РЕФЕРАТ

Магістерська дисертація містить: 110 с., 32 рис., 17 табл., 4 дод., 39 джерел.

Актуальність роботи. Процес розкрою біополімерних матеріалів є складним ресурсномістким процесом, що залежить від багатьох факторів. Зростання обсягів виробництва біополімерних матеріалів, зокрема шкіри і хутра, призводить до зростання конкуренції у цій галузі та вимагає підвищення якості продукції, що виробляється. Тому оптимізація процесу розкрою біополімерних матеріалів дозволить зменшити кількість відходів, знизити вартість кінцевого продукту, підвищити якість такої продукції та зробить її конкурентоспроможною.

Зв'язок роботи з науковими програмами, планами, темами. Роботу виконано у рамках науково-дослідної роботи 16.02.49 ДБ «Фізико-хімічні основи формування біополімерних матеріалів спеціального призначення»

Мета і задачі дослідження. Мета роботи – оптимізація процесу з планування розкрою біополімерних матеріалів. Задачами дослідження є аналіз існуючих автоматизованих систем з планування розкрою шкіри; формалізація завдання з розкрою шкіри; оптимізація процесу планування розкрою шкіри; розроблення програмного модулю для оптимізації схем розкрою шкіряного матеріалу.

Об'єкт дослідження – технологічний процес розкрою матеріалу на деталі шкіргалантерейних виробів.

Методи дослідження – комп'ютерне моделювання, теорії оптимізації.

Практичне значення одержаних результатів. Практичне значення результатів. Запропонований алгоритм оптимізації процесу планування розкрою біополімерного матеріалу може бути рекомендований для практичного використання. Створене програмне забезпечення може бути застосоване у складі програмного забезпечення інформаційно-обчислювальних систем для оптимізації існуючих виробництв, що

використовують дані процеси, або для нових підприємств, що планують використовувати вказану технологію.

Апробація результатів дисертації. Апробація результатів дослідження здійснювалася на науково-практичних та науково-технічних міжнародних конференціях: основні положення та результати роботи оприлюднено на 7-й міжнародній науково-практичній конференції "Комп'ютерне моделювання в хімії та технологіях і системах сталого розвитку – КМХТ-2019", Київ (2019) та V міжнародній науково-технічній конференції "Комп’ютерне моделювання та оптимізація складних систем - КМОСС-2019, Дніпро (2019)

Публікації. Основні положення та результати дослідження опубліковано в матеріалах конференцій "Комп'ютерне моделювання в хімії та технологіях і системах сталого розвитку – КМХТ-2019" та "Комп’ютерне моделювання та оптимізація складних систем - КМОСС-2019"

БІОПОЛІМЕРНИЙ МАТЕРІАЛ, РОЗКРОЮВАННЯ, КОМП’ЮТЕРНЕ МОДЕЛЮВАННЯ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, СТАРТАП-ПРОЕКТ

РЕФЕРАТ

Магистерская диссертация содержит 110 с., 32 рис., 17 табл., 4 доп., 39 источников.

Актуальность работы. Процесс раскроя биополимерных материалов является сложным ресурсоемким процессом, зависит от многих факторов. Рост объемов производства биополимерных материалов, в частности кожи и меха, приводит к росту конкуренции в этой области и требует повышения качества продукции. Поэтому оптимизация процесса раскроя биополимерных материалов позволит уменьшить количество отходов, снизить стоимость конечного продукта, повысить качество такой продукции и сделает ее конкурентоспособной.

Связь работы с научными программами, планами, темами. Работа выполнена в рамках научно-исследовательской работы 16.02.49 ДБ «Физико-химические основы формирования биополимерных материалов специального назначения»

Цель и задачи исследования. Цель работы - оптимизация процесса по планированию раскроя биополимерных материалов. Задачами исследования является анализ существующих автоматизированных систем по планированию раскроя кожи; формализация задачи по раскрою кожи; оптимизация процесса планирования раскроя кожи; разработка программного модуля для оптимизации схем раскроя кожаного материала.

Объект исследования - технологический процесс раскроя материала на детали кожгалантерейных изделий.

Методы исследования - компьютерное моделирование, теории оптимизации.

Практическое значение результатов. Предложенный алгоритм оптимизации процесса планирования раскроя биополимерного материала может быть рекомендован для практического использования. Созданное программное обеспечение может быть применено в составе программного

обеспечения информационно-вычислительных систем для оптимизации существующих производств, использующих данные процессы, или для новых предприятий, которые планируют использовать указанную технологию.

Апробация результатов диссертации. Апробация результатов исследования осуществлялась на научно-практических и научно-технических международных конференциях: основные положения и результаты работы обнародован на 7-й международной научно-практической конференции "Компьютерное моделирование в химии и технологиях и системах устойчивого развития - КМХТ-2019", Киев (2019) и V международной научно-технической конференции "Компьютерное моделирование и оптимизация сложных систем - КМОСС-2019, Днепр (2019)

Публикации. Основные положения и результаты исследования опубликованы в материалах конференций "Компьютерное моделирование в химии и технологиях и системах устойчивого развития - КМХТ-2019" и "Компьютерное моделирование и оптимизация сложных систем - КМОСС-2019"

БИОПОЛИМЕРНЫЙ МАТЕРИАЛ, РАСКРОЙ, КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ, ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ, СТАРТАП-ПРОЕКТ

ABSTRACT

The master's thesis contains: 110 pages, 32 fig., 17 tables, 4 apps, 39 sources.

Relevance of work. The process of cutting biopolymer materials is a complex resource-intensive process that depends on many factors. The increase in production of biopolymer materials, in particular leather and fur, leads to increased competition in this field and requires improved production quality. Therefore, optimizing the process of cutting biopolymer materials will reduce the amount of waste, reduce the cost of the final product, improve the quality of such products and make it competitive.

Relationship with working with scientific programs, plans, topics. The work was performed within the framework of scientific research 16.02.49 DB «Physico-chemical bases of formation of special purpose biopolymer materials».

The aim and objectives of the study. The aim of the work is to optimize the process of planning cutting of biopolymer materials. The objectives of the study are to analyse the existing automated systems for leather cutting planning; formalization of the task of cutting the leather; optimization of leather cutting planning process; development of a software module for optimization of cutting schemes of leather material.

The object of the study is the technological process of cutting the material on the details of leather goods.

Research Methods - Computer Modeling, Optimization Theory.

The practical significance of the results. The proposed algorithm for optimizing the cutting process of biopolymer material can be recommended for practical use. The created software can be applied as part of software of information-computing systems for optimization of the existing productions using these processes, or for new enterprises planning to use the specified technology.

Evaluation of the results of the thesis. Presentation of research results was carried out at scientific-practical and scientific-technical international conferences: the main provisions and results of work were announced at the 7th international

scientific-practical conference "Computer Modeling in Chemistry and Technology and Systems of Sustainable Development - KMHT-2019", Kyiv (2019) and the 5th International Scientific and Technical Conference "Computer Modeling and Optimization of Complex Systems - CMOSS-2019, Dnipro (2019).

Publications. The main findings and results of the study are published in the conference materials "Computer Modeling in Chemistry and Technology and Systems for Sustainable Development - KMHT-2019" and "Computer Modeling and Optimization of Complex Systems - KMOSS-2019".

BIOPOLYMER MATERIAL, PATTERN CUTTING, COMPUTER SIMULATION, SOFTWARE, STARTUP PROJECT

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ 12
ВСТУП 13
1 ОСОБЛИВОСТІ ТЕХНОЛОГІЧНОГО ПРОЦЕСУ РОЗКРОЮ БІОПОЛІМЕРНИХ МАТЕРІАЛІВ 14
1.1 Технології виробництва біополімерних матеріалів 14
1.2 Моделювання процесу розкрою шкіри 23
Висновки за розділом 1 25
2 МОДЕЛЮВАННЯ ТА ОПТИМІЗАЦІЯ ПРОЦЕСІВ РОЗКРОЮ 26
2.1 Постановка задачі оптимізації 26
2.2 Обґрунтування вибору алгоритму оптимізації 28
2.2.1 Алгоритм оптимізації бджолиними колоніями 28
2.2.2 Генетичний алгоритм 29
Висновки за розділом 2 33
3 СТРУКТУРА ТА ПРИНЦИПИ ПОБУДОВИ EASYCUT 34
3.1 Структура EasyCut 37
3.2 Модуль обробки інформації зі сканера 39
3.3 Модуль введення даних, відображення та збереження 41
Висновки за розділом 3 49
4 ПРОГРАМА ТА МЕТОДИКА ВИПРОБУВАНЬ EASYCUT 50
4.1 Об’єкт випробувань 50
4.2 Мета випробувань 50
4.3 Загальні положення 51
4.4 Обсяг випробувань 51
4.5 Методика проведення випробувань 53
4.6 Вимоги з випробувань програмних засобів 58
4.7 Перелік робіт, що проводяться після завершення випробувань 58
4.8 Умови і порядок проведення випробувань 58
4.9 Матеріально-технічне забезпечення випробувань 59

4.10 Метрологічне забезпечення випробувань 59
Висновки за розділом 4 59
5 РОЗРОБКА СТАРТАП ПРОЕКТУ 60
5.1. Резюме: конкретизація бізнес-ідеї, мети стартапу, об'єкту дослідження, місця розробки у інноваційному ланцюжку цінності 60
5.2. Аналіз зовнішнього та внутрішнього середовища стартапу. Ключові фактори успіху. Договір на виконання НДР 62
5.3. Визначення потенційних споживачів 67
5.4. Ціна інноваційної пропозиції на ринку 68
5.5. Концепція бізнес-моделі проекту та карта бізнес-процесів реалізації проекту 75
5.6. Ризики стартап-проекту та методи управління ними 76
ВИСНОВКИ 79
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ 80
ДОДАТОК А 85
ДОДАТОК Б 86
ДОДАТОК В 87
ДОДАТОК Г 109

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

А – амортизація основних фондів, грн;

НДР – науково-дослідна робота;

ОбК – обігові кошти, грн;

ОЗ – основні засоби, грн;

ООП – об’єктно-орієнтоване програмування;

П – прибуток, грн;

ПЗ – програмне забезпечення;

Р – рентабельність, грн;

С – собівартість, грн;

ТОВ – товариство з обмеженою відповідальністю;

ФОП – фонд оплати праці, грн;

Ц – ціна, грн;

ВСТУП

Біополімерні матеріали, до яких відносяться шкіра та хутро, широко використовуються для виготовлення одягу, взуття та аксесуарів. Важливим завданням при проектуванні виробів зі шкіри та хутра є оптимізація процесу розкрою матеріалу на заготовки з метою зменшення обсягу відходів без зниження якості отриманих виробів.

Значний досвід вирішення оптимізаційних задач дозволяє отримувати рішення як для конкретних застосувань, так і в узагальненому вигляді. Проте особливості процесу розкрою шкіри та хутра вимагають спеціального опрацювання. У даній роботі розглянуті питання оптимізації процесу розкрою таких біополімерних матеріалів, як шкіра та хутро, обґрунтовано вибір методів оптимізації та досліджено їх ефективність, розроблено інтелектуальну підсистему підтримки прийняття рішень та розглянуто шляхи її адаптації до інформаційно-обчислювального комплексу типового підприємства з виготовлення шкіряних та хутрових виробів.

Вирішення сформульованих вище завдань виконано із застосуванням новітніх інформаційних технологій. Сучасні підприємства з виготовлення шкіряно-хутрових виробів здійснюють розкроювання матеріалів із застосуванням спеціального програмного забезпечення (ПЗ), однак значна частина таких програм пропонують можливість розкроювати матеріал лише прямокутних форм. Запропоновані у роботі рішення з автоматизації процесу розкрою дозволяють оптимізувати процес підготовки до розкрою і проектування схем розкрою, зменшити вплив людського фактору, зменшити кількість відходів та, як наслідок, знизити витрати на їх утилізацію й шкідливий вплив на навколишнє середовище.

У рамках виконання стартап проекту конкретизовано бізнес-ідею, визначено потенційних споживачів, обґрунтовано ціну інноваційної пропозиції на ринку, сформульовано концепцію бізнес-моделі проекту, виконано оцінку ризиків стартап-проекту та методи управління ними.

1 ОСОБЛИВОСТІ ТЕХНОЛОГІЧНОГО ПРОЦЕСУ РОЗКРОЮ БІОПОЛІМЕРНИХ МАТЕРІАЛІВ

Біополімерний матеріал (шкіра) – міцний і гнучкий довговічний матеріал, що отримується шляхом вироблення шкір в умовах традиційного господарства або промислового підприємства. Окрему категорію представляє хутряне виробництво. Шкіра використовується в різних областях, від виробництва взуття і одягу до палітурки книжок і виготовлення оббивки меблів і шкіряних шпалер. Проводиться безліч сортів шкіри з різноманітними властивостями [1].

У шкіри дійсно високої якості етапів технологічного циклу її обробки до готового виробу повинно бути чим менше, тим краще. «Обличчя» шкури (верхній шар шкіри) має бути повністю збережено, якщо не сказати, що підкреслено. Це означає, що повинні зберегтися зморшки і пори, наявні на шкірі, яка піддається обробці [2].

1.1 Технології виробництва біополімерних матеріалів

Технологія виробництва шкіри (переробка шкір до готової шкіри) складається з трьох етапів: підготовка шкіри, дублення шкіри, обробка шкіри [3].

1.1.1 Підготовчі процеси та операції.

Мета відмоки - привести шкуру в стан, максимально наближається до парному як за ступенем обводнення, так і по мікроструктурі. В процесі відмоки з шкури видаляються бруд, кров, консервуючи речовини, розчинні білки (альбуміни і глобуліни). Шкура при цьому поглинає значну кількість води. Відбувається зв'язування води з активними групами пептидних ланцюгів колагену (CO, -NH-, NHJ, COO-, OH ') за рахунок водневих зв’язків і сил електростатичного притягання. Це так звана вода гідратації. Вона не видаляється при механічному віджиманні і не здатна розчиняти інші речовини. Вміст води гідратації - 40-50% від маси сухого колагену.

Інша частина вологи, що міститься в шкірі після відмоки, називається вологим набуханням. Особливо сильне набухання дерми, яке називають нажором, відбувається в розчинах лугів і кислот.

Ступінь нажора дерми в підготовчих процесах істотно впливає на якість готової продукції. Зовні нажор виражається в збільшенні товщини дерми на 30-100% залежно від топографічного ділянки. При сильному нажоре структура дерми стає гомогенною і склоподібної. Вода утворює з білком єдину систему. В результаті нажора дерма стає пружною. Ступінь цієї пружності використовують як критерій нажора [3].

Рівномірність, отже, і режим відмоки залежать від маси шкури, способу її консервування та обладнання, що застосовується.

Шкури великої маси і більш товсті вимагають більшої тривалості відмоки для досягнення необхідного ступеня обводнення. Підвищення температури значно прискорює обводнення шкур, але ступінь обводнення при цьому знижується. У типових методиках рекомендується проводити відмоку при температурі 18-22 ° С.

Для прискорення відмоки застосовують обострювачі, які викликають іонізацію активних груп колагену, руйнують іонні і водневі зв'язку. Всі ці зміни підвищують проникність дерми, сприяють прискоренню її обводнення. Як обострювач застосовуються карбонат (Na2C03), сульфід (Na2S) або сульфіт (Na2S03), а також ПАР, що знижують поверхневий натяг води і поліпшують змочуваність шкіри.

Механічний вплив, яке викликається обертанням апаратури або перемішуванням, значно прискорює відмоку. Однак занадто тривале обертання барабана або занадто інтенсивне перемішування призводить до пухкості і отдуші.

Максимальна обводнення мокросолених шкір досягається за 6-8 год відмоки. Однак рівномірний розподіл води по товщині шкури спостерігається

лише через 10-12 год в при температурі 27-30 ° С або через 16-18 год при 22-25 ° С.

В результаті обводнення і вимивання розчинних білків в процесі відмоки відбувається збільшення внутрішньої поверхні структурних елементів дерми та зміна її фізіко-механічних властивостей. Обводненна шкура стає м'якшою, гнучкою і здатною взаємодіяти з хімічними речовинами [3].

Недостатня і нерівномірна обводненість шкури по товщині і площі в процесі відмоки може викликати стяжку і садку лицьовій поверхні шкіри в наступних процесах або поява її жорсткості. Великі втрати білкових речовин при відмоці призводять до пухкості шкіри лицьового шару. Бактеріальне ушкодження сировини є причиною появи на шкірі пошкоджень лицьової поверхні.

Знежирення. Жирові речовини природного походження необхідно видаляти ще й тому, що вони, перебуваючи в жирових клітинах, не здатні змащувати колагенові волокна і, отже, не можуть надати їм м'якість, еластичність.

У волосяному покриві жир розподіляється на поверхні стрижня волоса у вигляді плівки або крапель. Невелика кількість жиру знаходиться всередині волосся. До жиру прилипають різні забруднення (пісок, бруд, залишки корму). При обробці сировини жир ускладнює перебіг таких процесів, як відмока, фарбування, облаштування волосяного покриву.

Знежирення волосяного покриву овчини - один з основних процесів її вичинки. Волосяний покрив після знежирення набуває блиск і крихкість, рівномірність і «жвавість» забарвлення. Знежирення ведуть до вмісту жиру в волосі 1,5-2%. При більш низькому вмісті жиру погіршуються фізико-механічні властивості волоса, з'являються крихкість і ламкість, знижується стійкість до стирання.

Під час знежирювання шерсть жир гідролізується і розщеплюється на холестерин і вищі жирні кислоти, які з оцтовим ангідридом в присутності

сірчаної кислоти дають характерне зелене забарвлення. Цю реакцію використовують при експрес-метод визначення наявності жиру на волосині.

Обезволашування і зоління - це обробка сировини в лужному середовищі суспензією гідроксиду кальцію з додаванням сульфіду натрію. Якщо мета обезволашування - ослаблення зв'язку волоса з дермою з подальшим зняттям волосяного покриву на машинах або хімічне знищення волосяного покриву - його розчинення, то метою зоління є поділ або розпушення структури дерми.

Видалення волосяного покриву і епідермісу проводять наступним способами: зоління без збереження вовни, зоління зі збереженням шерсті, намазний спосіб, ферментативний спосіб [3].

Міздріння сировини - механічне видалення зі шкури підшкірної клітковини (міздрі). Цей процес сприяє більш швидкому і рівномірному протіканню наступних рідинних обробок. Процес міздріння сировини слід після процесу промивання перед відмокою. Якщо сировину піддають міздрінню на м'ясокомбінаті, то на шкіряних заводах цю операцію не проводять. Для більш повного видалення підшкірної клітковини проводять друге міздріння голини, що знаходиться в стані нажора. У виробництві шкір хромового дублення міздріння зазвичай проводять один раз в голині, в стані нажора.

Міздріння шкур здійснюють на міздрільних машинах. Шкура завішується на прогумований вал, до якого примикають рифлені транспортуючі вали. При включенні машини шкіра подається назустріч ножового валу (1400 об / хв), спіральні ножі якого зрізають міздрю.

Голина, отримана після завершення відмочувально-зольних процесів і механічних операцій, містить значну кількість з'єднань кальцію, як в сорбованом (-4%), так і в хімічно зв'язаному стані (-1,7%). Зоління голини знаходиться в набряклому стані.

Метою обеззолювання є ліквідація нажора і видалення з голини з'єднань кальцію і сірчистого натрію. Обеззолюванню передує промивка голини

проточною водою, при цьому віддаляються бруд і адсорбований гідроксид кальцію.

Після закінчення процесу м'якшення голину піддають пікелюванню, тобто обробці розчином нейтральної солі і кислоти, який називається пікелювальним розчином, або пікелем. Зазвичай використовують сірчану, мурашину або оцтову кислоти і хлорид натрію або сульфат амонію.

Призначення пікелювання - надання голини кислотності без нажору перед хромовим дубленням, а також консервування голини до дублення.

Найчастіше застосовуються пікелювальні розчини, що складаються з сірчаної кислоти і хлориду натрію або з сірчаної кислоти, органічної кислоти (мурашина, оцтова, фталева) і хлориду натрію.

Після закінчення підготовчих процесів і операцій в шкіряному виробництві напівфабрикат передають на дублення.

1.1.2 Дублення

Виробництво шкіри - це сукупність хімічних, фізико-хімічних і механічних процесів і операцій, серед яких одним з найважливіших є процес дублення.

Для дублення використовують широкий спектр хімічних речовин різного характеру.

Дублення є структурування білка дерми дубильними речовиною. Структурування, або формування, структури дерми означає фіксування досягнутої в підготовчих процесах підвищеного ступеня поділу волокон, в результаті чого істотно змінюються фізико-механічні властивості дерми.

Як відомо, основною складовою частиною дерми шкури є хвилясті білки: колаген (до 80%) і еластин. Крім того, в шкурі містяться глобулярні білки: альбумін, глобуліни.

Після дублення напівфабрикат стає пористим, волокна шкіри втрачають здатність склеюватися при сушінні. Збільшуються міцність при розтягуванні, стійкість до гниття і дії підвищеної температури і ферментів.

Однак всі види дублення призводять до усадки площі напівфабрикату. Усадка відновлюється частково в наступних механічних операціях. Термостійкість шкіри характеризується температурою зварювання, при якій починається зміна (зменшення) розмірів випробуваного зразка.

Процес дублення незворотній. Дубитель вимивається і не переходить в розчин з вичиненої напівфабрикату. Залежно від застосовуваного дубителя розрізняють дублення неорганічними і органічними речовинами, комбіноване дублення, а також дублення полімерами.

Мета нейтралізації шкір після дублення полягає в зниженні їх кислотності. В результаті нейтралізації відбувається подальше зміцнення зв'язку фіксованих хромових комплексів. Одночасно з цим спостерігається зміна пов'язаних хромових комплексів: з внутрішньої сфери витісняються кислотні залишки функціональними групами колагену, в комплекс частково входять аніонні залишки нейтралізують речовин, а також гідроксильні групи.

В якості нейтралізуючих речовин використовуються: бікарбонат натрію $NaHC0_3$, карбонат натрію Na_2C0_3, оксалат натрію NaOOC-COONa, тетраборат натрію (бура) $Na_2B_40_7$, гідрофосфат натрію Na_2HP0_4.

Для нейтралізації зазвичай використовують розчини з pH не більше 9, так як при pH більше 9 хромові комплекси руйнуються і відбувається раздублювання шкіри.

При нейтралізації з зовнішніх шарів кислота видаляється легше, ніж з середніх.

Нейтралізуючі речовини зазвичай додають в кількості 2% від маси напівфабрикату.

Після промивання шкіра повинна негайно надходити на фарбування і жирування. В іншому випадку залишилася в середньому шарі кислота буде дифундувати на поверхню лицьового шару і перешкоджати жируванню.

1.1.3 Обробка шкіри

Оброблення шкіри і хутра є основним інструментом у підвищенні конкурентоспроможності готової продукції. За допомогою оброблювальних процесів і операцій шкірі і хутрі задаються необхідні утилітарні і естетичні властивості.

Оброблювальними вважаються технологічні процеси і операції, які проводяться після дублення. Це - фарбування, наповнення, жирування, сушка, зволоження, облагороджування, механічні оздоблювальні операції, покривні фарбування.

Призначення обробки:

- надання шкірі і хутрі красивого зовнішнього вигляду;
- надання шкірі і хутрі необхідних фізико-механічних властивостей;
- збільшення виходу шкіри по площі.

Всі процеси обробки можна розділити на дві групи:

1) процеси, основу яких складають фізико-хімічні та хімічні закономірності;

2) операції, що базуються на механічних впливах на шкіру і хутро.

До перших відносяться: фарбування, відбілювання, наповнення, жирування, сушка, зволоження, облагороджування волосяного покриву, покривні фарбування; до других - розводка, розбивка, тяжка, шліфування, відкочування, прокатка, розчісування і стрижка волосяного покриву, обрізка країв, вимір площі. Ці дві групи обробок чергуються в залежності від видів вироблюваних шкіри і хутра.

Мета фарбування:

- надати напівфабрикату певний колір;
- облагородити забарвлення менш цінних видів сировини шляхом їх імітації під більш цінні види;
- усунути недоліки природного забарвлення: жовтизну каракулю і ін.

Жирування шкіряного і хутряного напівфабрикату полягає у введенні в дерму жируючих речовин, які, адсорбуючись на поверхні структурних елементів дерми, а також розташовуючись між ними, поділяють їх і надають гнучкість, м'якість і підвищену вологостійкість. Одночасно жируючи речовини збільшують взаємне ковзання структурних елементів, полегшуючи орієнтацію їх під впливом деформуючих зусиль. В результаті дерма набуває підвищену міцність і пластичність.

Технологічний процес виробництва шкіри і хутряного напівфабрикату побудований таким чином, що жирування завжди передує сушінні. Введені в дерму жируючи речовини, розташовуючись між структурними елементами колагену, перешкоджають взаємодії активних центрів суміжних елементів білка і утворення надлишкових поперечних зв’язків за участю дублячих з'єднань. Все це призводить до зниження усадочних напружень в структурі дерми, відповідно, до зменшення незворотною усадки напівфабрикату. Відомо, що найбільша усадка дерми відбувається в процесі видалення вологи гідратації, яка розташовується в найдрібніших капілярах структури колагену. Присутність жирових речовин в цих капілярах запобігає склеювання їх стінок, а значить, і необоротну усадку. Таким чином, рівномірний розподіл жирують речовин в дермі забезпечує зниження усадки, тобто збільшення виходу площі напівфабрикату, його високу міцність, м'якість і пластичність.

Розподіл жируючих речовин по верствам напівфабрикату триває в ході подальших оздоблювальних операцій, і особливо при пролежці, сушці і пресуванні.

Ефективність жирування залежить від підбору жирових речовин, їх кількості, техніки проведення процесу жирування. Різні за своєю природою жирові речовини неоднаково впливають на жорсткість, щільність, вологоємність, міцність, зносостійкість шкіри і хутра. Тому для додання шкірі і хутрі комплексу необхідних властивостей зазвичай використовують суміші різних жирових речовин.

В результаті проведення механічних операцій шкіра та хутро набувають необхідний зовнішній вигляд і упругопластичні властивості.

Механічні операції обробки шкіри діляться на наступні групи.

Операції для вирівнювання товщини напівфабрикату - стругання, шліфування.

Операції для видалення надлишку вологи - віджимання.

Операції для додання лицьовій поверхні необхідного зовнішнього вигляду - розводка, шліфування, пресування, прасування.

Стругання призначено для вирівнювання товщини вичиненої напівфабрикату, а також для отримання чистої, гладкої шкіри. Зістругують до 1%. Вологість при струганні не повинна бути більше 60%, інакше можливе утворення підрізів.

Шліфування. Мета - вирівняти товщину, надати ворсистість лицьовій поверхні шкіри (наприклад, велюр). Шліфування застосовується при виробленні шкір зі штучною лицьовою поверхнею. Шліфування проводять, в залежності від виду шкіри, два рази і більше. Наприклад, велюр шліфують три рази і більше. Напівфабрикат після шліфування містить велику кількість пилу. Пил віддаляється продувний струменем повітря на спеціальних машинах.

Віджимання. Після дублення і пролежить напівфабрикат зазвичай містить 65-75% вологи. Мета операції - механічне видалення вологи з вичиненої напівфабрикату на віджимних валових машинах або, наприклад, у виробництві юфти, шкір для низу взуття, технічних - на гідравлічних віджимних пресах. В результаті вологість знижується до 55-60% для напівфабрикату хромового дублення і до 45-52% - для юфти і шкір для низу взуття. Зайва волога негативно впливає на операції стругання в виробництві шкір хромового дублення, а також жирування розплавом жирують речовин у виробництві юфти і шкір для низу взуття.

У виробництві шкір хромового дублення виконують також другий віджимання вологи на валовій машині до тих же значень вологості

напівфабрикату після фарбувально-жиро-вальних процесів. Після чого підвищуються якість виконання розводки напівфабрикату і вихід шкір по площі.

Розводка. Мета - розгладити на розвідний машині лицьову поверхню. Усунути наявні складки, зморшки, згини, розправити периферійні ділянки. В результаті покращується зовнішній вигляд, збільшується вихід шкіри по площі на 1,5-2,0%. Вологість хромового напівфабрикату перед розводкою повинна становити 55-60%, напівфабрикату комбінованого дублення - 48-50%.

Пресування і прасування сприяють формуванню лицьовій поверхні шкіри, наданню їй красивого зовнішнього вигляду, завершення процесу плівкоутворення з покривний фарби, підвищенню адгезії покривний плівки. Пресування проводять 2-3 рази за допомогою гідропроцесів при температурі 65-100 ° С. Верхні плити - знімні. Вони можуть бути гладкими або з тих чи інших малюнком.

1.2 Моделювання процесу розкрою шкіри

До сих пір існуючі методи розкрою матеріалів для виробів зі шкіри засновані на виконанні одним працівником двох груп операцій - розміщення шаблонів і вирубання (вирізання) деталей. При цьому продуктивність праці і раціональне використання матеріалів в значній мірі залежать від суб'єктивних якостей кожного виконавця. Підвищення продуктивності праці обмежується фізичними можливостями робітника і послідовним способом розкрою.

Сучасні математичні методи і обчислювальна техніка дозволяють вирішувати завдання оптимального розміщення деталей на матеріалі, визначати прогресивні норми використання і потреба в сировині.

Для цього складаються такі завдання:

1. побудова оптимальних модельних шкал (розрахунок коефіцієнта укладання);
2. формування схем розміщення деталей;
3. оптимізація розмірів матеріалу;
4. розрахунок оптимальних комбінацій моделей для спільного розкрою;
5. розрахунок оптимального варіанта замовлення матеріалів (шкіри, штучні і синтетичні шкіри), необхідного для випуску заданого асортименту деталей і вибору оптимальних варіантів схем розміщення;
6. оптимізація цільового використання матеріалів для деталей низу взуття;
7. розрахунок потреби в матеріалах на етапі підготовки виробництва.

Основна увага в роботах, пов'язаних із застосуванням ЕОМ при розкрої матеріалів, приділяється оптимізації схем розкрою, так як від них залежать норми використання і потреба в сировині.

1.3. Постановка задачі

Метою даної роботи є розробка комп’ютерно-інтегрованих засобів для вирішення задачі оптимізації процесу планування розкрою шкіри.

У відповідності до поставленої мети сформульовані такі задачі: проаналізувати сучасний стан проблеми оптимізації процесів планування розкрою біополімерних матеріалів; визначити критерії оптимізації, цільову функцію та обмеження; запропонувати алгоритм та створити програмний продукт для оптимізації процесу планування розкрою; застосувати розроблену програму для оптимізації процесу планування розкрою шкіряних матеріалів.

Висновки за розділом 1

Проаналізовано особливості технологічного процесу розкрою біополімерних матеріалів на прикладі шкіряного виробництва та визначено основні проблеми існуючих методів розкрою. Встановлено, що більшість заводів з виробництва хутряних виробів не використовують автоматизоване проектування схем розкрою, а існуючі програмні продукти не дозволяють розкроювати вироби складної форми.

Вивчено сучасні методи моделювання та оптимізації процесу розкрою шкіри, обґрунтовано вибір генетичних алгоритмів для оптимізації процесу розкрою.

Сформульовано постановку задачі та виділено основні підзадачі відповідно до головної мети роботи.

2 МОДЕЛЮВАННЯ ТА ОПТИМІЗАЦІЯ ПРОЦЕСІВ РОЗКРОЮ

2.1 Постановка задачі оптимізації

Оптимізація розкрою полягає в ітераційному формалізованому процесі пошуку таких параметрів розташування заготовок на площині, за яких головна функція з урахуванням заданих обмежень – кількість матеріалу та його розміри – приймає найкращі значення [4].

Математична постановка задачі полягає в розміщенні плоских геометричних об'єктів на заготовках заданих розмірів з мінімальними відходами матеріалу і урахуванням існуючих обмежень. Обмеження першого типу – геометричні – є класичними і визначаються умовами приналежності заготовок до області розміщення, їх взаємного не перетинання.

Умови автоматизованого виробництва розширюють цей список обмеженнями другого типу – технологічними, які визначаються характеристиками розкрійного устаткування і організаційно-технологічними особливостями виробництва.

Проблема оптимізації полягає в двох найважливіших пунктах. Перший пункт – це формулювання критерію оптимальності. Формулювання критерію базується на конкретній, чітко визначеній суті постановки задачі, тобто критерій оптимальності напряму пов'язаний із потребами виробництва. Критерій оптимальності зазвичай під час розв'язування виробничих завдань входить до складу цільової функції. Виробничі завдання можуть мати як неперервний, так і дискретний характер. Таким чином, число допустимих рішень задачі є скінченним, а значення аргументів цільової функції повинні бути цілочисельними: максимально можлива кількість заготовок певного виду з одиниці матеріалу, що розкроюється, чи максимально можлива кількість різних деталей, що допущенні для розміщення на розкладці тощо.

Критеріями ефективності для задач розкроювання можуть слугувати такі показники, що забезпечуватимуть економію ресурсів (енергетичних і/або, матеріальних і/або, трудових). Найчастіше в якості критеріїв ефективності

приймають величину відходів з одиниці матеріалу, що розкроюється; тривалість циклу процесу розкроювання; вартість експлуатації апаратурного обладнання тощо.

З урахуванням вищезазначеного, задача оптимізації процесу розкрою біополімерного матеріалу полягає у пошуку таких параметрів розташування заготовок на площині, для яких забезпечується максимальне заповнення матеріалу деталями. Цільова функція має наступний вигляд:

$$P = \frac{\sum_{i=1}^{r} n_i S_i}{Sm} \cdot 100 \to \max, \qquad (2.1)$$

де P – відсоток використаного матеріалу; S_m – загальна площа матеріалу, м^2; S_i – площа одного типу деталі з надлишком, м^2; n_i - кількість деталей, r - кількість видів деталей.

Другий аспект проблеми оптимізації полягає у виборі найефективнішого методу рішення задачі оптимізації відповідно до вибраного критерію. З метою вирішення задач дискретної оптимізації використовують математичні методи. Суть більшості з цих математичних методів полягає в підборі таких вершин n-гранника рішень задачі, що забезпечують скорочення працемісткості їх пошуку в порівнянні з простим підбором. До найбільш універсальних методів, належать методи цілочисельного і лінійного програмування.

Слід зазначити, що при спрощенні обмежувальних умов і головних функцій, час, необхідний для пошуку оптимального розв’язку, з підвищенням складності розкроювання, зростає в експоненційній залежності. Тому для вирішення задач розкрою доцільно використовувати евристичні чи метаевристичні алгоритми.

Метаевристичні методи є дуже перспективним; їх перевагою є здатність оминати локальні оптимуми. Також вони відображають більш загальний підхід, що може бути застосований до різноманітних оптимізаційних проблем.

Евристичні алгоритми дозволяють знайти прийнятний розв’язок задачі серед багатьох розв’язків протягом певного реального проміжку часу, але не здатні гарантувати того, що цей розв’язок буде найкращим.

Розглянемо особливості застосування евристичних та метаевристичних алгоритмів до вирішення задачі оптимізації розкрою біополімерного матеріалу.

2.2 Обґрунтування вибору алгоритму оптимізації

2.2.1 Алгоритм оптимізації бджолиними колоніями

Алгоритм оптимізації бджолиними колоніями є метаевристичним методом. Більшість метаевристик запозичені з природи для розв'язання складних проблем оптимізації (наприклад, процес відпалу, мурашина колонія, рій частинок, імунна система, рій ос чи колонія бджіл). Серед метаевристичних методів варто виділити ті, що моделюють поведінку колоній агентів (Swarm Intelligence). У природі такий інтелект мають групи суспільних комах: колонії термітів, мурах, бджіл. Метод бджолиної колонії є одним з нових методів оптимізації. Це собою ітеративний мультиагентний метод випадкового пошуку. За рахунок того, що ідея взята з природи (виконується моделювання поведінки бджіл при пошуку нектару) і метод заснований на мультиагентному підході, оптимізаційний процес у роботі методу бджолиної колонії характеризується високою ефективністю [5].

Для вибору нових місць з нектаром декілька сотень бджіл-розвідників відправляються на пошуки. Інші бджоли чекають їхнього повернення до вулика. Коли бджола знайшла таке джерело, вона повертається у вулик і за допомогою спеціального танцю (природа якого й надалі не повністю розгадана) повідомляє іншим бджолам про місце розташування цієї ділянки з нектаром, а також про ймовірні запаси нектару там, що свідчить про якість джерела. Певна частина бджіл полетіла на місце, вказане бджолою-розвідником, за нектаром. Інші вирушили на місце, вказане іншою бджолою.

Таким чином і пошуки джерела, і збір нектару не зосередженні в одній зоні, а охоплюють велику територію з багатьма ймовірними джерелами.

Якщо абстрагуватись від бджіл і перейти до термінів комбінаторної оптимізації, то алгоритм починається з запуску певної кількості агентів, розкиданих випадковим чином по простору розв'язків. Задачу розкрою можна розглядати як задачу комбінаторної оптимізації на перестановках. Кожна перестановка – це точка в комбінаторному просторі. Отже, спочатку випадковим чином обирається кілька точок. Потім обраховується якість місць (значення цільової функції в точці комбінаторного простору), що були знайдені бджолами-розвідниками. Тобто ми визначаємо, яким варіантам розв'язків відповідають кращі значення цільової функції. Після цього, місця з найкращими значеннями цільової функції, обираємо для того, щоб в їхні околи були відправлені інші бджоли (увага зосереджується на ділянках, поблизу яких були знайдені хороші значення цільової функції) для пошуку глобального оптимуму [5].

У програмному продукті це може бути реалізовано дослідженням (підрахунком значення цільової функції) у точках з околу (у нашому випадку окіл – це множина перестановок, що відрізняються від даної на одну транспозицію). Далі, провівши аналіз якості знайдених джерел, ми знову виділяємо серед них найперспективніші. За такою схемою продовжується пошук кращих розв'язків, поки не настане якась з умов зупинки (наближення з певною точністю до точної нижньої межі цільової функції чи коли протягом декількох ітерацій не було покращення цільової функції, або воно було незначним). Щоб не залишитись в якомусь з локальних мінімумів, періодично можна додавати до точок, що досліджується, декілька випадкових.

2.2.2 Генетичний алгоритм

Одним із варіантів для рішення задачі оптимізації процесу розкрою є застосування генетичного алгоритму. При цьому алгоритмі спершу необхідно розглянути критерії та параметри його роботи. Знаходження раціонального

розміщення деталей на матеріалі, що розкроюється має відбуватися за наступних умов: сторони заготовки, що розкроюється, повинні бути паралельними до сторін матеріалу; заготовки, що розкроюються, не повинні перетинатися зі сторонами матеріалу й одна з одною. Схема розміщення розкройок на листах, відповідна до вищенаведених умов, називається упаковкою. Відношення площі упакованих заготовок до площі листів використаного матеріалу або відносні (%) втрати матеріалу характеризують якість упаковки.

Ідея алгоритму розв’язку задачі полягає в наступному. Для оптимізації приймають будь-яку упаковку заготовок. Після цього змінюють порядок розміщення елементів на матеріалі. Під час роботи алгоритму вибирають таку комбінацію заготовок, щоб генерувалася найвища якість упакування. У випадку, коли ширина ряду заготовок більша ширини сегмента, відбувається обертання заготовок допоки вони не помістяться на сегменті. У випадку, коли ширина заготовок більша або дорівнює ширині сегмента, ця заготовка переходить на наступний ряд сегмента. Далі необхідно відсортувати рядки з заготовками в сегменті за мінімумом втрат матеріалу. У випадку, коли втрати сегмента рівні нулю, маніпуляції з ним закінчуються. Цей сегмент переходить на наступний етап. Тобто заготовки, що знаходяться в цьому сегменті, відправляються в шаблон. Це призводить до більш швидкого знаходження результату оптимального розкроювання. Для виконання задачі розкроювання з використанням генетичного алгоритму дані представляють у вигляді генів. Для цього необхідно визначити параметри задачі, які потрібно налаштувати. Після цього вибирають кількість розрядів в кожному гені. Під час цього необхідно врахувати, те що, чим більша кількість розрядів, тим краще, оскільки досягається більша точність, але велика кількість розрядів також підвищує час пошуку розв’язку. Коли обрано параметри, їх розрядність та кількість необхідно визначити, як записувати дані. Опис хромосоми рішення задачі здійснюється декількома способами. Часто пропонують вказувати

заготовку з координатами її розташування. В цьому випадку розв’язок буде подаватися в трьох хромосомах у вигляді координат та кута повороту деталей, що буде ускладнювати процес знаходження рішення [4].

Отже, за дослідженнями останніх років, генетичні алгоритми можуть суттєво впливати на результати розв’язків задач із розміщення та розкроювання.

Для вибору основного алгоритму для програмного забезпечення, було виконано аналіз швидкодії обох алгоритмів, за допомогою декількох експериментів. У таблиці 2.1 наведено результати обчислювального експерименту.

Параметр «складність фігур» визначає рівень складності для аналізу програмою площі та країв фігури. Наприклад, прямокутник чи квадрат мають просту форму та швидко обробляються – вони потрапляють у категорію «прості», однак, овал чи складний багатокутник буде оброблятися довше, тому вони потрапляють у категорію «складні».

Таблиця 2.1 – Результати аналізу швидкодії досліджуваних алгоритмів

Як видно з таблиці, генетичний алгоритм працює набагато стабільніше та швидше майже в усіх випадках. У той же час, алгоритм бджолиних колоній іноді працює набагато швидше за генетичний, але це не стабільно. Тому для подальшого застосування обрано генетичний алгоритм.

Роботу генетичного алгоритму зображено на рисунку 2.1.

Рисунок 2.1 – Блок-схема генетичного алгоритму

Блоки алгоритму:

1) «CheckError» - Спочатку роботи алгоритму перевіряється сумарна площа фігур з надлишком, якщо їх площа більша за площу листа, робота алгоритма закінчується.

2) «CheckCountFigure» - Далі перевіряється наявність деталей, якщо деталі закінчились, робота зупиняється.

3) «getFigure» - Підбирає деталь за площею та кутом повороту для шаблону.

4) «CheckTemplate» - Перевіряє, чи є такий шаблон у списку шаблонів.

5) «FillNewTemplate» - Додає новий шаблон.

6) «FillList» - Заповнює сегменти листа шаблонами деталей.

7) «Statistic» - підраховує відсоток залишків, площі зайнятої деталями та відсоток залишок листа.

Отже, генетичний алгоритм може бути адаптований для вирішення задачі оптимізації розкрою біополімерних матеріалів.

Висновки за розділом 2

Сформульована задача оптимізації для процесу розкрою шкіряного матеріалу, що дозволило узагальнити розглянуті нижче задачі оптимізації та уніфікувати програмний засіб.

Обґрунтовано необхідність застосування генетичних алгоритмів оптимізації для задачі розкрою біополімерного матеріалу та виконано адаптацію генетичного алгоритму з метою використання його в програмному засобі, описаному нижче.

3 СТРУКТУРА ТА ПРИНЦИПИ ПОБУДОВИ EASYCUT

Програмне забезпечення EasyCut було створено у середовищі Qt.

Qt - це вільний інструментарій із віджетами з відкритим кодом для створення графічних інтерфейсів користувача, а також крос-платформних додатків, які працюють на різних програмних та апаратних платформах, таких як Linux, Windows, macOS, Android або вбудована система. Кодова база, поки залишається нативним додатком із власними можливостями та швидкістю. В даний час Qt розробляється компанією Qt, публічно зареєстрованою компанією, та проектом Qt під управлінням з відкритим кодом, за участю окремих розробників та організацій, які працюють над просуванням Qt. Qt доступний як під комерційними ліцензіями, так і з відкритим кодом.

Qt використовується для розробки графічних інтерфейсів користувачів (GUI) та багатоплатформних додатків, які працюють на всіх основних платформах настільних ПК та більшості мобільних або вбудованих платформ. Більшість програм GUI, створених за допомогою Qt, мають натурний інтерфейс, і в цьому випадку Qt класифікується як інструментарій віджетів. Також можуть бути розроблені програми, не-GUI, такі як інструменти командного рядка та консолі для серверів. Прикладом такої програми, яка не є GUI, що використовує Qt, є веб-рамка Cutelyst.

Qt підтримує різні компілятори, включаючи компілятор GCC C ++ та набір Visual Studio і має широку підтримку інтернаціоналізації. Qt також забезпечує Qt Quick, що включає мову декларативного сценарію під назвою QML, що дозволяє використовувати JavaScript для надання логіки. З Qt Quick стала можливою швидка розробка додатків для мобільних пристроїв, в той час як логіка все ще може бути записана з власним кодом, щоб досягти найкращої можливої продуктивності.

Інші функції включають доступ до бази даних SQL, розбір XML, JSON-аналіз, управління потоками та мережева підтримка.

Мовою програмування обраний C++.

C ++ - мова програмування загального призначення, створена Bjarne Stroustrup як розширення мови програмування C, або C з класами. Мова значно розширилася з часом, а сучасний C ++ має об’єктно-орієнтовані, загальні та функціональні особливості на додаток до засобів для маніпулювання низьким рівнем пам'яті. Він майже завжди реалізовується як компільована мова, і багато виробників надають компілятори C ++, включаючи Фонд вільного програмного забезпечення, LLVM, Microsoft, Intel, Oracle та IBM, тому він доступний на багатьох платформах.

C ++ був розроблений з ухилом до системного програмування та вбудованого, обмеженого ресурсами програмного забезпечення та великих систем, що забезпечує його продуктивність, ефективність, та гнучкість використання. C ++ також виявився корисним у багатьох інших контекстах, основними перевагами яких були програмна інфраструктура та обмежені ресурси, включаючи додатки для настільних ПК, сервери (наприклад, електронна комерція, веб-пошук або SQL-сервери) та критично важливі для роботи програми (наприклад, телефон вимикачі або космічні щупи).

C ++ стандартизована Міжнародною організацією зі стандартизації (ISO), остання стандартна версія ратифікована та опублікована ISO у грудні 2017 року як ISO / IEC 14882: 2017 (неофіційно відома як C ++ 17). Мова програмування C ++ була спочатку стандартизована в 1998 році як ISO / IEC 14882: 1998, яка потім була змінена стандартами C ++ 03, C ++ 11 і C ++ 14. Нинішній стандарт C ++ 17 замінює їх новими можливостями та збільшеною стандартною бібліотекою. Перед початковою стандартизацією в 1998 році C ++ був розроблений датським комп'ютерним вченим Б'ярном Струструпом у Bell Labs з 1979 року як розширення мови C; він хотів ефективної та гнучкої мови, подібної до мови C, яка також забезпечувала можливості високого рівня для організації програми. C ++ 20 - наступний плановий стандарт, що відповідає сучасній тенденції нової версії кожні три роки.

Для відображення та роботи інтерфейса з C++ було обрано мову QML.

QML (Qt Meta Language) - мова на основі JavaScript, створена для розробки програм, орієнтованих на користувальницький інтерфейс. Це частина Qt Quick, набору користувальницького інтерфейсу, створеного Digia разом із рамкою Qt. Мова QML використовується в першу чергу для мобільних додатків, де ключове введення, плавна анімація та хороший досвід користувача мають вирішальне значення. Документи QML описують дерево елементів. Елементи QML, які за замовчуванням поставляються із Qt, - це складний набір блоків, графічних елементів (таких як прямокутники чи зображення) та поведінки (наприклад, анімації та переходи). Ці елементи можна комбінувати для створення більш складних компонентів, для завершення програм, підключених до Інтернету.

Приклад цього типу додатків можна знайти на платформі для сенсорних пристроїв Canonical: Ubuntu Phone, де мова QML є одним із стовпів операційної системи. Операційна система Nokia MeeGo також мала підтримку цих програм.

Елементи QML можуть додавати функції, використовуючи код JavaScript, або в одному файлі, або надаючи .js файли. Також QML може мати розширені функції в C ++, використовуючи рамку Qt.

Для задачі сканування матеріалу, було використано «Arduino».

Arduino - компанія з програмним забезпеченням та програмним забезпеченням з відкритим кодом, спільнота проектів та користувачів, яка розробляє та виготовляє одноплановi мікроконтролери та набори мікроконтролерів для побудови цифрових пристроїв. Її продукція ліцензується за Ліцензією загальної публічної ліцензії GNU (LGPL) або General Public License (GPL) GNU, що дозволяє виробляти плати Arduino та розповсюджувати програмне забезпечення будь-ким. Дошки Arduino випускаються у продажу в заздалегідь зібраному вигляді або як комплекти "зробі сам" ("роби сам").

Конструкції плат Arduino використовують різноманітні мікропроцесори та контролери. Плати оснащені наборами цифрових та аналогових штифтів вводу / виводу (вводу / виводу), які можуть поєднуватися з різними розширювальними платами («щитами») або макетними дошками (для прототипування) та іншими схемами. На платах є послідовний інтерфейс комунікації, включаючи універсальну послідовну шину (USB) на деяких моделях, які також використовуються для завантаження програм з персональних комп'ютерів. Мікроконтролери можуть бути запрограмовані за допомогою мов програмування C і C ++. Окрім використання традиційних ланцюжків інструментів компілятора, проект Arduino забезпечує інтегроване середовище розробки (IDE) на основі мовного проекту Processing.

3.1 Структура EasyCut

Програмне забезпечення «EasyCut» складається з декількох модулів:

1) Обробки інформації зі сканера;
2) Введення даних в програму, виводу результатів та збереження даних;
3) Обчислення алгоритмом;
4) Модуль відображення – QML;
5) Модуль запису зі сканера.

В 1 модулі є 3 функції:

- Loop() – працює в циклі, дозволяючи програмі здійснювати обчислення і реагувати на них.
- Setup() – запускається на початку роботи програми та задає стартові параметри.
- Draw() – для запису у файл даних зі сканера.

В 2 модулі є 2 класа:

- Table – базовий клас для таблиць.
- ListOutput – клас для відображення результатів

- Controller

Для оптимізації роботи всього програмного забезпечення та відповідно до принципів ООП було створено клас «Controller» який являється сполучною ланкою між всіма модулями.

В 3 модулі є 1 клас:

- Algorithm – клас, де реалізований алгоритм.

Усі попередні модулі були «Back-end» (тобто сама робота програми) частиною програми. QML модуль є «Front-end» частиною програми, тобто для відображення.

Рисунок 3.1. – Блок-схема «EasyCut»

1 та 5 модуль викликається з основної частини програми, але працює окремо.

Початкове вікно «EasyCut» зображено на рисунку 3.1.

Рисунок 3.1. – Початкове вікно «EasyCut»

3.2 Модуль обробки інформації зі сканера

Перед початком роботи зі сканером «Arduino» необхідно підключити та настроїти цей модуль. Для того щоб запустити модуль «Scan» необхідно в головному вікні програми натиснути на кнопку «Project», та натиснути на кнопку «Scan»(рис.3.2).

Рисунок 3.2. – Відкриття модуля «Scan» в «EasyCut»

Після цього відкривається консольне вікно (рис.3.3).

Рисунок 3.2. – Консольне вікно «Scan» в «EasyCut»

Після цього необхідно вказати порт USB в який підключено «Arduino».

Якщо модуль підключиться успішно буде виведено повідомлення (рис.3.3).

Рисунок 3.3. – З’єднання успішне

Але якщо порт введений невірно, або «Arduino» не працює буде повідомлення (рис.3.4).

Рисунок 3.4. – Помилка

Далі буде виконуватися збереження даних у файл.

Для того щоб зупинити програму необхідно натиснути на клавішу «Enter» (рис.3.5).

Рисунок 3.5. – Зупинення сканування

Після успішної роботи програми буде створений файл «output» в якому збережені дані з сканера.

Після цього вікно консолі «Scan» закриється, і можна буде завантажити файл у програмі «EasyCut» натисненням на кнопку «Open» (рис.3.6).

Рисунок 3.6. – Завантаження файлу

3.3 Модуль введення даних, відображення та збереження

Для введення даних у ПЗ «EasyCut» використовуються таблиці – це зручно для перегляду та редагування інформації (рис 3.2).

Рисунок 3.2 – Таблиця для введення даних

При натисненні на таблицю з’являється меню редагування(рис 3.3)

Рисунок 3.3 – Вікно для редагування даних

Далі потрібно обрати фігуру, яку потрібно буде вирізати на матеріалі. На вибір є Circle (коло), Rectangular (прямокутник) та Polygon (складна фігура).

При натисненні кнопки Polygon з’являється вікно введення точок та розмірів (рис.3.4)

Рисунок 3.4 – Вікно для введення точок

В цьому вікні записуються точки та відстань між ними, а також є лінія між ними дугою чи прямою.

Для введення дуги необхідно оставити галку «Is arc», а також, у напрямку часової стрілки необхідно її вводити чи навпаки.

Наприклад, для введення прямої вводимо 2 точки та не додаємо дугу (рис.3.5)

Рисунок 3.5 – Приклад 1

Тепер другий приклад, де використано дуги, та задані точку для центру дуги та у яку сторону її необхідно вигинати (рис.3.6).

Рисунок 3.6 – Приклад 2

На наступному прикладі показана введена у програму устілка для взуття(рис.3.7)

Рисунок 3.7 – Приклад 3 «устілка»

При натисненні на кнопку Circle (коло) з’являється спеціальне вікно для кола (рис.3.8.)

Рисунок 3.8 – Вікно введення для «Circle»

При натисненні на кнопку Rectangular (прямокутник) з’являється спеціальне вікно для прямокутника (рис.3.9.)

Рисунок 3.9 – Вікно введення для «Rectangular»

Усі введені дані та фігури можна зберегти за допомогою кнопки «Save». Файл зберігається у форматі XML.

Для задання листа матеріалу для розкроювання у нижню таблицю (рис.3.10.) вводимо дані так само як і попереду.

Рисунок 3.10 – Введення параметрів для листу розкроювання

Далі необхідно натиснути на кнопку «Start» та програма розпочне розміщення фігур або деталей на листу.

Після роботи у правому вікні з’являться результати роботи програми (рис.3.11) – на рисунку наведено приклад роботи програми з 2 деталями: устілка (у кількості 100) та ремінь (у кількості 10) на листі 500x500 з невеликим відступом.

Рисунок 3.10 – Приклад повної роботи програми

Цей лист розкрою можна зберегти натисненням на кнопку «Save».

За результатами роботи програми можна подивиться у вікні «Statistic», яке з’являться після натиснення на відповідну кнопку (рис.3.11).

Рисунок 3.11 – Вікно статистики

Після роботи, є можливість додати нові деталі на лист матеріалу. За допомогою кнопки «Accept», підтверджується розкрій деталей на листу.

Після цього, необхідно додати нову деталь, та натиснути на кнопку «Start»(рис.3.12)

Рисунок 3.12 – Додавання деталей

Для надання форми листу матеріалу довільної форми, необхідно у таблиці «Stock» обрати необхідну фігуру, та ввести необхідні дані про фігуру (рис.3.13).

Рисунок 3.13 – Задання форми листу матеріалу

Після додання форми матеріалу, необхідно натиснути на кнопку «Start». Після цього у правому вікні з’являться результати програми (рис.3.14).

Рисунок 3.14 – Приклад роботи ПЗ з листом довільної форми

Важливим є налаштування параметрів. Вони є у спеціальному вікні, яке з’являється після натиснення на кнопку «Settings» (рис.3.15)

Рисунок 3.15 – Вікно налаштувань

«Cut thickness» - Товщина зрізу, задає параметри відступу між деталями.

«Overlap cuts» - Задає параметр, що вказує, чи можливо накладання відступів однієї деталі на іншу.

«Allow part in part» - Задає параметр, що вказує, чи можливо накладати одну деталь в іншу, якщо там є місце для неї.

«Starting corner» - задає параметр, що вказує, з якого кута листа необхідно починати розміщення деталей.

«Optimization» - повзунок, який вказує рівень оптимізації розміщення деталей. На найнижчому рівні, розміщення деталей буде виконуватися з найнижчим рівнем оптимізації – перший вдалий результат розміщення буде прийнятим. На найвищому рівні поки результат не буде найкращим буде виконуватися процес розміщення.

Висновки за розділом 3

Запропоновано структуру та принципи побудови програмного засобу, описано функції та призначення форм програми, наведені системні вимоги для використання представленого програмного засобу. Визначено місце програмного засобу в системі інтелектуальної підтримки підприємства.

Розроблено інструкцію користувача, в якій покроково описано порядок роботи програми для вирішення задачі оптимізації. Розглянуто особливості використання програмного засобу для планування розкрою деталей складних форм.

4 ПРОГРАМА ТА МЕТОДИКА ВИПРОБУВАНЬ EASYCUT

Дана програма і методика випробувань системи розкрою біополімерних матеріалів призначена для оптимізації розкрою біополімерних матеріалів.

Розділ встановлює правила і порядок проведення випробувань і документування для перевірки підготовленості «EasyCut» до експлуатації. Розділ містить переліки перевірок, які необхідно виконати при випробуваннях.

4.1 Об’єкт випробувань

4.1.1 Найменування системи

Система оптимізації розкрою біополімерних матеріалів «EasyCut».

4.1.2 Область застосування системи

Система оптимізації розкрою біополімерних матеріалів «EasyCut» є технологічною інформаційною системою і призначена для процесу розкрою біополімерних матеріалів та для сканування деталей.

Система призначена для введення інформації, її обробки, зберігання результатів, візуалізації даних, які можуть бути використані на етапі планування або вже в процесі відповідного виробництва.

Користувач програми може скористатися функціоналом оптимізації розкрою та сканування деталей.

4.2 Мета випробувань

Метою проведених за цією програмою та методикою випробувань системи оптимізації розкрою біополімерних матеріалів «EasyCut» є визначення функціональної працездатності системи на етапі проведення випробувань.

Програма випробувань повинна засвідчити працездатність системи оптимізації розкрою біополімерних матеріалів «EasyCut» відповідно до функціонального призначення.

4.3 Загальні положення

4.3.1 Перелік керівних документів, на підставі яких проводяться випробування

Приймальні випробування системи оптимізації розкрою біополімерних матеріалів «EasyCut» проводяться на підставі цієї програми і методики приймальних випробувань.

4.3.2 Місце і тривалість випробувань

Місце проведення випробувань – підприємство.

Тривалість випробувань 3 (три) робочі дні.

4.3.3 Перелік пропонованих на випробування документів

Для проведення випробувань Виконавцем пред'являються такі документи:

- Завдання на магістерську дисертацію на створення системи оптимізації розкрою біополімерних матеріалів «EasyCut»;
- Технічний проект системи оптимізації розкрою біополімерних матеріалів «EasyCut».

4.4 Обсяг випробувань

4.4.1 Перелік етапів випробувань і перевірок

В процесі проведення приймальних випробувань повинні бути протестовані наступні підсистеми створення системи оптимізації розкрою біополімерних матеріалів «EasyCut»:

- Введення даних в програму, виводу результатів та збереження даних;
- Обчислення алгоритмом;
- Модуль відображення – QML;

Всі підсистеми випробовуються одночасно на коректність взаємодії підсистем, вплив підсистем один на одного, тобто випробування проводяться комплексно.

Приймальні випробування включають перевірку:

- повноти і якості реалізації функцій;
- виконання кожної вимоги, що відноситься до інтерфейсу Системи;
- роботи користувача в діалоговому режимі;
- повноти дій, доступних користувачеві, і їх достатність для функціонування системи;
- складності процедур діалогу, можливості роботи користувачів без спеціальної підготовки;
- реакції системи на помилки користувача;
- практичної здійсненності рекомендованих процедур.

4.4.1.1 Перелік перевірок при установці

Перелік перевірок при установці наведено у наступній таблиці 4.1.

Таблиця 4.1 – Перелік перевірок

4.4.2 Випробування підсистеми введення даних в програму, виводу результатів та збереження даних

Випробування підсистеми введення та відображення робочих процесів спрямовані на перевірку коректності обробки вхідних даних, зміни показників оптимізації.

Перевіряються процедури і параметри:

- створення деталі у таблиці;
- заповнення полів у таблицях;

- наявність необхідних елементів управління та редагування;

- коректність збереження введених даних і представлення деталей на відображені;

- побудова результатів та збереження їх.

4.4.3 Випробування підсистеми обчислення алгоритмом

Випробування підсистеми обчислення алгоритмом спрямовані на перевірку працездатності та ефективності роботи розрахунку розкрою.

4.4.4 Випробування підсистеми модуля відображення QML;

Випробування підсистеми модуля відображення QML спрямовані на перевірку коректного відображення у інтерфейсі програми всіх частин, також швидкодії реагування програми на змінення даних.

4.5 Методика проведення випробувань

Методика та програма випробувань системи оптимізації розкрою біополімерних матеріалів «EasyCut» передбачає послідовне тестування функцій та підсистем програми. Система розроблена у середовищі Qt на мові програмування C++ та QML.

Головне вікно системи оптимізації розкрою біополімерних матеріалів «EasyCut» показано на рис.4.1.

Рисунок 4.1 – Головне вікно «EasyCut»

За замовчуванням після запуску системи відкривається головне вікно. Якщо порушень немає, усі таблиці та вікно виводу результатів порожні.

Оператор має можливість переглядати дані деталей, додавати нові деталі, редагувати дані деталей та листу розкрою, регулювати параметри розкрою. Наприклад, на рис. 4.2 показано як додається нова деталь, на рис. 4.3 – регулювання параметрів, на рис. 4.4 показано як виглядають таблиці з деталями, а на рис. 4.5 – показано як виводяться результати програми.

Рисунок 4.2 – Відображення вікна додання нової деталі

Рисунок 4.3 – Відображення вікна налаштувань

Рисунок 4.4 – Відображення таблиць

Рисунок 4.5 – Відображення результатів

Після роботи з даними, є можливість перейти до вкладки «Statistic» рис. 4.6. На цій вкладці автоматично проводиться аналіз даних розкрою.

Рисунок 4.6 – Відображення вікна «Statistic»

В цьому вікні можна переглянути відсоток зайнятого матеріалу, відсоток залишків розкрою а також час роботи програми.

Також можна зберегти деталі або лист розкрою у файл. Після натиску кнопки «Save».

Рисунок 4.7 – Відображення вікна «Save»

Користувач може відкрити інформацію про програму «About».

Таким чином, розглянуто основні функції та режими роботи розподіленої системи керування якістю водних об’єктів.

4.6 Вимоги з випробувань програмних засобів

Випробування програмних засобів системи оптимізації розкрою біополімерних матеріалів «EasyCut» проводяться в процесі функціонального тестування Системи і її навантажувального тестування.

Інших вимог щодо випробувань програмних засобів системи оптимізації розкрою біополімерних матеріалів «EasyCut» не пред'являється.

4.7 Перелік робіт, що проводяться після завершення випробувань

За результатами випробувань робиться висновок про відповідність системи оптимізації розкрою біополімерних матеріалів «EasyCut» ТЗ на Систему і можливості здачі системи оптимізації розкрою біополімерних матеріалів «EasyCut» в дослідну експлуатацію. При цьому виробляється (при необхідності) доопрацювання програмних засобів і документації.

4.8 Умови і порядок проведення випробувань

Випробування системи оптимізації розкрою біополімерних матеріалів «EasyCut» повинні проводитися на цільовому обладнанні Замовника. Обладнання повинно бути надано в тій конфігурації, яка запланована для початкового розгортання системи.

Під час випробувань проводиться повне функціональне тестування.

При проведенні приймальних випробувань доступ до Системи надається обмеженому колу користувачів.

Дані користувачі працюють з Системою, виконуючи свої службові обов'язки, тобто розміщують, редагують, регулюють параметри, піддаючи тим самим системи оптимізації розкрою біополімерних матеріалів «EasyCut» повнофункціональному тестуванню протягом встановленого терміну.

4.9 Матеріально-технічне забезпечення випробувань

Приймальні випробування проводяться на програмно-апаратному комплексі Замовника в наступній мінімальній конфігурації:

Робоче місце:

- ПК;
- Операційна система MS Windows 7 або вища;
- Програма Qt.

4.10 Метрологічне забезпечення випробувань

Програма випробувань не вимагає використання спеціалізованого вимірювального обладнання.

Висновки за розділом 4

Розроблена методика та програма випробування програмного засобу, проведена апробація отриманих результатів для розкрою деталей різних форм. За результатами апробації встановлено, що програмний продукт може бути впроваджений на підприємствах, що здійснюють виготовлення виробів зі шкіри.

5 РОЗРОБКА СТАРТАП ПРОЕКТУ

5.1. Резюме: конкретизація бізнес-ідеї, мети стартапу, об'єкту дослідження, місця розробки у інноваційному ланцюжку цінності

Бізнес-ідея: створення компанії, що буде займатися постачанням пристроїв та розробкою програмного забезпечення для них, для компаній які займаються розкроюванням матеріалів.

Метою стартапу є створення компанії, з власною командою, власними спеціалістами, що не відносяться до сфери розробки, для сприянню клієнтом автоматизувати та/або перенести певні власні функціональні можливості до пристроїв, шляхом створення програмного забезпечення.

Назва: Керування розкроюванням матеріалів

Тема: Керування процесом розкроювання матеріалів на підприємствах

Продукт: Пристрій з програмним забезпеченням.

Суб'єкт замовлення: компанії, які мають на меті спростити роботу з розкроюванням матеріалів.

Об'єкт дослідження: наявні напрацювання щодо програмного забезпечення.

Місце розробки у інноваційному ланцюжку цінностей: B2B модель. B2C модель не застосовується, оскільки проект орієнтується на роботі саме з бізнесом.

Кваліфікація персоналу. Розробники мають вищу освіту за напрямом «автоматизація та комп'ютерно інтегровані технології»

Ринок збуту. На початковому етапі створення компанії, в якості ринку збуту розглядається Україна та інші країни пострадянського союзу. В подальшому планується вихід на ринок ЄС та США.

Конкурентні переваги. В наш час існує велика кількість різних компанії, що здатні виконати завдання на замовлення. Однак більшість з них мають такі проблеми, як недостатньо кваліфікований персонал, відсутність спеціалістів в тій чи іншій галузі розробки. Ми пропонуємо повний спектр послуг та маємо на меті якнайшвидше виконати проект.

Вартість розробки. Повний шлях пристрою з програмним забезпеченням від обговорення архітектури до повного впровадження, повної підтримки буде коштувати замовнику близько $5,000 (в розрахунку, що час виконання займе 2 неділі). В залежності від обраних опцій розробки, складності проекту та часу реалізації даного проекту, ціна може корегуватись.

Ринкова ціна. На сьогоднішній день компанії, що займаються розробкою пристроїв, беруть з замовника щонайменше $100,000.

Період повернення капіталовкладень - 0,3 роки.

5.2. Аналіз зовнішнього та внутрішнього середовища стартапу. Ключові фактори успіху. Договір на виконання НДР

Середовище діяльності будь-якого підприємства можна охарактеризувати за допомогою наступної схеми:

Рисунок 5.1 – Зовнішнє і внутрішнє середовище підприємства

Представлена схема є зручною для оцінки потенційних можливостей, що надає зовнішнє середовище, а також загроз, які можна від нього чекати.

Таблиця 5.1. Аналіз загроз і можливостей зовнішнього середовища

Зовнішнє середовище, яке непідконтрольне нашому підприємству, досліджується за загрозами і можливостями: – як вплинуть на нашу діяльність на тому чи іншому ринку сучасні тенденції розвитку науки та техніки, економічної ситуації у світі (країні, регіоні), зміна державної політики та законодавства країн- імпортерів нашої продукції та постачальників [10].

До факторів зовнішнього оперативного середовища відносять конкурентів, постачальників, посередників, споживачів (табл. 5.2).

Таблиця 5.2 – Аналіз факторів зовнішнього оперативного середовища

Внутрішнє середовище підприємства – це його організація, технологічні особливості діяльності, кадри (їх кількісний та якісний склад,спеціалізація, рівень освіти, досвід), забезпеченість основними та оборотними засобами, стан основних засобів (рівень зношеності).

Аналіз внутрішнього середовища підприємства забезпечує визначення сильних та слабких сторін в процесі реалізації стартап-проекту, що саме буде сприяти забезпеченню розробки, впровадженню, а що створюватиме

перешкоди (ризики) в розробці, впровадженні та реалізації ідеї стартап-проекту (табл. 5.3) [1].

Таблиця 5.3 – Переваги і недоліки внутрішнього середовища

На підставі аналізу факторів зовнішнього і зовнішнього оперативного середовищ було визначено ключові фактори успіху мобільного додатку з візуалізації очищення промислових стічних вод. Під ключовими факторами успіху було розглянуто ті, на які підприємство може самостійно впливати під час виробництва і реалізації продукту. Ключові фактори успіху варто надано у вигляді діаграми Шонфільда.

Таблиця 5.4 – Ключові фактори успіху проекту за методом Шонфільда

Таким чином, бачимо, що наш продукт поступається конкурентам за обсягом виробництва та гарантією. Але, за рахунок кращої ціни, високої якості та технічної підтримки наша компанія отримує переваги над конкурентами, що дозволить нам розширювати команду більшою кількістю якісних розробників, що з час збільшить обсяги нашого виробництва та зможе надати додаткові гарантії, що продукт зможе конкурувати з відповідними аналогами.

5.3. Визначення потенційних споживачів

Особливістю розробки стартап-проектів є отримання оцінки потенційного споживача ще на етапі формування інноваційної ідеї. Тому було досліджено потреби потенційних споживачів.

Таблиця 5.5 - Плановий обсяг продукції по місяцям на перший рік

5.4. Ціна інноваційної пропозиції на ринку

Для фінансування нашого стартапу будемо використовувати власні кошти. В собівартість продукціїї закладатимуться усі витрати, а з прибутку формуватиметься фонд розвитку підприємства. Така модель фінансування найбільш прийнятна для нас, оскільки ми маємо достатньо вільних коштів та не хочемо ні з ким ділити прибуток.

Ціноутворення - це процес обґрунтування, затвердження та перегляду цін і тарифів, визначення їх рівня, співвідношення та структури.

Порівняємо ціну за різними методами ціноутворення на ринку.

Основні методи ціноутворення:

Методи ціноутворення, що ґрунтуються на врахуванні витрат називаються витратними.

Ц = С + П,

де Ц – ціна одиниці товару, грн;

С – собівартість одиниці товару, грн;

П – величина прибутку, яку бажає отримати підприємство від реалізації одиниці товару, грн.

Витрати на оплату праці: = 140 000 грн/міс

Амортизація основних фондів: 111 000 грн/міс

Місячна собівартість продукції:

С = 2 836 092грн/міс

Собівартість одиниці продукції: 3 095 652 / 400 = 7 100 грн/од. Очікуваний прибуток з одиниці продукції: 92 900 грн/од..

Отже, за методом повних витрат прогнозована ціна продукту становитиме:

Ц = П + С = 92 900 + 7 100 = 100 000 грн/од.

Головна перевага даного методу – легкість розрахунків. Проте є недоліки. Поперше не береться до уваги чинник попиту на товар, а по-друге ціна, порахована за витратним методом практично завжди завищена.

На виробництві обов’язковими працівниками, які необхідні для виконання відповідного обсягу робіт і повної комплектації робочих місць протягом зміни, є: головний інженер, інженер з апаратного програмного забезпечення, інженер з програмного забезпечення.

Таблиця 5.6 – Персонал компанії

Нарахування ЄСВ: ФОП * 0,22 = 1 680 000 · 0,22 = 369 600

Розрахуємо витрати на оплату праці:

ФОП + Нарахування ЄСВ = 1 680 000 + 369 600 = 2 049 600 грн/рік

Відрахування на соціальні заходи здійснюються за встановленим законодавством ставками від витрат на оплату праці і складає 22%.

Затрати на комплектуючі зручно привести у вигляді таблиці 5.7

Таблиця 5.7 – Розрахунок вартості комплектуючих для виробництва

Річні затрати на комплектуюч: Зс = 100 280 000 грн./рік

Витрати на електроенергію. Розрахуємо витрати на електроенергію за нерегульованим тарифом, тариф за приєднану потужність: Тпр = 2.5 грн/кВт; Потужність обладнання: Ноб = 60 кВт/т; Освітлення цілодобове: Нос = 30 кВт/ добу. Підприємство працює 8 годин на добу, 250 днів на рік. Річні витрати на електроенергію:

$З_{е/е} = П_{пр} \cdot Т_{пр} + Т_{нерег} \cdot (Н_{об} \cdot В_{год} + Н_{ос} \cdot 250) =$

$= 5000 \cdot 2.5 + 1 \cdot (60 \cdot 734{,}864 + 30 \cdot 250) = 64\ 091{,}84$ грн/рік

Витрати на опалення офісу. Загальна площа: 50 $м^2$; тарифна ставка на опалення: 38 грн./м2 міс; Сезон опалення: 6 місяців. Зопал. = 50 · 38 · 6 = 11 400 грн/рік

Амортизаційні відрахування. Здійснюються за прийнятими методами і нормами

Таблиця 5.8 – Розрахунок вартості ОЗ підприємства

Сумарна вартість основних фондів:

ОФ = 120 000 + 150 000 + 50 000 + 150 000 = 470 000 грн./рік

Розраховуємо величину амортизаційних відрахувань:

А = 120 000·0,05 +(150 000 + 50 000 + 150 000)·0,3 = 6000 + 105 000 = 111 000 грн.

Сумарні цехові витрати наведено у таблиці 5.9

Таблиця 5.9 – Сумарні затрати офісу

Ціна реалізації кінцевої продукції , розрахуємо ціну річного випуску продукції:

$В^{грн}_{рік}$ = Ц · $В_{рік}$ = 100 000 · 400 = 40 000 000 грн

П = $В^{грн}_{рік}$ − С = 40 000 000 – 2 836 092 = 37 163 908 грн

Рентабельність підприємства:

$Р = \frac{П}{С} \cdot 100\% = \frac{37163908}{2836092} = 1301{,}4\%$

ОбФ = С − А − ЗП = 2 836 092− 111 000 − 2 049 600= 675 492 грн.

Коефіцієнт економічної ефективності:

$Е = \frac{П}{К} = \frac{П}{ОФ+ОбФ} = \frac{37163908}{470000+675492} = 32{,}4$

Період повернення капіталовкладень:

$Т_{пов} = \frac{1}{Е} = \frac{1}{32{,}4} = 0{,}03$ р.

Фондовіддача основних засобів виробництва:

$$Ф_В = \frac{В^{грн}_{рік}}{ОФ} = \frac{40000000}{470000} = 85{,}1\ грн/грн$$

Фондоємність:

$$ФЄ = \frac{1}{ФВ} = \frac{1}{85{,}1} = 0{,}012$$

Розглянемо метод точки беззбитковості. Це такий метод, при якому підприємець прагне встановити таку ціну, яка забезпечить йому бажану величину чистого прибутку. Це метод вивчення взаємозв'язку між витратами і доходами при різному рівні виробництва, і саме тому він надзвичайно корисний на стадії підготовки й аналізу майбутнього проекту, а також на стадії його реалізації. Рівень беззбитковості по прибутку досягається при такому обсязі реалізації, виручки від якого досить для покриття всіх операційних витрат, включаючи амортизацію; рівень беззбитковості по грошовому потоці може бути отриманий, якщо замінити суму зносу основних активів на суму, необхідну для погашення заборгованості.

П = Ц – С;

Ц = С, звідси П = 0.

Розрахуємо собівартість продукції за витратами, враховуючи витрати на технічне обслуговування обладнання (на технічне обслуговування обладнання ми заклали таку ж суму, яка йде на виготовлення однієї одиниці продукції). На 400 штук ми витрачаємо комплектуючих на 600 000 грн, тобто на одну одиницю продукції ми витрачаємо 1 500 грн.

Позначимо точку беззбитковості через X у штуках. Витрати за рік складатимуть:

470000 +2 049 600+ 64 092 + 11 400 + 1 500*X = 100 000 * X

98 500*X = 2 595 092

X = 26,3.

Точка беззбитковості – 27 шт/рік

Метод встановлення ціни на основі відчутної цінності товару обумовлений специфічними підходами до роботи на ринку. Розрахунок робиться на певну категорію покупців, які погоджуються платити гроші не тільки за вартість товару, а й за комплекс інших послуг: доплати, пов'язані з доставкою, обслуговування тощо. Рівень послуг, що надаються, визначити безумовно важко. Тут доречно вести мову про престижні товари, про особливі послуги, за які за різних обставин і різних умов покупець погоджується платити будь-які гроші. Ось чому для фірми важливо бути надзвичайно чутливою до змін попиту на товар. Тому введемо додаткову ціну за налаштування програмного забезпечення персонально для кожного споживача

$Ц_{дод.}$=3000 грн

$Ц=Ц_{одн}+Ц_{дод}$=100 000+3000=103 000 грн/одн

Таблиця 5.10 – Основні техніко - економічні показники

За знайденими техніко-економічними показниками можна зробити висновок, що дане підприємство є прибутковим

5.5. Концепція бізнес-моделі проекту та карта бізнес-процесів реалізації проекту

Складено карту бізнес-процесів виконання стартап проекту (табл. 5.11).

Таблиця 5.11 – Карта бізнес-процесів виконання стартап проекту

Визначено фактори і елементи бізнес-процесів методом системного аналізу (табл. 5.12).

Таблиця 5.12 – Системний аналіз бізнес-процесів стартапу

5.6. Ризики стартап-проекту та методи управління ними

Стартап як нововведення має багато ризиків на різних стадіях планування, розробки та реалізації проекту. Розглянемо основні ризики, які можуть мати місце для нашого підприємства (табл. 5.13).

Таблиця 5.13 – Страхування на методи управління ризиками

Методи страхування: універсального рецепту від всіх проблем немає, але всі проблеми завжди можна вирішити. Конкуренція, політика, непередбачуваність технологій завжди можна замінити креативним, а також інноваційним підходом.

В нашій справі є один хороший тон, а саме завжди якісно писати код веб-платформи, щоб в будь-який момент можна було розбити дану платформу на декомпозиційні частини і замінити будь-яку стару на будь-яку нову частину. Такий підхід в будь-якій IT сфері приведе до успіху.

До більшості ризиків ми готові, і суттєвих втрат не понесемо. Єдиний реальний ризик – це конкуренція, але завдяки інноваціям і ціні цей стартап буде успішним.

ВИСНОВКИ

1. Проведений огляд науково-технічної літератури показав, що в даний час більшість заводів з виробництва хутряних виробів не використовують автоматизоване проектування схем розкрою, а існуючі програмні продукти не дозволяють розкроювати вироби складної форми.

2. Сформульована задача оптимізації для процесу розкрою шкіряного матеріалу у загальному вигляді, що дозволило узагальнити задачі оптимізації та уніфікувати програмний засіб. Обґрунтовано необхідність застосування генетичних алгоритмів оптимізації для задачі розкрою біополімерного матеріалу та виконано адаптацію генетичного алгоритму для програми.

3. Запропоновано структуру та принципи побудови програмного засобу, описано функції та призначення форм програми, наведені системні вимоги для використання представленого програмного засобу. Визначено місце програмного засобу в системі інтелектуальної підтримки підприємства.

4. Розроблена методика та програма випробування програмного засобу, проведена апробація отриманих результатів для розкрою деталей різних форм. За результатами апробації встановлено, що програмний продукт може бути впроваджений на підприємствах, що здійснюють виготовлення виробів зі шкіри.

5. Сформульовано бізнес-ідею стартап проекту, визначено потенційних споживачів продукту, обґрунтовано ціну інноваційної пропозиції на ринку, сформульовано концепцію бізнес-моделі проекту, складено карту бізнес-процесів реалізації проекту, виконано оцінку ризиків стартап-проекту та запропоновано методи управління ними.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Вікіпедія [Електронний ресурс] : [Веб-сайт]. – Електронні дані. – Режим доступу: https://ru.wikipedia.org/wiki/Кожа_(материал) – Назва з екрана.
2. Производство кожи [Електронний ресурс] : [Веб-сайт]. – Електронні дані. – Режим доступу: http://koja-meh-podklad.ru/proizvodstvo-koji/ – Назва з екрана.
3. Хімія та технології шкір [Електронний ресурс] : [Веб-сайт]. – Електронні дані. – Режим доступу: https://studme.org/99986/tovarovedenie/himiya_i_tehnologiya_kozhi_i_meha – Назва з екрана.
4. Застосування генетичного алгоритму [Електронний ресурс] : [Веб-сайт]. – Електронні дані. – Режим доступу: http://ena.lp.edu.ua:8080/bitstream/ntb/7414/1/21.pdf – Назва з екрана.
5. Розв’язання задачі розкрою алгоритмом оптимізації бджолиною колонією [Електронний ресурс] : [Веб-сайт]. – Електронні дані. – Режим доступу: https://cyberleninka.ru/article/n/rozvyazannya-odnieyi-zadachi-rozkroyu-algoritmom-optimizatsiyi-bdzholinoyu-kolonieyu/viewer – Назва з екрана.
6. Вікіпедія [Електронний ресурс] : [Веб-сайт]. – Електронні дані. – Режим доступу: https://ru.wikipedia.org/wiki/Qt – Назва з екрана.
7. Форум Qt [Електронний ресурс] : [Веб-сайт]. – Електронні дані. – Режим доступу: https://web.archive.org/web/20140325005611/http:/blog.qt.digia.com/blog/2009/08/21/qt-declarative-api-changes/#comment-4727 – Назва з екрана.
8. Форум Qt [Електронний ресурс] : [Веб-сайт]. – Електронні дані. – Режим доступу: http://developer.qt.nokia.com/wiki/GettingStartedQMLRussian – Назва з екрана.
9. Архів форуму Qt [Електронний ресурс] : [Веб-сайт]. – Електронні дані. – Режим доступу:

https://web.archive.org/web/20090518012534/http:/labs.trolltech.com/blogs/category/labs/graphics/kinetic/declarative_ui/ – Назва з екрана.

10. Економічна частина магістерської дисертації: розроблення стартап-проекту: [Електронний ресурс]: навч. посіб. для студ. спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» та спеціальності 161 «Хімічні технології та інженерія» / О.А. Підлісна, Ю.В. Тюленєва ; КПІ ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 0,2 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2019. – 32 с
11. Технология изделий из кожи [Текст]: / Зыбин Ю.П. / «Легкая индустрия» 1975.-464 с.
12. Технология изделий из кожи [Текст] / Фукин В.А., Калита А.Н. /Легпромбытиздат, 1988. –272 с.
13. Конструювання взуття. [Текст] / Лиокумович В.Х. / «Легкая индустрия». 1975, с.169-176.
14. Нормирование расхода обувных материалов. [Текст] / Шустерович М.Л., Зайцева П.С. / Гизлегпром, 1953, - 215 с.
15. Испирян Г.П. Проблема оптимального использования сырья и материалов. [Текст] / Испирян Г.П./ «Легкая индустрия», 1971. –286 с.
16. Проектування процесу виробництва взуття : [Текст] / В. П. Нестеров /НКМ ВС, 1992.- 304
17. Иванов М.Н. и др. Товароведение обувных товаров. - М.: Экономика, 1990, 363 с.
18. Павлин А.В., Мирошников Е.А. Товароведение обувных товаров.: Учебное пособие для товароведных факультетов ВУЗов. - М.: Экономика, 1983, 248с.
19. Павлин А.В., Мирошников Е.А. Товароведение обувных товаров.: Учебное пособие для товароведных факультетов ВУЗов. - М.: Экономика, 1983, 248с.

20. Павлин А.В., Мирошников Е.А. Товароведение обувных товаров.: Учебное пособие для товароведных факультетов ВУЗов. - М.: Экономика, 1983, 248с.
21. Песоцкий В. Украинский рынок обуви. Развитие, проблемы и перспективы, В. Песоцкий , 2007, 24с.
22. Матеріалознавство виробів [Електронний ресурс] URL: http://dn.khnu.km.ua/dn/k_default.aspx?M=k0327&T=04&lng=1&st=0
23. Матеріалознавство виробів [Електронний ресурс] : [Веб-сайт]. – Електронні дані. – Режим доступу: http://dn.khnu.km.ua/dn/k_default.aspx?M=k0327&T=04&lng=1&st=0 – Назва з екрана.
24. Harald Dyckhoff. A typology of cutting and packing problems. / European Journal of Operational Research. – 1990. – Volume 44, Issue 2. – P. 145–159. Johnson D.S. Worst-case performance bounds for simple one-dimensional bin-packing algorithms // Johnson D.S., Demers A., Ullman J. D., Garey M.R.,Graham R.L. – SIAM Journal on Computing – 1974. – P. 299–325. 3. Kryvyy R. Orthogonal placement of different overall components in plane / Rostyslav Kryvyy, Maryan Lobur, Sergiy Tkatchenko // Proc. Of the X-th International Conference TCSET. – Lviv-Slavske, 2010. – P.329.
25. Kryvyy R. Possibilities of the use of patterns in MEMS design // Kryvyy R, Lobur M., Tkatchenko S., Darnobyt Y. // Proceeding XVI Ukrainian-Polish Conference on “CAD in Machinery Design. Implementation and Educational Problems”. – 2008. – P. 45–46.
26. Бабаев Ф.В. Оптимальный раскрой материалов с помощью ЭВМ / Ф.В. Бабаев. – М.: Машиностроение, 1982. – 168 с. 6. Балабанов В.Н., Скобцов Ю.А., Фонотов А.М.
27. Информационное обеспечение в задачах рационального раскроя и упаковки // Наукові праці ДонНТУ. – Вип. 148. – С.119–126. 7. Валеева

А.Ф. Конструктивная эвристика для задачи прямоугольной упаковки / Вестник Башкирского университета. – 2006. – №3. – С. 5–6.

28. Грицюк Ю.І. Проблема моделювання карт і оптимізації плану розкрою плитних деревних матеріалів на меблеві заготовки // Зб. наук.-техн. пр. Науковий Вісник НЛТУ України. – 2006. – Вип. 16.7. – С. 102–110.
29. Джон Х. Холланд. Генетические алгоритмы // В мире науки. – 1992. – №9 – №10. – С. 32–40.
30. Лебедев Б.К. Методы поисковой адаптации в задачах автоматизированного проектирования СБИС: монография. – Таганрог: Изд-во ТРТУ, 2000. – 192 с.
31. Лебедев В. Б. Планирование СБИС методом адаптивного поиска // Перспективные информационные технологии и интеллектуальные системы. – 2000. – №4. – С. 55–65.
32. Лебедев В.Б. Планирование СБИС методом генетического поиска // Перспективные информационные технологии и интеллектуальные системы. – 2000. – №2 – С. 97–105.
33. Мукачева Э.А. Модели и методы расчета раскроя – упаковки геометрических объектов // Э.А. Мукачева, М.А. Верхотуров, В.В. Мартынов. – Уфа: УГАТУ. – 1999. – 217 с.
34. Мукачева А.С. Генетический алгоритм поиска минимума в задачах двумерного гильотинного раскроя // А.С. Мукачева, А. В. Чиглинцев // Информационные технологи. – 2001. – №3. – С. 27–31.
35. Мукачева Э.А. Рациональный раскрой промышленных материалов. Применение АСУ / Э.А. Мукачева. – М.: Машиностроение, 1984. – 176 с. 20. www.optimalprograms.com/RealCut2d.htm.
36. Гэри М. Вычислительные машины и труднорешаемые задачи / М. Гэри, Д. Джонсон. – М. : Изд-во "Мир", 1982. – 416 с.

37. Гуляницкий Л.Ф. Комбинаторный подход к решению одного класса задач размещения / Л.Ф. Гуляницкий, С.А. Малышко. – К., 1988. – 19 с. – (Препр. / АН УССР. Ин-т кибернетики им. В.М. Глушкова, 88-32).
38. Олейник Ал.А. Интеллектуальный метод мультиагентного поиска в многомерном пространстве с использованием прямой святи между агентами / Ал.А. Олейник, С.А. Субботин // Складні системи і процеси. – 2008. – № 2. – С. 102-108.
39. Talbi E. Methaheuristics / E. Talbi. – New Jersey: John Wiley & Sons, Inc. – 2009. – 618 p.

ДОДАТОК А

Методика перевірки комплектності і якості комплекту ПЗ при установці

У випробуванні перевіряється:

1) комплектність програмного забезпечення, наданого Виконавцем для проведення випробувань;

2) якість інсталяції програмного забезпечення на обладнанні, призначеному для проведення випробувань.

Послідовність дій при випробуваннях:

1) аналіз наповнення електронних носів;

2) інсталяція ПЗ.

Учасник робочої групи має виконати інсталяцію ПЗ з інсталяційного диску послідовно виконуючи кроки.

Система витримала випробування, якщо:

1) Комплектність електронних носіїв наступна:

USB-Disk містить:

– папку App з файлами клієнтського ПЗ;

– папку Docs з файлами експлуатаційної документації у форматі pdf.

2) у результаті інсталяції ПЗ на комп‘ютері:

– створено папку C:\ProgramFiles\EasyCut з файлами ПЗ;

– при інсталяції не відбувалось збоїв.

ДОДАТОК Б

Методика перевірки комплектності і якості документації ПЗ

Послідовність дій при перевірці:

1) комплектність експлуатаційної документації системи;

2) якість експлуатаційної документації системи.

Система витримала випробування, якщо:

1) Надано повний комплект експлуатаційної документації;

2) За результатами вивчення документації у Замовника немає претензій до якості у жодному документі наданої документації.

ДОДАТОК В

Програмний код ПЗ «EasyCut»

EasyCut.pro

```
QT       += core gui
DEFINES += QT_DEPRECATED_WARNINGS
DEFINES += PROJECT_PATH="\\\"$$PWD\\\""
CONFIG -= app_bundle
CONFIG += c++1z
CONFIG += qtquickcompiler
CONFIG += qzxing_qml
greaterThan(QT_MAJOR_VERSION, 4): QT += widgets
TARGET = PaintFigure
TEMPLATE = app

SOURCES += main.cpp\
    algorithm.cpp \
    figure.cpp \
    table.cpp \
    controller.cpp \
    frontWindow.qml

HEADERS  +=    figure.h \
    algorithm.h \
    table.h \
    controller.h \

include(platforms/platforms.pri)
```

main.cpp

```
#include "mainwindow.h"
#include <QApplication>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    MainWindow w;
    w.show();

    return a.exec();
}
```

frontWindow.qml

```
import QtQuick 2.12
import QtQuick.Layouts 1.12
import QtQuick.Controls 2.12
import QtGraphicalEffects 1.12

EtPage {
    subTitle: 'EasyCut
    backVisible: true
    onBackClicked: app.popPage()
    hideFooter: true

    function updateEvent(m_table_figure, m_s_table) {
        table_figure = m_table_figure
        s_table = m_s_table
    }

    Flickable {
        id: data_scroll
        anchors.fill: parent
        contentHeight: column.height + 15

        ColumnLayout {
            id: column
            anchors.left: parent.left; anchors.right: parent.right
            anchors.leftMargin: 15; anchors.rightMargin: 15

            Image {
                id: image
                Layout.topMargin: 12
                Layout.fillWidth: true
```

```
    Layout.maximumHeight: 168
    Layout.minimumHeight: 168
    source: imgSrc
    // verticalAlignment: Image.AlignTop
    fillMode: Image.PreserveAspectCrop; autoTransform: true
    sourceSize.width: width; sourceSize.height: height
    z: 1

    layer.enabled: true
    layer.effect: OpacityMask {
        maskSource: Rectangle {
            width: image.width
            height: image.height
            radius: 6

            Rectangle {
                anchors.bottom: parent.bottom
                width: parent.width; height: 10
            }
        }
    }
}

Rectangle {
    id: bottomShadowRect
    Layout.fillWidth: true
    height: 100
    Layout.topMargin: -10
    radius: 6
    color: '#f8f8f8'

    EtText {
```

```
        id: textIm
        anchors.top: parent.top
        anchors.left: parent.left
        anchors.topMargin: 14
        anchors.leftMargin: 12
        text: nameEvent
        font.family: avHeavy.name; font.pixelSize: 16
        font.weight: Font.Medium;
        color: '#3f3f40'
        wrapMode: Text.WordWrap
        lineHeight: 0.85
    }

    EtText {
        anchors.top: textIm.bottom
        anchors.left: parent.left
        anchors.topMargin: 6
        anchors.leftMargin: 12
        text: dateEvent
        font.family: avHeavy.name; font.pixelSize: 12
        font.weight: Font.Medium; color: '#fa5448'
        wrapMode: Text.WordWrap
    }
}

RoundImage {
    Layout.topMargin: 12
    Layout.fillWidth: true
    Layout.maximumHeight: 75
    radius: 5
    visible: false
```

```
LinearGradient {
    anchors.fill: parent
    start: Qt.point(0, 0)
    end: Qt.point(300, 0)
    opacity: 0.2
    gradient: Gradient {
        GradientStop { position: 0.0; color: "black" }
        GradientStop { position: 1.0; color: "transparent" }
    }
 }

EtText {
   text: qsTr('LANG_CAPITALE', “table data”)
   x: 12; y: 14
   font.family: avHeavy.name; font.pixelSize: 12;
   font.weight: Font.Medium
}

EtText {
   text: qsTr('LANG_CAPITALE', “table source”)
   x: 12; y: 14
   font.family: avHeavy.name; font.pixelSize: 12;
   font.weight: Font.Medium
}

Image {
   id: mapPoint
   anchors.right: parent.right; anchors.rightMargin: 51
   source: 'pic/map-point.svg'; y: 23
   sourceSize.height: 25; sourceSize.width: 19
}
```

```
    DropShadow {
        anchors.fill: mapPoint
        source: mapPoint
        horizontalOffset: 0; verticalOffset: 6
        samples: 10
        color: '#50a80b0b'
    }
}

RowLayout {
    Layout.topMargin: 10
    Layout.fillWidth: true
    visible: false

    EtText {
        Layout.fillWidth: true
        text: qsTr('LANG’,”output data”)
        font.family: avMedium.name; font.pixelSize: 14
        font.weight: Font.Medium; color: '#3f3f40'
        verticalAlignment: Text.AlignVCenter
    }

    Row {
        spacing: 5

        Repeater { // temp
            model: 4
            delegate: RoundImage {
                source: 'pic/pictEven' + (index + 1) + '.jpg'
                width: 35; height: 35
                radius: 17.5
            }
```

```
        }

        Rectangle {
            width: 35; height: 35
            radius: 17.5
            color: '#f8f8f8'

            EtText {
                anchors.centerIn: parent
                font.family: avMedium.name; font.pixelSize: 12
                font.weight: Font.Medium
                text: '5+'
                color: '#3f3f40'
            }
        }
    }
}

EtText {
    Layout.topMargin: 19
    text: qsTr('LANG_DESCR', 'DESCRIPTION')
    font.family: avHeavy.name; font.pixelSize: 16
    font.weight: Font.Bold;
}

EtText {
    Layout.topMargin: 10
    Layout.fillWidth: true
    text: mdescrEvent
    font.family: avMedium.name; font.pixelSize: 14
    font.weight: Font.Medium; color: '#3f3f40'
    wrapMode: Text.WordWrap
```

```
        }

        EtButton {
          Layout.topMargin: 27
          Layout.fillWidth: true
          height: 43
          text: qsTr('LANG’, ‘Start')

          onClicked: {
            app.push('contolAlgorithm', { goingName: nameTable, goingLocation: mLocationEvent,
                        goingStart: start, goingEnd: end })
          }
         Row{
          Layout.topMargin: 20
                QTable
                {
                        Id: tables
                        anchorsFill: parent
                        margin: top
                        source: m_tableFigure
                }
         }
        }
      }
    }
  }
```

Controller.h

```
#ifndef CONTROLLER_H
#define CONTROLLER_H

#include <QGraphicsScene>
#include <QGraphicsSceneMouseEvent>
#include "figure.h"

class Controller : public QGraphicsScene
{
   Q_OBJECT
   Q_PROPERTY(int typeFigure
            READ typeFigure WRITE setTypeFigure
            NOTIFY typeFigureChanged)

public:
   explicit PaintScene(QObject *parent = 0);
   ~PaintScene();
  void startOptimize(QList<figure> figures);
   int typeFigure() const;
   void setTypeFigure(const int type);

   enum FigureTypes {
      PolynomType,
      RectangleType,
      CircleType
   };

signals:
   void typeFigureChanged();
```

```
private:
    Figure *tempFigure;
    int m_typeFigure;

private:
    void pressEvent(QGraphicsSceneMouseEvent * event);
    void moveEvent(QGraphicsSceneMouseEvent *event);

};

#endif // CONTROLLER_H
```

controller.cpp

```
#include "controller.h"
Controller:: Controller (QObject *parent) : QGraphicsScene(parent)
{
}

Controller::~ Controller()
{
}

int Controller::typeFigure() const
{
   return m_typeFigure;
}

void Controller::setTypeFigure(const int type)
{
   m_typeFigure = type;
}

void Controller::moveEvent(QGraphicsSceneMouseEvent *event)
{
   tempFigure->setEndPoint(event->scenePos());
   this->update(QRectF(0,0,this->width(), this->height()));
}

void Controller::pressEvent(QGraphicsSceneMouseEvent *event)
{
   switch (m_typeFigure) {
   case RectangleType: {
      Rectangle*item = new Rectangle (event->scenePos());
```

```
        item->setPos(event->pos());
        tempFigure = item;
        break;
    }
    case CircleType: {
        Circle *item = new Circle (event->scenePos());
        item->setPos(event->pos());
        tempFigure = item;
        break;
    }
    case PolynomType: {
        Polynom*item = new Polynom (event->scenePos());
        item->setPos(event->pos());
        tempFigure = item;
        break;
    }
    }
    this->addItem(tempFigure);
}

void Controller::startOptimize()
{
            Algorithm->start(this->getAllFigures());
}
```

**figure.h **

```
#ifndef FIGURE_H
#define FIGURE_H

#include <QObject>
#include <QGraphicsItem>
#include <QDebug>

class Figure : public QObject, public QGraphicsItem
{
   Q_OBJECT
   Q_PROPERTY(QPointF startPoint
            READ startPoint WRITE setStartPoint
            NOTIFY pointChanged)
   Q_PROPERTY(QPointF endPoint
            READ endPoint WRITE setEndPoint
            NOTIFY pointChanged)
public:
   explicit Figure(QPointF point, QObject *parent = 0);
   ~Figure();

   QPointF startPoint() const;
   QPointF endPoint() const;
   Int angle = 0;
   QString color = “”;
   void setStartPoint(const QPointF point);
   void setEndPoint(const QPointF point);

signals:
   void pointChanged();
```

```
private:
    QPointF m_startPoint;
    QPointF m_endPoint;

    QRectF boundingRect() const;

public slots:
    void update();
};

#endif // FIGURE_H
```

figure.cpp

```
#include "figure.h"
#include <QPainter>

Figure::Figure(QPointF point, QObject *parent) :
   QObject(parent), QGraphicsItem()
{
   this->setStartPoint(mapFromScene(point));
   this->setEndPoint(mapFromScene(point));
   connect(this, &Figure::pointChanged, this, &Figure::update);
}

Figure::~Figure()
{
}

QRectF Figure::boundingRect() const
{
   return QRectF((endPoint().x() > startPoint().x() ? startPoint().x() : endPoint().x()) ,
                 (endPoint().y() > startPoint().y() ? startPoint().y() : endPoint().y()) ,
                 qAbs(endPoint().x() - startPoint().x()) + 10,
                 qAbs(endPoint().y() - startPoint().y()) + 10);
}

void Figure::update()
{
   this->update((endPoint().x() > startPoint().x() ? startPoint().x() : endPoint().x()) ,
                (endPoint().y() > startPoint().y() ? startPoint().y() : endPoint().y()) ,
                qAbs(endPoint().x() - startPoint().x()),
                qAbs(endPoint().y() - startPoint().y()) );
```

```
}

void Figure::setStartPoint(const QPointF point)
{
    m_startPoint = mapFromScene(point);
    emit pointChanged();
}

void Figure::setEndPoint(const QPointF point)
{
    m_endPoint = mapFromScene(point);
    emit pointChanged();
}

QPointF Figure::startPoint() const
{
    return m_startPoint;
}

QPointF Figure::endPoint() const
{
    return m_endPoint;
}
```

algorithm.h

```
   #ifndef ALGORITHM_H
#define ALGORITHM_H
#include <QList>
#include <QJsonObject>
#include <QJsonArray>
#include <QJsonDocument>
#include <QByteArray>
#include <QDebug>
#include <QTimer>
#include "controller.h"
class Algorithm : public QObject
{
   Q_OBJECT
private:
   void tempForImages(QVariantMap &map, bool isList);
private:
   QList<figure> figures;
   void connectSignals();
public:
   Algorithm (QObject *parent = nullptr);
   ~Algorithm () = default;
   bool Error() const;
signals:
   // signal for send to qml& controller
   void send(const int &count, const float &percent_waste,
         const float &percent_fill, const QList<figure> figures);
public slots:
   void start(const QList<figure> figures);
};
#endif // ALGORITHM_H
```

algorithm.cpp

```
#include "algorithm.h"

#include <QImage>
#include <QDebug>
#include <QBrush>
#include <QPen>
#include <QPainter>
#include <QImageReader>

Algorithm::Algorithm(QObject *parent)
    : QObject(parent)
{
    connectSignals();
}

void Algorithm::connectSignals()
{
    connect(Controller, &Controller::start, this, &Algorithm::start);
    connect(this, &Algorithm::start, Controller, &Controller::error);
}

bool Algorithm::error() const
{
    return m_error;
}

void Algorithm::tempForImages(QVariantMap &map, bool isList)
{
    QStringList images = map["images"].toStringList();
    BigNumber newImage = getNameForNewFile(images);
```

```
    for (QString &image : images)
    {
        QString originalImage = QUrl(image).toLocalFile();
        QImageReader imageReader(originalImage);
        if (imageReader.format().toLower() == "heic" || imageReader.format().toLower() == "heif")
        {
            imageReader.setAutoTransform(true);
            QImage imageToConvert = imageReader.read();
            QString randomFormat = BigNumber::random(10).toString();
            imageToConvert.save("tmp/" + randomFormat, "JPG", 90);
            if (QImageReader("tmp/" + randomFormat).canRead())
                originalImage = "tmp/" + randomFormat;
        }

        emit send;
        image = newImage.toString();
        newImage++;
    }
    map["images"] = images;
}

void Algorithm::start(QList<figure> _figures)
{
    QList<figure> figures = _figures;
    QSector x(figures.getMax().width,figures.getMax().height());
    for (auto f : figures)
    {
            if(checkS(f.getS(), f.getPoint))
            {
                    fillS(s, f);
                    fillNT(s, f);
```

```
        }
        else
        {
                auto out = checkT(figures);
                fillNT(out);
        }
    }
    emit send(figures);
}
```

table.h

```
#ifndef TABLE_H
#define TABLE_H

#include <QObject>
#include <QGraphicsItem>
#include <QDebug>

class Table : public QObject, public QTable
{
   Q_OBJECT
   Q_PROPERTY
            READ TableColumn WRITE TableColumn
   Q_PROPERTY
            READ TableRow WRITE TableRow
public:
   explicit Table(Table table, QObject *parent = 0);
   ~Table();

#endif // TABLE_H
```

ДОДАТОК Г

Програмний код модуля сканування «Scan»

```
#include "IIOF266USB.h"
#define PERIOD_SCAN 250
long lastBlinkMillis;
boolean ledState;
long lastScanMillis;

void setup()
 {
  Serial.begin(5200);
  Serial.println();

  pinMode(LED_LTIN, OUTPUT);

USB.mode(USB_STA);
  USB.disconnect();
  delay(100);
}
void loop()
{
  long currentMillis = ms();

if (currentMs – lastBlinkMs > PERIOD_SCAN)
  {
    digitalWrite(LED_LTIN, ledState);
    ledState = !ledState;
    lastBlinkMs = currentMs;
  }
  if (currentMs - lastScanMs > SCAN_PERIOD)
```

```
  {
    USB.scanNetworks(true);
    Serial.print("\nScan start ... ");
    lastScanMillis = currentMillis;
  }
  int n = USB.scanComplete();
  if(n >= 0)
  {
    Serial.printf("\n");
    for (int i = 0; i < n; i++)
    {
      Serial.printf("%d: %s, Ch:%d
(%ddBm) %s\n", i+1, USB.SSID(i).c_str(), USB.channel(i), USB.RSSI(i), USB.encryptionType(i) == ENC_
TYPE_NONE ? "open" : "");
    }
    USB.scanDelete();
  }
}
```