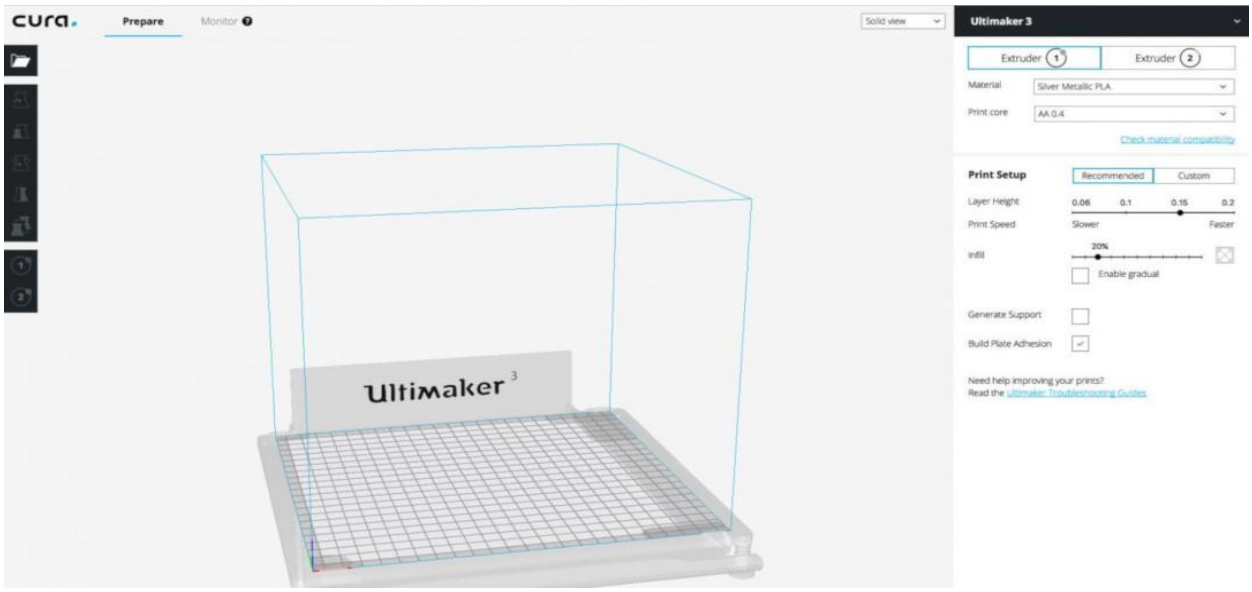

Рисунок 1.8 – Інтерфейс Cura [18]

Серед недоліків можна виокремити потребу в ручному коригуванні налаштувань друку в деяких випадках та повільну швидкість роботи на слабких комп’ютерах.

Отже, кожне розглянуте програмне забезпечення має своє унікальні характеристики та переваги. Вибір конкретного рішення залежить від специфіки виробничих завдань, вимог до продуктивності, виду верстата, а також від фінансових можливостей виробництва.

VCarve Pro та Siemens NX пропонують потужні інструменти для складних операцій, але мають високу вартість та потребують спеціальних навичок для роботи з програмним забезпеченням.

Mach3 – збалансований вибір для керування верстатами з ЧПК на середніх та малих підприємствах, оскільки має великий набір налаштувань та в цей час прийнятну вартість.

Водночас відкриті системи, як LinuxCNC, забезпечують гнучкість та можливість налаштування, що може бути важливим для малих підприємств.

LightBurn Software – оптимальний вибір для лазерної обробки, оскільки збалансоване за функціональністю та вартістю.

Cura – найпопулярніше програмне забезпечення для підготовки моделей для 3D-друку, пропонує широкий набір функцій для налаштування параметрів друку та оптимізації моделей.

В даному розділі був проведений аналіз актуальності роботи, визначені основні характеристики та принципи роботи верстатів з ЧПК та проведений огляд і порівняльний аналіз подібних систем.

2 АНАЛІЗ ТА ПРОЄКТУВАННЯ СИСТЕМИ

2.1 Аналіз та визначення вимог до системи

Програмне забезпечення, яке автоматизує генерування G-code для ЧПК верстатів та 3D принтерів, повинне відповідати декільком важливим вимогам, щоб бути ефективним і зручним у використанні. Серед них, є вимоги до інтерфейсу користувача, підтримки певних типів файлів, обробка G-code, безпека тощо. Уточнення цих вимог допоможе визначити, що програмне забезпечення буде не тільки потужним інструментом для роботи з ЧПК та 3D друком, але й зручним і доступним для користувачів різного рівня підготовки. Розглянемо більш детально функціональні та нефункціональні вимоги.

2.1.1 Функціональні вимоги

В даному розділі наведемо перелік основних функціональних вимог, яким має задовольняти створене програмне забезпечення. Розглянемо докладно кожну з функціональних вимог.

Одна з найголовніших вимог до створюваної системи – це керування верстатами, а саме:

- забезпечення можливості управління різними типами верстатів, основні з яких фрезерні, верстати лазерної різки та гравіювання та верстати 3D-друку;
- можливість встановлення та зміни параметрів роботи через інтерфейс (наприклад, швидкість руху робочого інструменту, потужність лазерного променя, температура 3D-друку, параметри системи охолодження верстата та інші);
- можливість роботи програмного забезпечення одночасно з декількома верстатами.

Забезпечення моніторингу та діагностики роботи верстата також одна з основних вимог:

– середа керування повинна забезпечувати моніторинг стану верстатів (температура, споживана потужність, швидкість руху інструменту тощо);
– коректне надання повідомлень про відхилення від стандартних параметрів роботи;
– можливість зупинки роботи верстата у разі виникнення критичної ситуації, яка може призвести до виходу обладнання з ладу.

Вимога до моніторингу кількості робочого часу та створення звітності – складова інтеграції середи керування в загальну виробничу систему, а саме:

– обчислення планової кількості часу виконання завдання та збереження реальної кількості витраченого часу на обробку задачі;
– збір та аналіз даних про продуктивність, тривалість роботи, витрат матеріалів тощо.

Можливість інтеграції з системами управління виробництвом (ERP), системами планування ресурсів підприємства (MRP) та іншими.

Також важливими вимогами до програмного забезпечення є можливість проєктування та моделювання завдань та підтримка автоматизованої підготовки завдань. В ці вимоги входить:

– вбудована CAD/CAM система для проєктування засобами графічного інтерфейсу;
– моделювання процесу виконання завдання (мається на увазі емуляція руху інструменту для створення виробу);
– можливість перетворення задач у сумісний для роботи верстатів формат (наприклад, перетворення растрових зображень у векторний формат) засобами штучного інтелекту.

Остання основна вимога – підтримка та моніторинг допоміжних систем. Під цим мається на увазі отримання даних про стан верстата впродовж роботи від системи охолодження, вентиляції, витяжки тощо.

2.1.2 Нефункціональні вимоги

В цьому розділі докладно розглянемо нефункціональні вимоги до системи.

Продуктивність роботи системи. Система повинна забезпечувати високу швидкість обробки даних та реагування на команди користувача. Час від натискання кнопки в програмі до виконання верстатом повинен бути мінімальним та складати декілька мілісекунд.

Безпека та збереження даних. Розглянемо цю вимогу докладніше:

- гарантія безперебійної роботи системи або коректної зупинки чи завершення роботи системи навіть у випадку відмови окремих компонентів;
- автоматичне резервне копіювання на випадок виходу обладнання з ладу або екстреного завершення роботи програми;
- збереження резервних копій в хмарних сховищах;
- забезпечення захисту від несанкціонованого доступу до системи та її даних.

Масштабованість. Підтримка розширення у кількості підключених верстатів та в обсягах оброблюваних даних. Наприклад, повинна бути можливість додавання нових верстатів без значного впливу на продуктивність та швидкість відгуку програми.

Надійність. Вимога містить забезпечення високої доступності системи, мінімізацію часу простою та гарантію безвідмовної роботи.

Одна з важливих нефункціональних вимог – *зручність використання*. Наведемо більш детальний опис:

- інтерфейс користувача повинен бути інтуїтивно зрозумілим і зручним для операторів;
- система повинна підтримувати декілька іноземних мов;
- наявність легко зрозумілих інструкцій користування системою у довідкових матеріалах.

Підтримка та оновлення. Уточнимо цю вимогу:

– доступність документації;
– потужна та швидка підтримка, яка забезпечує швидке виправлення помилок та швидку відповідь на запитання користувачів системи;
– оновлення програмного забезпечення без зупинки виробничого процесу.

Кросплатформність. Важливою вимогою до розроблюваного програмного забезпечення є підтримка системи на різних операційних системах, як Windows, Linux та Mac OS. До цього варто додати підтримку вебверсії застосунку, оскільки на цей час існує дуже обмежена кількість систем керування верстатами, які мають вебверсію.

2.2 Діаграма прецедентів

Для візуалізації функціональних вимог та пояснення роботи системи було розроблено діаграму прецедентів.

Діаграма прецедентів (use case diagram) – це графічне подання взаємодії між користувачами (акторами) та системою через різні функції або процеси, що виконуються системою. Основним компонентом діаграми прецедентів є «прецеденти», або «випадки використання», які описують окремі функціональні можливості системи. Актори представляють користувачів або інші системи, що взаємодіють із розроблюваною системою. Актори можуть бути людьми, програмами або обладнанням, що взаємодіють із системою ззовні [19].

Розроблена діаграма прецедентів зображена на рисунку 2.1.

Детальний опис прецедентів розглянемо далі.

Прецедент «Управління верстатом».

Актори: оператор.

Основний потік подій: оператор підключає верстат до обладнання, на

якому встановлено програмне забезпечення за допомогою USB-з’єднання або за допомогою бездротової технології. Встановлює початкові налаштування верстата. Обирає файл для виконання, перевіряє налаштування параметрів роботи верстата над цим файлом та запускає файл в роботу.

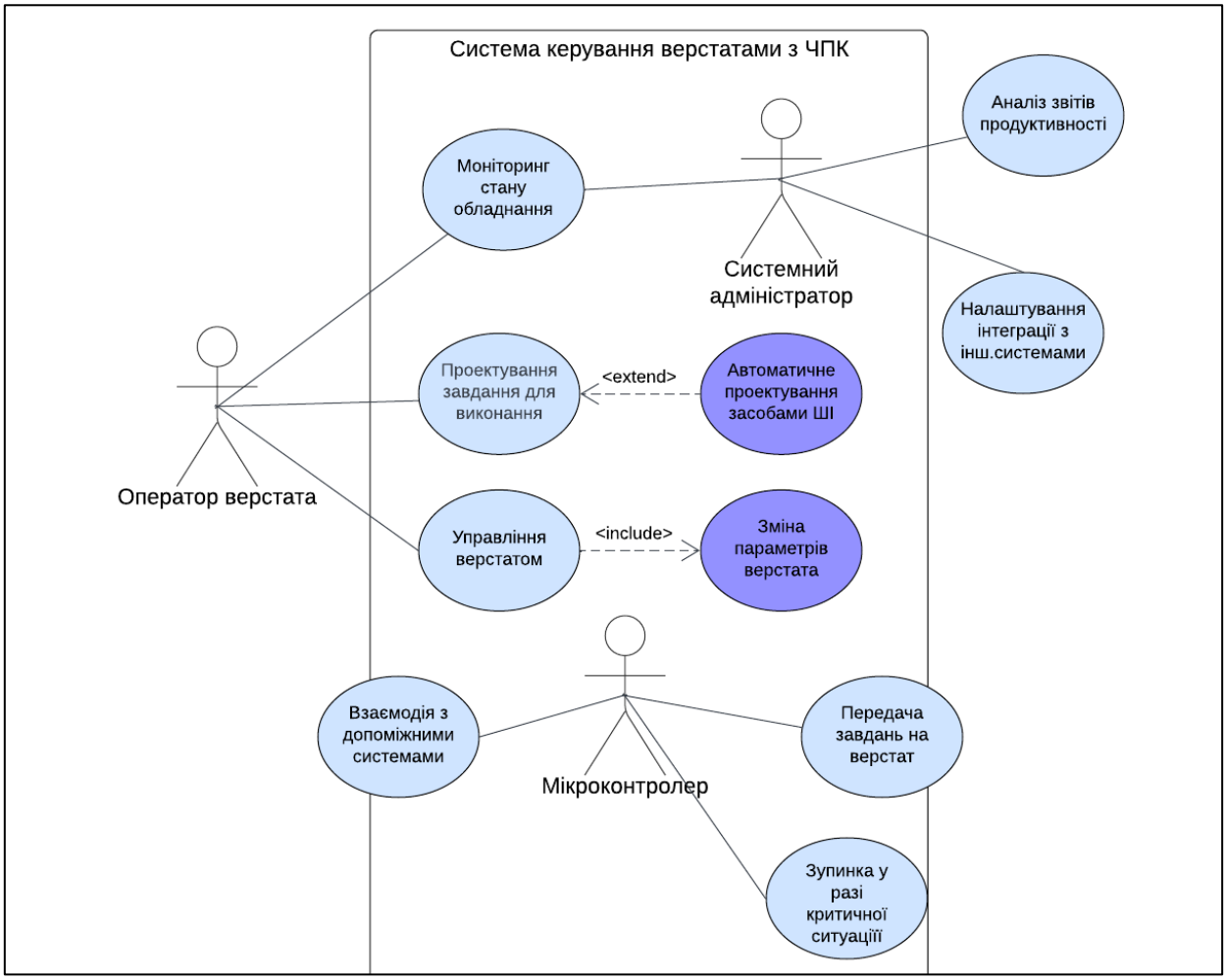

Рисунок 2.1 – Діаграма прецедентів

Прецедент «Зміна параметрів роботи верстата».

Актори: оператор.

Основний потік подій: оператор обирає потрібний верстат зі списку підключених. У налаштуваннях змінює параметри роботи верстата та зберігає зміни.

Альтернативний потік подій: якщо оператор не знаходить верстат в списку підключених, він виконує з’єднання верстата з ПЗ, або перезавантажує середу керування.

Прецедент «Проєктування завдання для виконання».

Актори: оператор.

Основний потік подій: оператор відкриває модуль проєктування завдання в системі. Використовуючи засоби графічного інтерфейсу, створює модель завдання. Перевіряє створену модель на відповідність параметрам та відправляє її для моделювання руху інструменту відповідно до створеної моделі. Перевіряє на відсутність помилок, холостого ходу, двійного проходу по траєкторії. Зберігає створену модель для подальшої обробки.

Альтернативний потік подій 1: оператор завантажує файл у відповідному форматі до системи. Система аналізує завантажений файл та генерує G-code для подальшої передачі завдання не верстат.

Альтернативний потік подій 2: якщо оператор виявляє помилки у створеній моделі під час перегляду моделювання руху інструменту, він відкриває інтерфейс моделювання, виправляє помилки в моделі та зберігає зміни.

Прецедент «Автоматичне проєктування засобами штучного інтелекту».

Актори: оператор.

Основний потік подій: оператор ініціює автоматичне проєктування моделі завдання, тобто відкриває в системі модуль автоматичного проєктування. Завантажує макет необхідної моделі у відповідному форматі (наприклад, растрове зображення) та налаштовує параметри для генерування моделі. Система аналізує макет та введені параметри та створює файл моделі у відповідному форматі. Оператор переглядає створений файл на відповідність макета та зберігає для подальшої обробки.

Альтернативний потік подій: якщо оператор виявляє невідповідність створеної моделі до макета, він повертається до початкових налаштувань, змінює параметри генерування моделі та знову запускає процес автоматичного проєктування. При вдалій спробі зберігає створений файл.

Прецедент «Моніторинг стану обладнання».

Актори: оператор, системний адміністратор.

Основний потік подій: система моніторингу збирає дані про стан обладнання (температура, споживана потужність, швидкість руху інструменту тощо). Оператор або системний адміністратор переглядають дані на інформаційній панелі. Система надсилає попереджувальні повідомлення у разі досягнення критичних значень або виникненні несправностей.

Прецедент «Налаштування інтеграції з іншими системами».

Актори: системний адміністратор.

Основний потік подій: системний адміністратор відкриває в програмі модуль інтеграції. Обирає системи, з якими необхідно встановити з’єднання та задає параметри з’єднання. Система перевіряє введені параметри та виконує спробу з’єднання. У разі успішного з’єднання система надсилає відповідне повідомлення. Адміністратор зберігає налаштування та тестує коректність інтеграції.

Альтернативний потік подій: якщо спроба з’єднання виявляється неуспішною, система надсилає відповідне повідомлення із можливими причинами помилки. Системний адміністратор виправляє помилки та повторює спробу з’єднання.

Прецедент «Аналіз звітів продуктивності».

Актори: системний адміністратор.

Основний потік подій: системний адміністратор відкриває модуль аналітики продуктивності роботи. Обирає період та операції для аналізу. Система генерує звіт про продуктивність (час роботи, час виконання завдань прогнозований та фактичний, простої в роботі, відхилення від плану). Системний адміністратор аналізує отриманий звіт.

Прецедент «Передача завдань на верстат».

Актори: мікроконтролер.

Основний потік подій: мікроконтролер отримує команди G-коду від керуючої програми, перетворює їх в електричні сигнали та передає на виконання системам верстата.

Прецедент «Зупинка у разі критичної ситуації»

Актори: мікроконтролер.

Основний потік подій: мікроконтролер в процесі моніторингу стану верстата виявляє помилки в його роботі. Передає сигнал на керуючу програму та зупиняє роботу верстата. Система надсилає повідомлення із можливими причинами зупинки роботи верстата.

Прецедент «Взаємодія з допоміжними системами».

Актори: мікроконтролер.

Основний потік подій: мікроконтролер отримує інформацію від драйверів про стан допоміжних систем (система охолодження, вентиляції тощо). Надсилає отриману інформацію в керуючу програму. Програма зберігає дані на інформаційній панелі.

2.3 Основи мови G-code

G-code (Geometric Code) – це мова програмування для верстатів з ЧПК, що розшифровується як «геометричний код» та використовується для керування автоматизованими верстатами з числовим програмним керуванням. Команди G-коду вказують верстату, як переміщувати інструмент, як швидко переміщувати інструмент, якого шляху дотримуватись, як змінювати інструменти та виконувати інші дії, необхідні для виробництва деталей із високою точністю.

У випадку таких верстатів, як токарний або фрезерний верстат, різальний інструмент керується цими командами, щоб дотримуватись певної траєкторії, відрізаючи матеріал, для отримання бажаної форми.

В такий же спосіб, у випадку адитивного виробництва або 3D-принтерів, команди G-коду надають інструкції верстату наносити матеріал шар за шаром, утворюючи точну геометричну форму.

G-code має доволі специфічну та структуровану форму. Цей код

формується із набору текстових інструкцій, які керують механічними та іншими параметрами принтера під час друку. Кожний командний рядок G-коду має певний синтаксис та відповідає лише одній команді.

Більшість G-code команд складаються з ідентифікації та параметрів і мають таку структуру: «G## X## Y## Z## F##». Наприклад, команда «G03 X30 Y30 I10 J0» визначає кругове інтерполювання проти годинникової стрілки, тобто використовується для виконання кругових або дугових рухів проти годинникової стрілки.

Нижче перераховані основні структурні елементи G-code.

Команди руху. Ці команди керують рухами осей принтера. Структура команди складається з власне самої команди та трьох координат. Наведемо перелік найчастіше використовуваних команд:

- «G00» – швидке переміщення без друку;
- «G01» – контрольоване переміщення з друкуванням;
- «G02» і «G03» – дугове переміщення для створення кругів або кривих за годинниковою стрілкою та проти відповідно;
- «G20» та «G21» – вибір одиниць виміру (G20 = дюйми, G21 = міліметри);
- «G17», «G18», «G19» – вибір робочої площини верстата (G17 – площина XY, G18 – площина XZ, G19 – площина YZ);
- «G28» – повідомляє верстату перемістити інструмент у його контрольну точку або початкове положення (щоб уникнути зіткнення, включається проміжна точка з параметрами X, Y і Z; інструмент пройде через цю точку перед переходом до контрольної точки; синтаксис команди «G28 X## Y## Z##»).

Команди налаштувань. Ці команди використовуються для встановлення різноманітних параметрів роботи принтера, таких як температура, швидкість подачі матеріалу тощо:

- «M104» – встановлення температури сопла;
- «M140» – встановлення температури платформи;

– «M109» – чекати до досягнення заданої температури сопла перед продовженням;
– «M190» – чекати до досягнення задано температури платформи.

Команди управління подачею матеріалу. Ці команди регулюють подачу філамента. E – параметр, який вказує величину філамента для подачі під час друку.

Параметри. Більшість G-code команд мають параметри, такі як координати X, Y, Z для вказівки позиції, а також `F` для контролю швидкості переміщення.

Коментарі. Коментарі в G-code починаються з символу «;». Вони не впливають на виконання програми, але корисні для пояснення частин коду, параметрів або змін.

Циклічні та умовні інструкції. Деякі розширені версії G-code дозволяють використання циклічних (повторення) та умовних (вибіркове виконання) інструкцій, хоча це не дуже поширено для стандартних 3D-принтерів.

Кінцевий код. На завершення друку, G-code містить інструкції для «закінчення» роботи, як-от вимкнення нагрівачів, вивід сопла з робочої зони тощо:

– «M84» – вимкнення крокових двигунів;
– «M104 S0» – охолодження сопла.

Ці елементи складають базову структуру програми G-code для 3D друку, дозволяючи детально контролювати всі аспекти процесу друку. Вони роблять G-code потужним інструментом для точного керування 3D принтерами, лазерними та іншими типами верстатів з ЧПК [20].

Наведемо приклад застосування простого коду керування (див. рис. 2.2).

За допомогою першої команди G00 інструмент переводиться з початкової позиції в точку B(5,5). Звідти починається «різання» зі швидкістю подачі 200 за допомогою команди G01. В інших рядках коду відбуваються переходи по вершинах фігури.

```
G21 G17 G90 F100
M03 S1000
G00 X5 Y5                ; point B
G01 X5 Y5 Z-1            ; point B
G01 X5 Y15 Z-1           ; point C
G02 X9 Y19 Z-1 I4 J0     ; point D
G01 X23 Y19 Z-1          ; point E
G01 X32 Y5 Z-1           ; point F
G01 X21 Y5 Z-1           ; point G
G01 X21 Y8 Z-1           ; point H
G03 X19 Y10 Z-1 I-2 J0   ; point I
G01 X13 Y10 Z-1          ; point J
G03 X11 Y8 Z-1 I0 J-2    ; point K
G01 X11 Y5 Z-1           ; point L
G01 X5 Y5 Z-1            ; point B
G01 X5 Y5 Z0
G28 X0 Y0
M05
M30
```

Рисунок 2.2 – Приклад коду керування

На рисунку 2.3 можна побачити результат роботи наведеного прикладу коду керування.

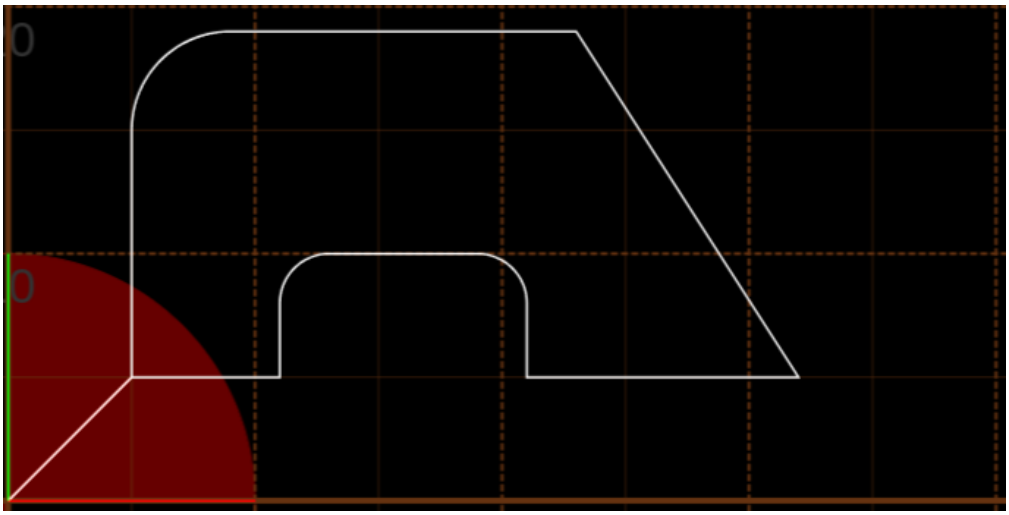

Рисунок 2.3 – Приклад роботи G-code

Отже, мова програмування G-code дозволяє точно керувати процесами обробки та друку, встановлюючи параметри рухів, швидкості, затримки та інші важливі аспекти операцій 3D-друку.

3 РОЗРОБКА ТА ТЕСТУВАННЯ ЗАСТОСУНКУ

В даному розділі описано засоби розробки та отриманий програмний застосунок. Розглянуто застосунок генерації керуючих кодів G-code для 3D друку, зокрема, для принтера CreateBox Dx.

3.1 Програмна реалізація застосунку для 3D-друку

Програми-слайсери для 3D друку відіграють ключову роль у процесі адитивного виробництва, перетворюючи тривимірні цифрові моделі на серію горизонтальних шарів. Ці шари, або «зрізи», використовуються 3D принтерами як інструкції для покрокового створення фізичних об’єктів.

Процес починається з імпорту 3D моделі у форматі, такому як STL, який зберігає зовнішню геометрію об’єкта у вигляді системи з тріангульних полігонів. Перед слайсингом користувач має встановити параметри друку, такі як товщину шару, розміри області друку.

Слайсер аналізує модель і перетворює її на серію шарів, визначаючи оптимальний маршрут руху сопла принтера для кожного шару. На основі цих розрахунків слайсер генерує G-code – текстовий файл з детальними інструкціями для 3D принтера, що керують всіма аспектами друку.

Після генерації, готовий G-code можна завантажити на 3D принтер через USB, SD-карту або мережу, дозволяючи принтеру автономно виробляти заданий об’єкт. Принтер використовує цей код для послідовного нанесення матеріалу шар за шаром, відтворюючи заданий у цифровій моделі об’єкт у фізичному вигляді.

Для програмної реалізації використовувався вебфреймворк Streamlit [21] та STL to G-Code Slicer API [22].

Приклад зовнішнього вигляду застосунку наведено на рисунку 3.1.

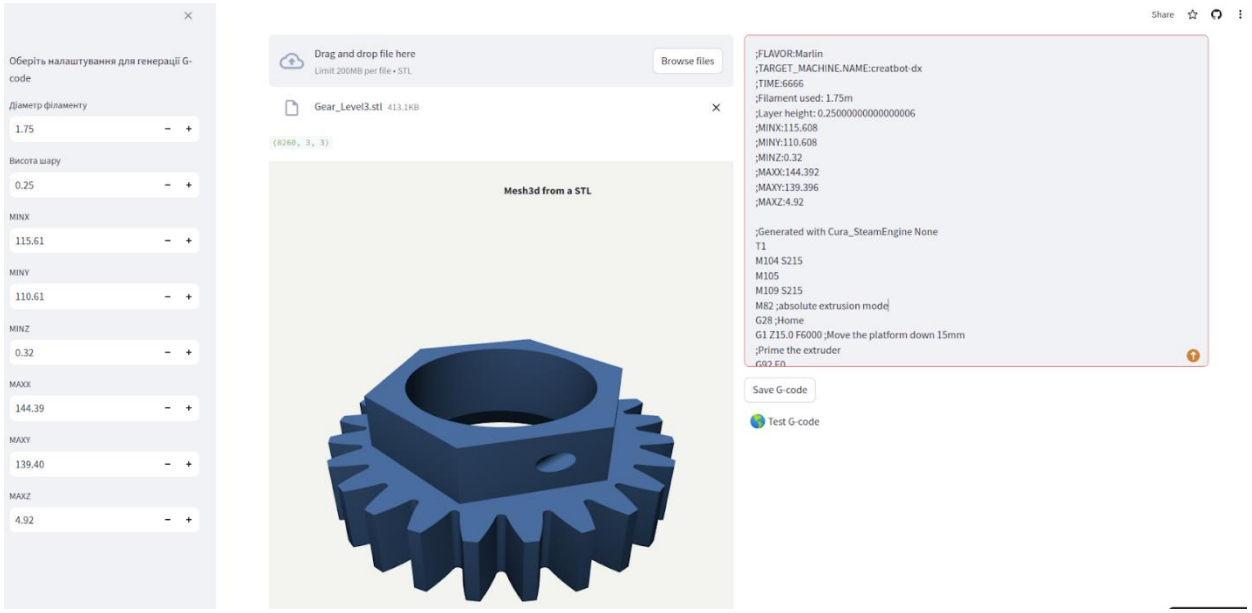

Рисунок 3.1 – Приклад роботи застосунку

Бокове меню призначене для завдання параметрів друку. Центральна частина – для виводу зображення, згенерованого з STL файлу для контролю якості моделі. Це важлива частина програм-слайсерів, оскільки якість 3D друку тісно пов’язана з якістю вхідної STL моделі, яка представлена у вигляді сітки з тріангульних полігонів. Роздільна здатність моделі, що визначається густотою та якістю цих полігонів, безпосередньо впливає на деталізацію та гладкість поверхонь готового виробу. Висока густота полігонів може забезпечити кращу деталізацію, але також збільшує розмір файлу і час обробки слайсером. Недоліки у структурі полігонів, як-от неправильна орієнтація або перекривання, можуть спричинити дефекти у друкованій моделі.

Масштабування та орієнтація моделі також мають важливе значення. Некоректний масштаб може призвести до проблем зі збіркою деталей, а неоптимальна орієнтація – до надмірного використання підтримок і ризику деформації під час друку. Технічні помилки у файлі, як-от зламані поверхні чи подвійні краї, перешкоджають правильному визначенню контурів та товщин стінок моделі, що може вплинути на міцність та естетику готового виробу.

Приклад згенерованого G-code можна побачити на рисунку 3.2. Структура коду для 3D друку передбачає визначення параметрів друку у

заголовку файлу gcode та в кінці. Приклад на рисунку 3.2 містить визначення меж друку, типу принтера та пластика.

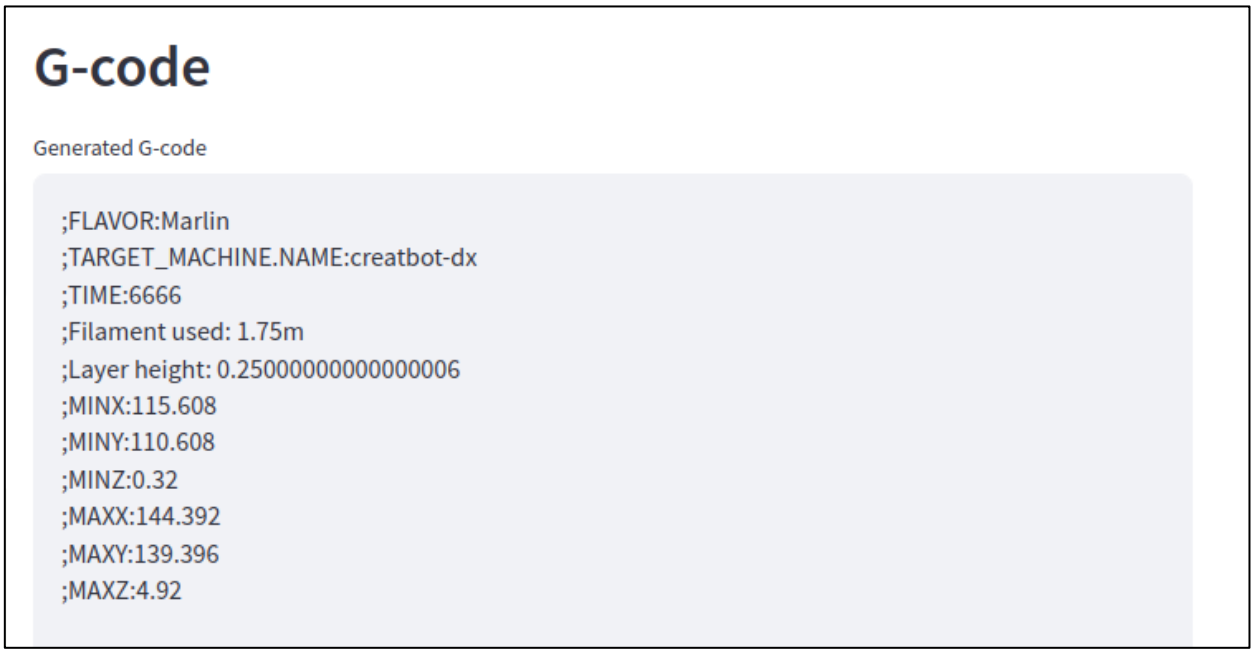

Рисунок 3.2 – Приклад згенерованого G-code

Наразі для генерації G-code використовується відкрите API STL to G-Code Slicer API. Його використання дозволяє автоматизувати процес перетворення 3D моделей у форматі STL до G-code, який можна використати для 3D друку. Цей процес включає кілька кроків:

– реєстрація та отримання API ключа;
– налаштування програмного середовища для роботи із запитами, використання бібліотеки Requests;
– відправка STL файлу до API через HTTP POST запит.

Результатом є посилання на файл з командами G-code у вигляді байт-рядка. Код функції генерації G-code наведений на рисунку 3.3.

Функцію візуалізації STL файлу засобами бібліотеки Plotly зображено на рисунку 3.4. Особливістю бібліотеки та її перевагою для подібних інженерних систем є її інтерактивність. Plotly генерує графіки, які дозволяють користувачам взаємодіяти з даними через інтерфейс. Користувачі можуть збільшувати та зменшувати масштаб, переміщатися по графіку, переглядати

деталі даних при наведенні курсора миші та інші інтерактивні елементи, які покращують наочність даних.

```
def get_gcode(stl_file):


    url = "https://stl-to-g-code-slicer.p.rapidapi.com/3dslicer-02/slice"

    # Define the path to the STL file you want to send
    stl_file_path = stl_file  # Replace with the actual file path

    # Load the STL file
    files = {"stl_file": open(stl_file_path, 'rb')}

    payload = {
        "param": "-s roofing_monotonic=true -s roofing_layer_count=0"
        # Add more parameters as needed without removing these two
    }

    headers = {"X-RapidAPI-Key": 'bf1ee2a9csdfsdfdhdhdefds2msh0bec7418b37e02ep16d6c3jsn21d5ebeba90c',
               "X-RapidAPI-Host": "stl-to-g-code-slicer.p.rapidapi.com",
    }

    response = requests.post(url, data=payload, files=files, headers=headers)

    return response
```

Рисунок 3.3 – Використання STL to G-Code Slicer API

```
def plot_stl_plotly(stl_filename):
    # https://chart-studio.plotly.com/~empet/15276/converting-a-stl-mesh-to-plotly-gomes/#/


    def stl2mesh3d(stl_mesh):
        # stl_mesh is read by nympy-stl from a stl file; it is  an array of faces/triangles (i.e. three 3d points)
        # this function extracts the unique vertices and the lists I, J, K to define a Plotly mesh3d
        p, q, r = stl_mesh.vectors.shape #(p, 3, 3)
        # the array stl_mesh.vectors.reshape(p*q, r) can contain multiple copies of the same vertex;
        # extract unique vertices from all mesh triangles
        vertices, ixr = np.unique(stl_mesh.vectors.reshape(p*q, r), return_inverse=True, axis=0)
        I = np.take(ixr, [3*k for k in range(p)])
        J = np.take(ixr, [3*k+1 for k in range(p)])
        K = np.take(ixr, [3*k+2 for k in range(p)])
        return vertices, I, J, K

    my_mesh = mesh.Mesh.from_file(stl_filename)
    my_mesh.vectors.shape
    vertices, I, J, K = stl2mesh3d(my_mesh)
    x, y, z = vertices.T
    colorscale= [[0, '#0a6da4'], [1, '#0a6da4']]
    mesh3D = go.Mesh3d(x=x, y=y, z=z, i=I, j=J, k=K, flatshading=True, colorscale=colorscale, intensity=z, name='Mesh', showscale=False)

    title = "Mesh3d from a STL"
    layout = go.Layout(paper_bgcolor='rgb(243,244,239)',
               title_text=title, title_x=0.5,
                      font_color='white',
               width=800,
               height=800,
               scene_camera=dict(eye=dict(x=1.25, y=-1.25, z=1)),
               scene_xaxis_visible=False,
```

Рисунок 3.4 – Візуалізація STL-файлу

У наступному розділі розглянемо приклади роботи розробленого застосунку.

3.2 Приклади роботи розробленого застосунку

Спочатку для тестування роботи застосунку виконаємо генерацію G-code для декількох STL моделей. Для перевірки якості згенерованих команд використовується онлайн емулятор WebGcode [23].

Після завантаження кодів емулятор показує 3D візуалізацію траєкторії інструмента, яка відповідає рухам, заданим у G-code. Це дозволяє візуально перевірити шлях руху екструдера та зони, де можуть виникнути зіткнення або інші проблеми. Так можна ідентифікувати частини G-code, де можуть виникнути помилки, такі як неправильні рухи, занадто близькі проходи або потенційні зіткнення. Також емулятор відображає швидкості переміщення екструдера, що дозволяє аналізувати, чи є швидкості друку оптимальними для кожної частини моделі. Занадто високі або низькі швидкості можуть впливати на якість друку. Результати емулювання тестових моделей наведено на рисунках 3.5 та 3.6 – це траєкторії руху 3D-принтера для різних деталей.

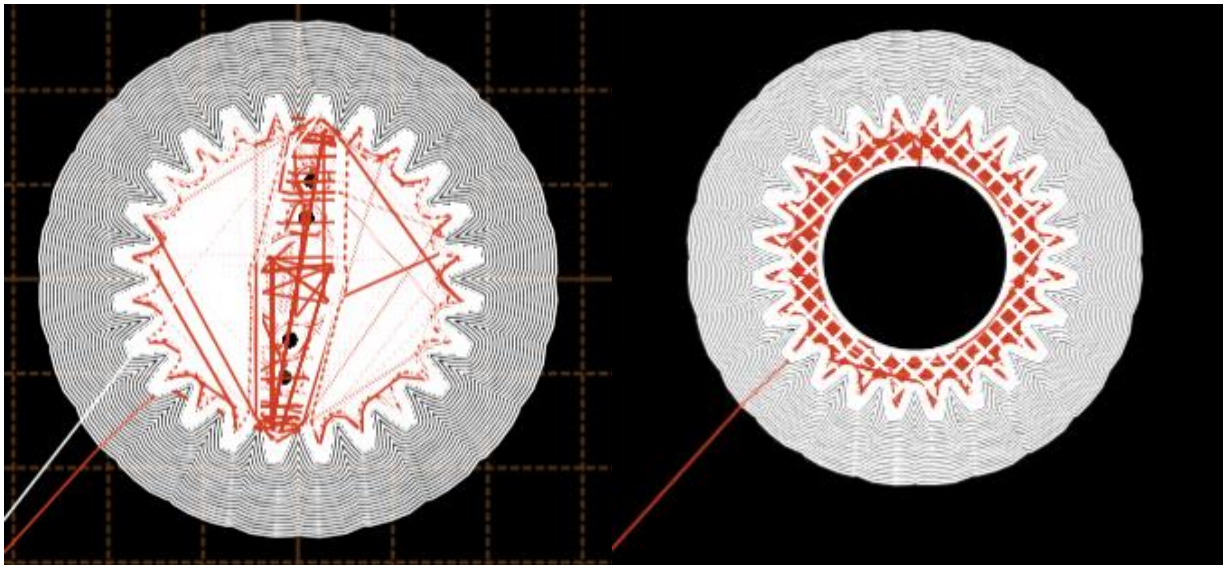

Рисунок 3.5

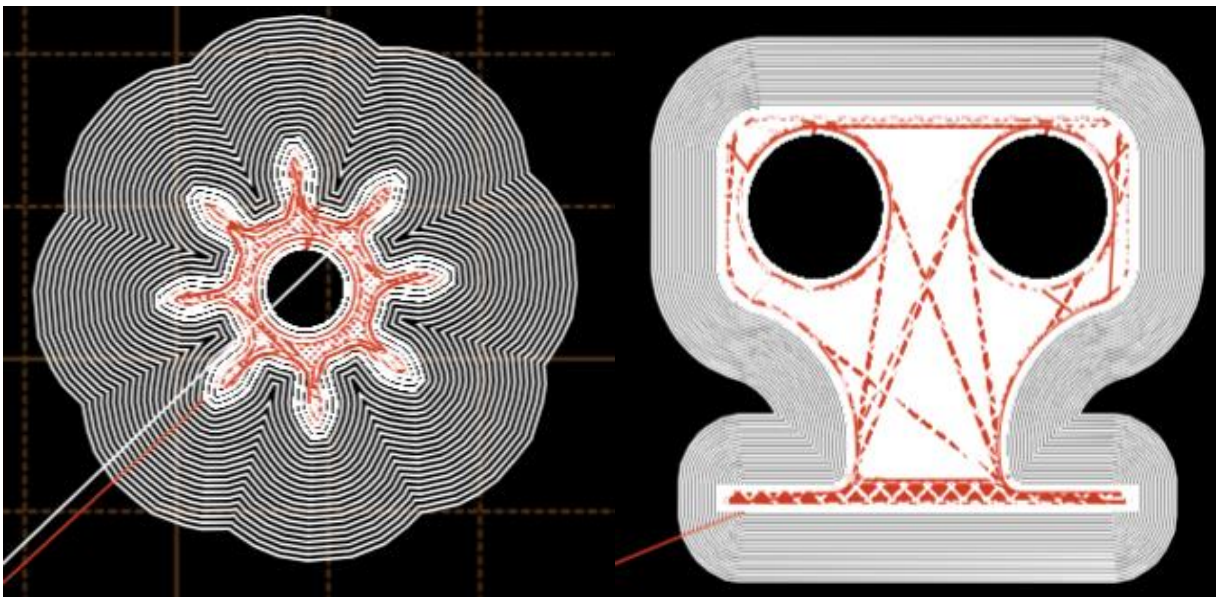

Рисунок 3.6

На рисунку можна побачити певні недоліки у рухах екструдера, оскільки деякі частини моделей залишаються не заповненими.

Найбільш вдалі моделі було роздруковано на 3D принтері CreateBox Dx (див. рис. 3.7).

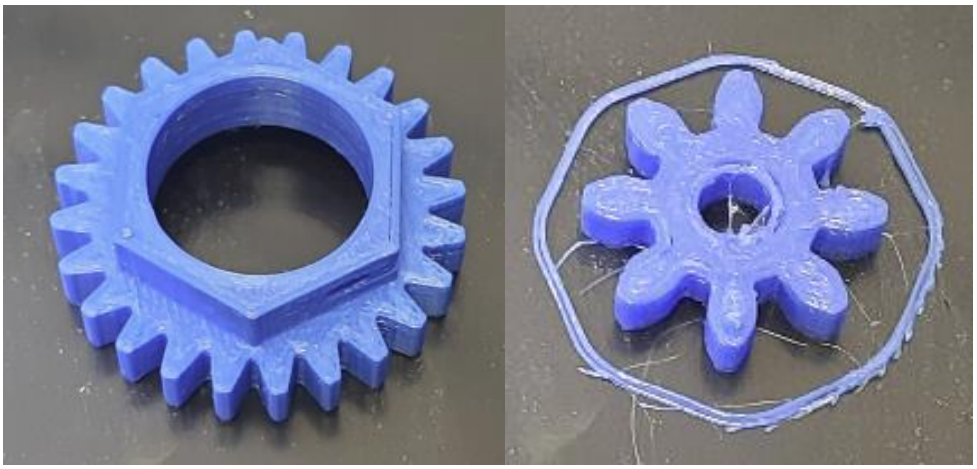

Рисунок 3.7 – Приклад виготовлення деталей

Отже, розроблений вебзастосунок для генерації G-code з моделей у форматі STL – важливий крок в автоматизації процесів підготовки моделей до обробки на верстатах з ЧПК. Подальше вдосконалення продукту та розширення функціонала дозволить ще більше автоматизувати процес підготовки моделей та керування верстатами з ЧПК.

ВИСНОВКИ

У кваліфікаційній роботі «Розробка автоматизованої середи керування верстатами» було створено застосунок для генерації керуючих G-кодів для 3D-друку.

У процесі роботи було виконано аналіз подібних наявних систем, розглянуто основні характеристики та принципи роботи верстатів з числовим програмним керуванням. Визначено основні функціональні та нефункціональні вимоги до розроблюваної середи керування і наведено візуалізацію основних вимог та функцій системи у вигляді діаграми прецедентів.

Основні результати дослідження та розробки:

- проведений опис сучасних систем керування верстатами з ЧПК, виділено основні переваги та недоліки кожної з систем, що дозволило визначити вимоги до розроблюваної системи;
- розглянуті основні характеристики верстатів з ЧПК;
- на прикладі лазерного верстата розглянуто докладніше принцип роботи верстатів з числовим програмним керуванням;
- визначено основні функціональні та нефункціональні вимоги до програмного забезпечення, на основі цього розроблено діаграму прецедентів з детальним описом кожного прецеденту;
- розроблено вебзастосунок генерації керуючих G-кодів засобами вебфреймворків Streamlit та STL to G-Code Slicer API.

Подальша робота над застосунком включає в себе розширення функціоналу застосунку, розробка інтерпретатора G-code для передачі завдання на 3D-принтер безпосередньо через застосунок та впровадження можливості налаштування параметрів 3D-принтеру.

ПЕРЕЛІК ПОСИЛАНЬ

1. Ingrassia D., Molenaar J.-M. Mastering Digitally Controlled Machines. Apress Media LLC., 2024. 285 p.
2. Radhakrishnan P. Computer Numerical Control Machines. New Delhi : New Central Book Agency, 2020. 167 p.
3. Suh S.-H., Kang S.-K., Chung D.-H., Stroud I. Theory and design of CNC systems. London : Springer Verlag, 2008. 456 p.
4. Overby A. CNC Machining Handbook: Building, Programming, and Implementation. McGraw-Hill\/TAB Electronics, 2010. 274 p.
5. Smith R. J. Arduino in Science: Collecting, Displaying, and Manipulating Sensor Data. Apress, 2021. 505 p.
6. ARDUINO UNO WiFi REV2. *Arduino Official Store*. URL: https://store.arduino.cc/products/arduino-uno-wifi-rev2?queryID=ee489fabf977e5974ed5f969afe4a2ee (дата звернення: 05.03.2024).
7. Лазерний верстат CO2 L-Laser L1490. *Larsen*. URL: https://www.larsen.ua/ua/shop/oborudovanie/lazernye-stanki/so2-lazernye-gravery-l-laser/lazernyy-stanok-co2-l-laser-l1490/ (дата звернення: 10.03.2024).
8. Powell J. CO_2 Laser Cutting. London : Springer London, 1998. 248 p.
9. VCarve. *Vectric*. URL: https://www.vectric.com/products/vcarve/ (дата звернення: 10.03.2024).
10. VCarve Pro V9.5 User Guide. *Vectric*. URL: https://docs.vectric.com/docs/V9.5/VCarvePro/ENU/Help/User%20Guide/Interface/Interface%20Overview.html (дата звернення: 10.03.2024).
11. NX software | Siemens Software. *Siemens Digital Industries Software*. URL: https://plm.sw.siemens.com/en-US/nx/ (дата звернення: 10.03.2024).
12. Chanatry W. Adaptive UI in NX – NX Design. NX Design. URL:

https://blogs.sw.siemens.com/nx-design/new-adaptive-ui/ (дата звернення: 10.03.2024).

13. LinuxCNC. URL: https://linuxcnc.org/ (дата звернення: 12.03.2024).
14. Newfangled Solutions Mach3. Newfangled Solution CNC Software Home of Mach3s. URL: https://www.machsupport.com/software/mach3/ (дата звернення: 12.03.2024).
15. Mach3 огляд інтерфейсу та основні функції. Поєднання Клавіш. Фрезерний верстат. URL: https://raptorcnc.com.ua/ua/a476953-mach3-obzor-interfejsa.html (дата звернення: 12.03.2024).
16. Lightburn Software. URL: https://lightburnsoftware.com/ (дата звернення: 14.03.2024).
17. Ultimaler Cura. *Ultimaker*. URL: https://ultimaker.com/software/ultimaker-cura/ (дата звернення: 14.03.2024).
18. Welcome to Ultimaker Cura 3.0. *Ultimaker*. URL: https://ultimaker.com/learn/welcome-to-ultimaker-cura-3-0/ (дата звернення: 14.03.2024).
19. Bittner K., Spence I. Use Case Modelling. Addison-Wesley Professional, 2002. 368 p.
20. Tran T. CNC Programming Tutorials Examples G & M Codes: G & M Programming Tutorial Example Code for Beginner to Advance Level CNC Machinist. Amazon Digital Services LLC – KDP Print US, 2019. 360 p.
21. A faster way to build and share data apps. *Streamlit*. URL: https://streamlit.io (дата звернення: 26.04.2024).
22. STL to G-Code API. *Postman*. URL: https://www.postman.com/slicegenius/workspace/slicegenius/collection/31963153-e30a125f-c968-4a34-8220-0caaf690c75a (дата звернення: 26.04.2024).
23. G-Code Simulator. *Git Hub Pages*. URL: https://nraynaud.github.io/webgcode/ (дата звернення: 15.05.2024).