МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ТЕХНОЛОГІЙ ТА ДИЗАЙНУ

ФАКУЛЬТЕТ МЕХАТРОНІКИ ТА КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

КВАЛІФІКАЦІЙНА РОБОТА

на тему:

Математичне та програмне забезпечення для інтерактивного проєктування щільних укладок плоских геометричних об'єктів зі складною конфігурацією зовнішнього контуру

Рівень вищої освіти другий (магістерський)
Спеціальність 122 Комп'ютерні науки
Освітня програма Комп'ютерні науки

Виконав: студент групи МгІТ2-22

_____ Антон ОВЧАРОВ

Науковий керівник д.т.н., проф. Віктор ЧУПРИНКА

Рецензент д.ф.-м.н.,проф. Сергій КРАСНИТСЬКИЙ

Київ 2023

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ТЕХНОЛОГІЙ ТА ДИЗАЙНУ

Факультет мехатроніки та комп'ютерних технологій

Кафедра комп'ютерні науки

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки

Освітня програма Комп'ютерні науки

ЗАТВЕРДЖУЮ

Завідувач кафедри КН

_____Володимир ЩЕРБАНЬ

«___»_____2023_____року

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТА

Овчарову Антону Сергійовичу

1.Тема роботи «Математичне та програмне забезпечення для інтерактивного проєктування щільних укладок плоских геометричних об'єктів зі складною конфігурацією зовнішнього контуру»

науковий керівник роботи д.т.н., професор Чупринка Віктор Іванович**,**

затверджені наказом вищого навчального закладу від «12» вересня 2023 року, № 210-уч.

2. Вихідні дані до роботи: розробка кафедри комп'ютерних наук, літературні джерела з автоматизованого проєктування щільних укладок

3. Зміст кваліфікаційної роботи(перелік питань, які потрібно розробити)

Вступ, РОЗДІЛ 1. Загальна постановка задачі щільної решітчастої укладки плоских геометричних об'єктів; РОЗДІЛ 2. Математичне забезпечення для інтерактивного проєктування щільних укладок плоских геометричних об'єктів зі складною конфігурацією зовнішнього контуру; РОЗДІЛ 3. Програмне забезпечення для інтерактивного проєктування щільних укладок плоских геометричних об'єктів зі складною конфігурацією зовнішнього контуру

4. Дата видачі завдання 08.2023р.

5. Консультанти розділів дипломної магістерської роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Вступ	д.т.н., проф. Чупринка В.І.		
Розділ 1	д.т.н., проф. Чупринка В.І.		
Розділ 2	д.т.н., проф. Чупринка В.І.		
Розділ 3	д.т.н., проф. Чупринка В.І.		
Висновки	д.т.н., проф. Чупринка В.І.		

6. Дата видачі завдання_____

№ з/п	Назва етапів дипломної магістерської роботи	Терміни виконання етапів	Примітка про виконання
1	Вступ	05.09. 2023	
2	Розділ 1	20.09.2023	
3	Розділ 2	03.10.2023	
4	Розділ 3	25.10.2023	
5	Висновки	28.10.2023	
6	Оформлення дипломної магістерської роботи (чистовий варіант)	29.11.2023	
7	Здача дипломної магістерської роботи на кафедру для рецензування (за 14 днів до захисту)		
8	Перевірка дипломної магістерської роботи на наявність ознак плагіату (за 10 днів до захисту)		
9	Подання дипломної магістерської роботи у відділ магістратури для перевірки виконання додатку до індивідуального навчального плану (за 10 днів до захисту)		
10	Подання дипломної магістерської роботи на затвердження завідувачу кафедри (з 7 днів до захисту)		

З завданням ознайомлений:

Студент _____ Антон ОВЧАРОВ
(підпис) (ім'я та прізвище)

Науковий керівник роботи _____ Віктор ЧУПРИНКА
(підпис) (ім'я та прізвище)

АНОТАЦІЯ

Овчаров А.С. Математичне та програмне забезпечення для інтерактивного проєктування щільних укладок плоских геометричних об’єктів зі складною конфігурацією зовнішнього контуру – Рукопис.

Дипломна магістерська робота за спеціальністю 122- «Комп’ютерні науки» – Київський національний університет технологій та дизайну, Київ, 2023 рік.

В роботі запропонований методи та алгоритми для інтерактивного проєктування щільних укладок плоских геометричних об’єктів зі складною конфігурацією зовнішнього контуру. Запропоновані алгоритми реалізовані в програмний продукт для інтерактивного проєктування щільних укладок плоских геометричних об’єктів зі складною конфігурацією зовнішнього контуру.

Розроблений програмний продукт має дружній інтерфейс та не потребує спеціальних знань з комп’ютерної техніки для роботи з ним.

Ключові слова: математичне та програмне забезпечення, щільні укладки, плоскі об’єкти

ABSTRACT

Ovcharov A.S. Mathematical and software for interactive design of dense stacks of flat geometric objects with complex external contour configuration - Manuscript.

Master's thesis in specialty 122- "Computer Science" - Kyiv National University of Technology and Design, Kyiv, 2023.

The work proposes methods and algorithms for interactive design of dense stacks of flat geometric objects with a complex configuration of the external contour. The proposed algorithms are implemented in a software product for interactive design of dense stacks of flat geometric objects with a complex external contour configuration.

The developed software product has a friendly interface and does not require special knowledge of computer technology to work with it.

Keywords: mathematical and software, dense stacking, flat objects

Зміст

ВСТУП

Однією з найбільш складних задач технологічної підготовки взуттєвого виробництва є проектування схем розкрою матеріалів на деталі взуття.

Одним з показників технологічності моделі є укладаємість деталей комплекта, яка характеризує щільність їх укладки при суміщенні. Для кожної нової моделі визначається показник укладаємості шляхом побудови так званих модельних шкал. Для кожної деталі комплекта будуються паралелограми суміщення. При цьому використовується прямолінійно – поступальна система розміщення шаблонів. Найчастіше застосовуються такі варіантиукладки:

- без повороту – суміщення в одну сторону;
- з поворотом деталей на 180° однієї відносноіншої;

Для визначення показника укладаємості слід знайти найбільш щільну укладку деталей у паралелограм. Таке суміщення забезпечує мінімальні міжшаблонні відходи при розкрої матеріалу.

Показник укладаємості є величиною, що нормується. Якщо для нової моделі взуття, яка готується до запуску у виробництво, цей показник значно відрізняється від нормативного значення (значно менший), це може означати значне збільшення собівартості моделі взуття, її нерентабельність. Звідси випливає важливість визначення цього показника.

При побудові модельних шкал слід враховувати такі технолочічні умови та обмеження:

- деталі слід розміщувати строго по прямолінійно – поступальній системі;
- слід розглянути різні можливі варіанти суміщення деталей без повороту однієї відносно іншої;
- проглянути варіанти суміщення деталей з поворотом 180° однієї відносно іншої.

Міжшаблонний місток між деталями повинен бути рівним нулю. При побудові паралелограма слід з’єднувати одноіменні точки на шаблонах, які мають однакову орієнтацію, в паралелограм повинно входити ціле число деталей.

Бурний прогрес у створенні нових поколінь ЕОМ та нових технологій програмування значно розширив можливості комп’ютерної техніки. Сучасні візуальні середовища розробки дозволяють створити програмне забезпечення з доступним, простим у користуванні, інтерфейсом користувача. Методи обробки інформації і системи вводу–виводу даних дають можливість без великих труднощів і затрат часу одержати результат, і проглянути його в текстовому чи графічному вигляді.

Мета роботи: використовуючи сучасні можливості обчислювальної техніки, розробити програмне забезпечення для розв’язання задачі пошуку найщільнішої укладки фігур на площині, яку виконано з використанням прямолінійно – поступального руху і повороту фігур на 180°.

Методи дослідження. Теоретичні дослідження роботи грунтуються на комплексі основних положень виробництва взуття та застосування комп'ютерних наук при виришенні поставленої задачі.

Експерементальні дослідження заключалися в тестуванні розробленого продукту для проектування декоративних елементів на деталях взуття.

Наукова новизна полягає у розробці математичного та програмного забезпечення для інтерактивної проєктування щільних укладок плоских геометричних обєктів зі складною конфігурацією зовнішнього контуру.

. **Апробація результатів магістерської роботи.** Основні положення і результати магістерської роботи протягом 2023 року були представлені та одержали позитивну оцінку на міжнародній науковій конференції:

Публікації. За темою магістерської роботи «Розробка математичного та програмного забезпечення для інтерактивної побудови щільних укладок плоских об’єктів» опубліковано одна наукових робота.

1. ЗАГАЛЬНА ПОСТАНОВКА ЗАДАЧІ ЩІЛЬНОЇ РЕШІТЧАТОЇ УКЛАДКИ ПЛОСКИХ ГЕОМЕТРИЧНИХ ОБ’ЄКТІВ

1.1. Основні визначення.

Умовою щільного суміщення деталей являється умова неперетину деталей між собою та дотикання сусідніх деталейю.Деталі не можуть бути повернуті одна відносно іншої на деякий кут. Враховуючи вказане вище, задачу визначення показника укладаємості можна зформулювати так: *Враховуючи перечислені вище технологічні умови і обмеження сумістити деталі так, щоб паралелограм, що характеризує суміщення мав найменшу площу.*

Суміщення, що характеризується паралелограмом з мінімальною площею, при розкрої матеріалу забезпечує отримання мінімальних міжшаблонних нормальних відходів.

При еврістичному пошуку найбільш щільних варіантів суміщення шаблонів, не можна гарантувати, що найкращий варіант не пропущено. В зв’язку з тим, що розглядається обмежена кількість варіантів, хоча теоретично відомо, що можливих варіантів суміщення деталей нескінчено багато. Більш того, результат роботи суттєво залежить від кваліфікації спеціаліста, який її виконує. Тому створення розрахункових теоретично – обгрунтованих методів є актуальною задачою.

Розробка математичної моделі такої задачі включає:

- вибір способу описання конфігурації фігур, що розміщуються;
- аналітичне описання системи розміщення;
- формулювання умов взаємного неперетину фігур;

– конструювання множини допустимих розв’язків;

– аналітичний (або алгоритмічний) запис функції цілі.

В 70-ті роки, коли почали інтенсивно впроваджувати ЕОМ для автоматизації технології підготовки виробництва, було зроблено багато спроб створити математичну модель розглядуваної задачі.

Вже на першому кроці при виборі способу описання геометрії деталей взуття виникли певні труднощі, зв’язані з складною конфігурацією взуттєвих деталей.

Були спроби застосувати аппарат теорії R-функцій для аналітичного описування контурів.

Зважаючи на складність аналітичної перевірки умов взаємного неперетину розміщуваних деталей, були спроби інформацію про деталь подавати у вигляді інформації про сукупність вписаних кругів, які покривають деталь, або вписаних трапецій.

Кожен з перелічених підходів не дав можливості суттєво наблизитися до розв’язання поставленої задачі через труднощі математичного плану при створенні математичної моделі, або неможливості забезпечити достатню точність розв’язку та універсальность розроблених алгоритмів.

Суттєво вдалося змінити ситуацію, коли в кінці 70-х років Ю. Г. Стояном [2] було запропоновано при розробці математичних моделей задач розміщення фігур використовувати спеціальну функцію – вектор-функцію щільного розміщення пари геометричних об’єктів.

Було показано, що для характеристики щільних розміщень фігур, можна з успіхом використовувати значення цієї функції. Більш того, у деяких випадках з’явилася можливість описувати всю сукупність варіантів щільних укладок фігур. Зокрема це стосується укладок, які розглядаються у даній роботі.

Застосування кусково – лінійної апроксимації контурів деталей для кодування інформації про їх форму, дозволило створити математичну модель

розглядуваної задачі, дослідити структуру області допустимих розвязків, та розробити ефективний алгоритм пошуку найкращого варіанту.

Плоска геометрична фігура – це замикання обмеженої однозв'язної області у просторі $R^{(2)}$.

При цьому, головна увага зосереджена на многокутниках, тобто на фігурах, границі яких є замкненими ламаними лініями. Таке звуження класу розглядуваних фігур є виправданим, тому що на практиці будь-яка заготовка може бути представлена як многокутник в межах деякої допустимої похибки.

Далі для зкорочення замість терміна "плоска геометрична фігура", будемо писати просто " фігура".

Розглянемо на площині дві фігури S_1 і S_2. Нехай $intS = S \setminus s$, де s - границя фігури S. Фігури S_1 і S_2 не перетинаються, якщо виконується умова:

$$int\, S_1 \bigcap int S_2 = \varnothing. \tag{1.1}$$

Якщо одночасно виконується умова

$$S_1 \bigcap S_2 \neq \varnothing, \tag{1.2}$$

то фігури S_1 і S_2 називаються розміщеними щільно.

Іншими словами, щільно розміщені фігури не мають спільних внутрішніх, але обов'язково мають спільні граничні точки.

Система фігур S_i, $i = \overline{1, p}$, утворює на площині *укладку*, якщо для кожної пари фігур виконується умова їх взаємного неперетину (1.1).

Позначимо через $S + \vec{a}$ фігуру, яка виникає при перміщенні кожної точки фігури S на вектор $\vec{a}$, і будемо називати її трансляцією фігури S.

Множина векторів

$$\vec{r} = n\vec{a}_1 + m\vec{a}_2, \quad n, m = 0, \pm 1, \pm 2, \ldots, \tag{1.3}$$

називається *решіткою* з базисом $\vec{a}_1, \vec{a}_2$, де $\vec{a}_1, \vec{a}_2$ – лінійно незалежні вектори, і позначається $\Lambda = \Lambda(\vec{a}_1, \vec{a}_2)$.

Абсолютна величина визначника складеного з координат базасних векторів решітки, називається визначником решітки і позначається $\det \Lambda$.

Розглянемо систему фігур

$$\bigcup_{n,m} S_{nm}, \qquad n, m = 0, \pm 1, \pm 2, \ldots, \tag{1.4}$$

яка складається з трансляцій $S_{nm} = S + n\vec{a}_1 + m\vec{a}_2$ фігури S на вектори решітки $\Lambda = \Lambda(\vec{a}_1, \vec{a}_2)$.

Якщо система (1.4) являється укладкою, то така укладка називається *решітчатою укладкою* фігури S, виконаною за решіткою $\Lambda = \Lambda(\vec{a}_1, \vec{a}_2)$. Решітка Λ в цьому випадку називається допустимою для укладки фігури S.

Щільність $\delta_S(\Lambda)$ решітчатої укладки фігури S можна охарактеризувати [8] з допомогою співвідношення

$$\delta_S(\Lambda) = \frac{|S|}{(det\Lambda)}, \tag{1.5}$$

де $|S|$ -площа фігури S; $\det \Lambda$ - визначник решітки $\Lambda = \Lambda(\vec{a}_1, \vec{a}_2)$, за якою виконана укладка.

В подальшому однаковими вважатимуться тільки фігури, які можуть бути переведені одина в іншу з допомогою трансляції на деякий вектор. З цієї точки зору фігура, яка виникає із заданої шляхом повороту на деякий кут, вважається відмінною від неї. Такі фігури в подальшому розглядаються як різні.

Нехай $S(0)$ - фігура з заданою орієнтацією, а $S(\pi)$ - фігура, яку одержано з неї поворотом на кут π. Складемо з цих фігур щільні однорядні розміщення

$$\bigcup_n (S(0)+n\vec{a}), \qquad \bigcup_n (S(\pi)+n\vec{a}), \qquad n=0,\pm1,\pm2,\ldots, \quad .$$

Чередуючи утворені ряди і щільно притискаючи їх, створимо на площині укладку W таким чином, щоб взаїмне розміщення ряду, складеного з фігур $S(0)$, по відношенню до сусідніх рядів із фігур $S(\pi)$, у всій укладці W було однаковим.

Укладка W являється об'єднанням двох решітчатих укладок $\bigcup_{n,m} S(0)+n\vec{a}_1+m\vec{a}_2$ і $\bigcup_{n,m} S(\pi)+n\vec{a}_1+m\vec{a}_2$, $n,m=0,\pm1,\pm2,\ldots,$ виконаних за решітками $\Lambda=\Lambda(\vec{a}_1,\vec{a}_2)$ з однаковим базисом $\vec{a}_1,\vec{a}_2$, де $\vec{a}_1=\vec{a}$, $\vec{a}_2=\vec{g}-\vec{p}$. Тому укладка виду W в проекті названа *подвійною решітчатою укладкою* фігур, які мають початкову орієнтацію і які які повернуті на кут π [9].

Система, яка складається з двох одночасно заданих на площині однакових неспівпадаючих решіток, одна з яких являється трансляцією другої на деякий вектор, називається подвійною решіткою і позначається $W=W(\vec{a}_1,\vec{a}_2,\vec{g})$. Тут $\vec{a}_1,\vec{a}_2$ – базис кожної з решіток системи, $\vec{g}$ - вектор їх взаємного зміщення. Така укладка визначається взаїмним розміщенням чотирьох сусідніх многокутників $S(0), S(0)+\vec{a}, S(\pi)+\vec{g}, S(\pi)+\vec{p}$ і може бути задана трьома векторами $\vec{a},\vec{g},\vec{p}$.

В силу самого способу формування укладки виконуються умови

$$intS(0)_{nm}\cap intS(0)_{kl}=\varnothing; \qquad intS(\pi)_{nm}\cap intS(\pi)_{kl}=\varnothing;$$
$$S(0)_{nm}\cap S(0)_{(n+1)m}\neq\varnothing, \qquad S(\pi)_{nm}\cap S(\pi)_{(n+1)m}\neq\varnothing,$$

(1.6)

$$intS(0)_{nm}\cap intS(\pi)_{kl}=\varnothing, \tag{1.7}$$

де $S(0)_{\mu\nu}=S(0)+\mu\vec{a}_1+\nu\vec{a}_2$; $S(\pi)_{\mu\nu}=S(\pi)+\vec{g}+\mu\vec{a}_1+\nu\vec{a}_2$; $\vec{a}_1=\vec{a}_2$; $\vec{a}_2=\vec{g}-\vec{p}$; $\mu=n,k$; $\nu=m,l$; $n,m,k,l=0,\pm1,\ldots.$ Це означає, що системи

$\bigcup_{n,m} S(0)_{nm}$ і $\bigcup_{n,m} S(\pi)_{nm}$, $n,m = 0,\pm1,\ldots,$ являються решітчатими укладками фігур $S(0)$ і $S(\pi)$. З цього випливає, що три вектори $(\vec{a},\vec{g},\vec{p})$ породжують подвійну решітку $W = W(\vec{a},\vec{g}-\vec{p},\vec{g})$, допустиму для подвійної решітчатої укладки V фігур $S(0)$ і $S(\pi)$ на площині. Множину таких решіток позначимо через $K^{(3)}$.

Позначимо через $det\,W$ визначник, однаковий для обох решіток, які складають подвійну решітку, і будемо називати *визначником* подвійної решітки [9]. Щільність $\delta_S(W)$ подвійної решітчатої укладки W конгруентних фігур $S(0)$ і $S(\pi)$ будемо характеризувати з допомогою співвідношення

$$\delta_S(W) = \frac{2|S|}{det\,W}. \tag{1.8}$$

1.2. Постановка задачі.

Припустимо, що многокутник S решітчасто розміщується на площині.

В дипломному проекті розглядаються подвійні решітчасті укладки многокутників із заданою орієнтацією $S(0)$ і $S(\pi)$ повернутих на кут π і розв'язується задача пошуку найщільнішої серед них.

Зформулюємо вказану задачу більш чітко:

Задача *Серед подвійних решітчатих укладок W многокутників $S(0)$ і $S(\pi)$ знайти таку $W^* = W(\vec{a}_1^{\,*},\vec{a}_2^{\,*},\vec{g}^{\,*})$, щоб щільність $\delta_S(W^*)$ подвійної укладки многокутників $S(0)$ і $S(\pi)$, виконаної за цією решіткою задовільняла співвідношенню*

$$\delta_S(W^*) = \max_W \delta_S(W). \tag{1.9}$$

Враховуючи, що $|S|$ – площа многокутника є сталою, задачу можна зформулювати по іншому:

Серед подвійних решіток $W = W(\vec{a}_1, \vec{a}_2, \vec{g})$*, допустимих для укладки фігур* $S(0)$ *і* $S(\pi)$*, знайти таку* $W^* = W(\vec{a}_1^{\,*}, \vec{a}_2^{\,*}, \vec{g}^{\,*})$*, детермінант якої має мінімальне значення.*

$$detW^* = detW(\vec{a}_1^{\,*}, \vec{a}_2^{\,*}, \vec{g}^{\,*}) = \min detW(\vec{a}_1, \vec{a}_2, \vec{g}), \qquad W \in K^{(3)} \quad (1.10)$$

Задача (1.10) являється задачою математичного програмування, для розв'язання якої необхідно побудувати і описати множину допустимих рішень $K^{(3)}$ і вказати метод пошуку W^* на цій множині. Покажемо, що розв'язання цих питань може бути одержано на основі вектор-функцій щільного розміщення розглядаємих фігур і їх годографів.

Основою для побудови областей допустимих розв'язків у всіх зформульованій задачі являються умови взаємного неперетину фігур в укладці. В зв'язку з цим інформації про форму фігур потрібно отримати інформацію про множину їх взаємних розміщенн, при яких вказані умови виконуються. Така інформація міститься у спеціальній функції – вектор-функції щільного розміщення об'єктів, які впровадженні Ю. Г. Стояном. Із значень відповідних вектор-функцій конструюються в даній роботі трійки векторів, які описують решітки, що складають області допустимих розв'язків розглядуваної в даній роботі задачі.

Розглянемо основні властивості вектор-функції щільного розміщення плоских геометричних фігур і їх годографів, і опишемо алгоритм розрахунку координат вершин годографа вектор – функції щільного розміщення двох різних многокутників, які зберігають постійну взаємну орієнтацію.

1.3.Вектор-функція щільного розміщення

Розглянемо фігури S_1 і S_2, які розміщенні на площині і зберігають постійну взаємну орієнтацію. Через O_1 і O_2 позначимо полюси, зафіксувавши внутрішні точки фігур. $X_1O_1Y_1$ і $X_2O_2Y_2$- координатні системи, зв'язаних відповідно з фігурами S_1 і S_2.

Фігура S_1 нерухомо закріплена на площині, а фігура S_2 – рухається. Нехай друга фігура S_2 рухається навколо першої (S_1) так, щоб при будь-якому їх взаємному положенні контури цих фігур мали спільні точки, тобто фігури дотикалися. Кожне таке положення характеризується вектором $\vec{r}_{12} = \overrightarrow{O_1O_2}$, який з'єднює полюси фігур S_1 і S_2.

Вектор-функцією щільного розміщення фігури S_2 відносно фігури S_1 називається вектор-функція, яка ставить у відповідність кожному щільному розміщенню фігур S_1 і S_2 значення вектора $\vec{r}_{12}$, при умові, що S_1 – нерухома.

Нехай θ – кут, між $\vec{r}_{12}$ і віссю O_1X_1. Тоді вектор-функцією щільного розміщення фігури S_2 відносно фігури S_1 може бути задана у вигляді

$$\vec{r} = \vec{r}_{12}(\theta) \qquad 0 \le \theta \le 2\pi \tag{1.11}$$

Годограф вектора $\vec{r}_{12}$ називається *годографом* вектор-функції щільного розміщення (1.11).

Позначимо через Φ_{12} область, яка обмежена годографом Γ_{12}, і дамо перелік найваждивіших властивостей годографа Γ_{12} і області Φ_{12}.

Форма і розміри області Φ_{12} не залежать від вибору полюсів фігур S_1 і S_2.

Годограф Γ_{12} вектор-функції щільного розміщення фігури S_2 відносно фігури S_1 фактично являється траєкторією руху полюса фігури S_2 при щільному його переміщенні відносно фігури S_1 зі збереженням взаємної

орієнтації. При такому переміщенні фігури S_2 траєкторії всіх його точок однакові. Тому, яка б з точок фігури S_2 не була б вибрана полюсом, годограф Γ_{12} вектора $\vec{r}_{12}$ буде мати одну і ту ж форму і ті ж розміри.

Якщо полюс фігури S_1 перемістити з точки O_1 в точку O_1', $\overrightarrow{O_1O_1'} = \vec{l}_1$, а полюс фігури S_2 перемістити з точки O_2 в точку O_2', $\overrightarrow{O_2O_2'} = \vec{l}_2$, то положення годограф Γ_{12}' вектора $\vec{r}'_{12} = \overrightarrow{O_1'O_2'}$ в координатній системі $X'_1O'_1Y'_1$ буде відрізнятися від положення годографа Γ_{12} вектора $\vec{r}_{12} = \overrightarrow{O_1O_2}$ в координатній системі $X_1O_1Y_1$ зміщенням на вектор $\vec{l}_2 - \vec{l}_1$.

Якщо полюси фігур S_1 і S_2 перемістити на один і той же вектор, то годографи Γ_{12} і Γ_{12}' вектор-функції щільного розміщення фігури S_2 відносно фігури S_1 у відповідних координатних системах будуть розміщені ідентично.

Нехай $S_1 = S_2 = S$, тобто S_1 і S_2 - однакові і однаково орієнтовані фігури, полюси яких вибрані у відповідних точках цих фігур. XOY - координатна система, яка зв'язана з нерухомою із двох фігур S.

Положення годографа Γ вектор-функції щільного розміщення двох конгруентних однаково орієнтоавних фігур в координатній системі XOY не залежить від того, яка пара відповідних точок вибрана полюсами.

Нехай $\vec{r} = \vec{r}_{12}(\theta)$, $0 \le \theta \le 2\pi$,- вектор-функції щільного розміщення фігури S_2 відносно фігури S_1.

Має місце співвідношення $\vec{r}_{21}(\theta) = -\vec{r}_{12}(\theta + \pi)$, яке означає центральну симетрію годографа вектор-функції щільного розміщення двох конгруентних однаково орієнтоавних фігур.

Найважливішими при розв'язанні зформульованих задач являються приведені нижче властивості вектор-функції щільного розміщення Γ_{12} і області Φ_{12}, яка обмежена її годографом.

Якщо полюс O_2 фігури S_2 розміщено у внутрішній точці області Φ_{12}, то фігури S_1 і S_2 мають спільні внутрішні точки (пертинаються).

Якщо полюс O_2 фігури S_2 розміщений поза областю Φ_{12}, то фігури S_1 і S_2 не мають спільні внутрішні точки (не пертинаються).

Якщо полюс O_2 фігури S_2 розміщений на межі області Φ_{12}, то фігури S_1 і S_2 розміщені щільно.

Перелічені властивості проілюстровані на малюнку 1.4.

Зауважимо, що годограф вектор-функції щільного розміщення многокутників являється замкненою ламаною лінією.

Дійсно, переміщення рухомого многокутника відносно нерухомого із збереженням у кожний момент руху дотикання контурів можливо тільки переміщення вершини одного многокутника по стороні іншого або переміщення сторіни одного по стороні іншого (мал 1.5). Траєкторія руху полюса рухомого многокутника фігури складається із відрізків прямих ліній і являється непреривною замкненою ламаною лінією [3].

Якщо m_1 і m_2 - кількість сторін у опуклих многокутників S_1 і S_2, (при чому $m_1 > m_2$), то годограф вектор-функції щільного розміщення одного із них відносно іншого являється ламаною лінією, яка налічує не більше ніж $m_1 + m_2$, але не менше, ніж m_1 ланок.

Розглянемо алгоритм розрахунку кординат вершин годографа вектор-функції щільного розміщення опуклих многокутників.

Нехай S_1 і S_2 - многокутники з полюсами O_1 і O_2. Домовимося додатним вважати такий напрямок обходу вершин фігури, при якому її

внутрішня частина.прилягає до границі зліва. Задамо інформацію про кожен многокутник у вигляді послідовності координат його вершин, які визначені в координатній системі зв'язаній з ним. Позначимо вершини многокутника S_1 через $A_i^{'}(x_i^{'}, y_i^{'}),\ \ i=\overline{1,m_1},$ а вершини многокутника S_2 через $A_j^{''}(x_j^{''}, y_j^{''}),\ \ j=\overline{1,m_2}.$ Припустимо, що многокутник S_1 нерухомий, а многокутник S_2 переміщується відносно S_1 в додатньому напрямку, щільно дотуляючись до нього. Це означає, що в кожний момент многокутники S_1 і S_2 мають спільні граничні точки.

Многокутники не перетинаються, якщо рух многокутника S_1 відбувається таким чином, що S_1 і S_2 знаходяться по різні боки від спільної опорної прямої, напрям якої співпадає із напрямом руху.

Припувстимо, що вершина $A_i^{'}(x_i^{'}, y_i^{'})$ многокутника S_1 суміщена з вершиною $A_j^{''}(x_j^{''}, y_j^{''})$ многокутника S_2 і многокутники S_1 і S_2 не перетинаються. Такому розміщенню многокутників S_1 і S_2 відповідає вершина $M_k(x_k, y_k)$ годографа Γ_{12}, з координатами

$$x_k = x_i^{'} - x_j^{''}, \qquad y_k = y_i^{'} - y_j^{''}. \tag{1.12}$$

Щоб обчислити координати наступної вершини годографа Γ_{12}, необхідно визначити напрямок і величину можливого переміщення $\vec{v}$ многокутника S_2 при умові його щільного розміщення відносно многокутника S_1 в кожний момент руху. Можливі три ситуації (мал 1.5):

– вершина многокутника S_1 рухається по відрізку границі многокутника S_2;
– вершина многокутника S_2 рухається по відрізку границі многокутника S_1;
– відрізок границі многокутника S_1 рухається вздовж границі многокутника S_2.

Введемо функцію:

$$z_{ij} = (x'_{i+1} - x'_i)(y''_{j+1} - y'_j) - (y'_{i+1} - y'_i)(x''_{j+1} - x'_j) \qquad (1.13)$$

Обчислимо її значення, вважаючи, що i та j відповідають суміщеним вершинам многокутників.

Якщо $z_{ij} > 0$, то пряма, на якій розміщений відрізок $A'_i A'_{i+1}$ границі многокутника S_1, являється опорною для обох многокутників, і вони розміщені по різні сторони від неї. Отже $\vec{v} = \overrightarrow{A'_i A'_{i+1}}$.

Якщо $z_{ij} < 0$, то опрною являється пряма, на якій позміщений відрізок $A''_j A''_{j+1}$ границі многокутника S_2 і $\vec{v} = \overrightarrow{A''_{j+1} A''_j}$.

Якщо $z_{ij} = 0$, то обидва відрізка, $A'_i A'_{i+1}$ і $A''_j A''_{j+1}$, лежать на опорній прямій, $\vec{v} = \overrightarrow{A'_i A'_{i+1}} + \overrightarrow{A''_{j+1} A''_j}$.

Координати чергової вершини $M_{k+1}(x_{k+1}, y_{k+1})$ годографа Γ_{12} можна обчислити за формулами

$$x_{k+1} = x_k + v_x, \qquad y_{k+1} = y_k + v_y, \qquad (1.14)$$

де v_x, v_y - проекції вектора $\vec{v}$ на координатні осі.

За початкове можна обрати таке розміщення многокутників S_1 і S_2, при якому вершина многокутника S_1 з максимальною абсцисою суміщена з вершиною многокутника S_2 з мінімальною абсцисою. Розрахунки завершується, коли координати чергової вершини годографа Γ_{12} співпадуть з координатами початкової вершини. В результаті розрахунків буде зформовано масив координат вершин годографа Γ_{12}, $M_k(x_k, y_k)$, $k = \overline{1, N}$, який є основним при подальшій побудові математичної моделі.

Записавши в параметричному вигляді рівняння ланок годографа Γ_{12}:

$$\begin{cases} x = x_k + (x_{k+1} - x_k)\tau_k \\ y = y_k + (y_{k+1} - y_k)\tau_k \end{cases} \quad 0 \le \tau_k \le 1, \quad k = \overline{1, N} \qquad (1.15)$$

отримаємо аналітичний вираз, що описує Γ_{12}.

2. МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ ІНТЕРАКТИВНОГО ПРОЄКТУВАННЯ ЩІЛЬНИХ УКЛАДОК ПЛОСКИХ ГЕОМЕТРИЧНИХ ОБ'ЄКТІВ ЗІ СКЛАДНОЮ КОНФІГУРАЦІЄЮ ЗОВНІШНЬОГО КОНТУРУ

2.1. Допустимі подвійні решітки для укладки многокутників.

Припустимо, що відомі вектор-функція щільного розміщення двох однаково орієнтованих фігур ($S(0)$ або $S(\pi)$),

$$\vec{r} = \vec{r}_{11}(\theta), \quad 0 \le \theta \le \pi, \qquad (2.1)$$

і вектор-функція щільного розміщення фігури $S(\pi)$ відносно фігури $S(0)$

$$\vec{r} = \vec{r}_{12}(\theta), \quad 0 \le \theta \le 2\pi. \qquad (2.2)$$

Позначимо через Γ_{11}, Γ_{12} - годографи вектор-функцій (2.1) і (2.2), а через Φ_{11}, Φ_{12} - області, які обмежені Γ_{11}, Γ_{12}.

Нехай $\vec{a}$ - значення вектор-функції (1.16), а $\vec{g}$ і $\vec{p}$ - значення вектор-функції (2.2), які відповідають точкам її годографа Γ_{12}, розміщеним по різні сторони від прямої $\vec{r} = \vec{a}t, \quad -\infty < t < +\infty,$ і які не належать $int(\Phi_{12} + n\vec{a}),$ $n = \pm 1, \pm 2, \ldots$. Тоді три вектори $(\vec{a}, \vec{g}, \vec{p})$ породжують подвійну решітку $W = W(\vec{a}, \vec{g} - \vec{p}, \vec{g}) \in K^{(3)}$.

Позначимо через $P_1(\vec{a})$ множину значень вектор-функції (1.17), яким відповідають точки годографа Γ_{12}, розміщені ліворуч від прямої $\vec{r} = \vec{a}t,$ $-\infty < t < +\infty,$ і які не належать $int(\Phi_{12} + n\vec{a}),$ $n = \pm 1, \pm 2, \ldots$. А через $P_2(\vec{a})$ - множину значень (2.2), яким відповідають аналогічні точки Γ_{12}, розміщені праворуч від вказаної прямої. Тоді можна вважати, що будь-які три вектори

$(\vec{a}, \vec{g}, \vec{p})$, які породжують подвійну решітку $W = W(\vec{a}, \vec{g} - \vec{p}, \vec{g}) \in K^{(3)}$ такі, що $\vec{g} \in P_1(\vec{a})$, $\vec{p} \in P_2(\vec{a})$. Визначник кожної з решіток дорівнює

$$detW = detW(\vec{a}, \vec{p}, \vec{g}) = \left|\vec{a} \times (\vec{g} - \vec{p})\right| \qquad (2.3)$$

Припустимо, що $S(0)$ - многокутник, і розглянемо задачу (1.6) в цьому випадку.

У відповідності до алгоритму, який вказаний вище, обчислемо координати вершин годографа Γ_{11} вектор-функції щільного розміщення пари однаково орієнтованих многокутників $S(0)$ і годографа Γ_{12} вектор-функції щільного розміщення многокутника $S(\pi)$ відносно многокутника $S(0)$, де $S(\pi)$ – многокутник, одержаний поворотом $S(0)$ на кут π. Годографи вектор-функцій щільного розміщення многокутників є замкненими ламаними лініями.

Пронумеруємо вершини годографів Γ_{11} і Γ_{12} в додатньому напрямку і запишемо рівняння відповідних вектор-функцій щільного розміщення в параметричному вигляді.

$$\vec{r}_1(t_i) = \vec{r}_{1,i} + (\vec{r}_{1,i+1} - \vec{r}_{1,i})t_i, \qquad 0 \le t_i < 1, \qquad i = \overline{1,2n_1}; \qquad (2.4)$$

$$\vec{r}_2(\tau_j) = \vec{r}_{2,j} + (\vec{r}_{2,j+1} - \vec{r}_{2,j})\tau_j, \qquad 0 \le \tau_j < 1, \qquad j = \overline{1,n_2}, \qquad (2.5)$$

де $\vec{r}_{1,i}$, $\vec{r}_{2,j}$ - значення відповідних вектор-функцій, які відповідають вершинам годографів Γ_{11} і Γ_{12}.

Будь-якій подвійній решітчатій укладці фігур $S(0)$ і $S(\pi)$ на площині може бути поставлена у відповідність подвійна решітка $W = W(\vec{a}_1, \vec{a}_2, \vec{g})$ породжена трьома векторами

$$\vec{a} = \vec{r}_1(t_i), \qquad \vec{g} = \vec{r}_2(\tau_j), \qquad \vec{p} = \vec{r}_2(\tau_u), \qquad t_i, \tau_j, \tau_u \in [0,1], \qquad (2.6)$$

при цьому $\vec{a}_1 = \vec{a}$, $\vec{a}_2 = \vec{g} - \vec{p}$ і виконуються умови (1.6) і (1.7).

Розіб'ємо множину $K^{(3)}$ – подвійних решіток, допустимих для укладки $S(0)$ і $S(\pi)$, на класи K_{iju}.

Областю визначення G_{iju} класу K_{iju} є множина точок (t_i, τ_j, τ_u), координати яких є значеннями параметрів відповідної трійки векторів (1.21), яка визначає подвійну решітчату укладку фігур $S(0)$ і $S(\pi)$.

Підсумовуючи сказане вище, зауважимо, що розглядувана задача розпалася на скінчену кількість підзадач, кожна з яких розв'язується у певному класі K_{iju}. Тобто відшукавши оптимальні розв'язки у кожному з класів, складемо зведену таблицю інформації про них; а далі простим порівнянням виберемо розв'язок всієї задачі. Він складається з інформації про допустиму подвійну решітку з мінімальним визначником, серед усіх допустимих решіток. Тобто

$$\det_{W \in K^{(3)}} W^* = \min_{iju} \det W^*_{iju}, \tag{2.7}$$

де

$$\det W^*_{iju} = \min_{(t_i, \tau_j, \tau_u) \in G_{iju}} W_{iju} \tag{2.8}$$

Розв'язання задачі (2.7), (2.8) потребує відповіді на запетання про те, з яких елементів iju складається множина $K^{(3)}$ і що собою являють області G_{iju} для $\forall iju \in K^{(3)}$. Покажемо, як отримати цю інформацію у нашій задачі.

Зазначимо, що вектор $\vec{a}$ може дорівнювати будь-якому значенню вектор-функції (2.5). Однак, враховуючи центральну симетрію останньої і властивості базисів решіток, достатньо розглянути тільки трійки $(\vec{a}, \vec{g}, \vec{p})$, де $\vec{a} \in A_1$,

$$A_1 = \{\vec{a} : \ \vec{a} = \vec{r}_1(t_i), \quad 0 \le t_i \le 1, \quad i = \overline{1, n_1}\} \quad . \tag{2.9}$$

Таким чином, значеннями індекса i являються цілі числа від 1 до n_1.

Якщо t_i зафіксовано і визначено $\vec{a} = \vec{r}_1(t_i)$, то значеннями векторів $\vec{g}$ і $\vec{p}$ можуть бути вибрані значення вектор-функції (2.5), які утворюють множини

$P_1(\vec{a})$ і $P_2(\vec{a})$, $\vec{g} \in P_1(\vec{a})$, $\vec{p} \in P_2(\vec{a})$. Номери ланок Γ_{12}, що належать фрагменту $P_1(\vec{a})$, визначають значення індекса j, а номери ланок Γ_{12}, які належать фрагменту $P_2(\vec{a})$, визначають значення індекса u. Перебираючи трійки iju, отримаємо номери класів з заданим значенням індекса i.

Змінюючи i у межах $\overline{1, n_1}$, і встановлюючи відповідні можливі значення індексів iju, отримаємо показники всіх класів решіток.

Для кожного значення вектора

$$\vec{a} = \vec{r}_1(t_i), \qquad 0 \le t_i < 1$$

фрагменти $P_1(\vec{a})$ і $P_2(\vec{a})$ мають властивість: їх кінцеві точки можна з'єднати вектором, рівним $\vec{a}$, тобто виконуються співвідношення:

$$P_1(\vec{a}): \quad \vec{r}_2(\tau_j) - \vec{r}_2(\tau_k) = \vec{r}_1(t_i), \quad \tau_j \in [0,1) \quad \tau_j \in [0,1) \tag{2.10}$$

$$P_2(\vec{a}): \quad \vec{r}_2(\tau_u) - \vec{r}_2(\tau_v) = \vec{r}_1(t_i), \quad \tau_u \in [0,1) \quad \tau_v \in [0,1) \tag{2.11}$$

$$0 \le t_i < 1$$

Записавши (1.19) у координатній формі, та підставивши (1.18), отримаємо аналітичний вираз для функції цілі у класі K_{iju}:

$$detW(\vec{r}_1(t_i), \vec{r}_2(\tau_j) - \vec{r}_2(\tau_u)) = \begin{vmatrix} x_i^{(1)} + X_i^{(1)} t_i (x_j^{(2)} + X_j^{(2)} \tau_j) - (x_u^{(2)} + X_u^{(2)} \tau_u) \\ y_i^{(1)} + Y_i^{(1)} t_i \ (y_j^{(2)} + Y_j^{(2)} \tau_j) - (y_u^{(2)} + Y_u^{(2)} \tau_u) \end{vmatrix} =$$

$$= t_i \left(\tau_j \begin{vmatrix} X_i^{(1)} & X_j^{(2)} \\ Y_i^{(1)} & Y_j^{(2)} \end{vmatrix} - \tau_u \begin{vmatrix} X_i^{(1)} & X_u^{(2)} \\ Y_i^{(1)} & Y_u^{(2)} \end{vmatrix} \right) + t_i \begin{vmatrix} X_i^{(1)} & x_j^{(2)} - x_u^{(2)} \\ Y_i^{(1)} & y_j^{(2)} - y_u^{(2)} \end{vmatrix} + \tau_j \begin{vmatrix} x_i^{(1)} & X_j^{(2)} \\ y_i^{(1)} & Y_j^{(2)} \end{vmatrix} - \tau_u \begin{vmatrix} x_i^{(1)} & X_u^{(2)} \\ y_i^{(1)} & Y_u^{(2)} \end{vmatrix} +$$

$$+ \begin{vmatrix} x_i^{(1)} & x_j^{(2)} - x_u^{(2)} \\ y_i^{(1)} & y_j^{(2)} - y_u^{(2)} \end{vmatrix} \tag{2.12}$$

де

$x_i^{(1)}, y_i^{(1)}$ – кооординати i-ої вершини годографа Γ_{11};

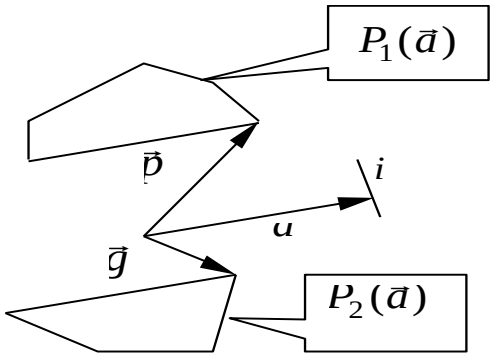
$x_j^{(2)}, y_j^{(2)}$ – кооординати j-ої вершини годографа Γ_{12};

$x_u^{(2)}, y_u^{(2)}$ – кооординати j-ої вершини годографа Γ_{12};

$X_s^{(q)} = x_{s+1}^{(q)} - x_s^{(q)}; Y_s^{(q)} = y_s^{(q)} - y_s^{(q)}$.

Функцуія цілі є квадратичною, бо має у складі члени з $t_i\tau_j$ та $t_i\tau_u$.

Наголосимо, що при укладці опуклих многокутників, опуклими являються замкнені ламані лінії, що є годографами Γ_{11}, Γ_{12}. Тому при кожному зафіксованому t_i в загальному випадку найменше значення має $detW$, коли за вектори $\vec{g}$ і $\vec{p}$ обираються радіуси-вектори кінцевих точок фрагментів (оскільки паралелограми, з яких утворюється основний паралелограм решітки



для подвійної укладки, мають найменші висоти).

Відповідні значення τ_j і τ_u можна визначити з (2.10), (2.11), записавши ці співвідношення у координатному вигляді:

$$\begin{cases} X_j^{(2)}\tau_j - X_k^{(2)}\tau_k = X_i^{(1)}t_i + x_i^{(1)} - x_j^{(2)} + x_k^{(2)} \\ Y_j^{(2)}\tau_j - Y_k^{(2)}\tau_k = Y_i^{(1)}t_i + y_i^{(1)} - y_j^{(2)} + y_k^{(2)} \end{cases}$$

$$\begin{cases} X_u^{(2)}\tau_u - X_v^{(2)}\tau_v = X_i^{(1)}t_i + x_i^{(1)} - {}_u^{(2)} + x_v^{(2)} \\ Y_u^{(2)}\tau_u - Y_v^{(2)}\tau_v = Y_i^{(1)}t_i + y_i^{(1)} - y_u^{(2)} + y_v^{(2)} \end{cases}$$

звідки

$$\tau_j = \frac{1}{\Delta_{jk}}\left(\Delta_{ik} t_i + \Delta_{ijk,k}\right) \tag{2.13}$$

$$\tau_k = \frac{1}{\Delta_{jk}}\left(\Delta_{jk} t_i + \Delta_{ijk,j}\right) \tag{2.14}$$

$$\tau_u = \frac{1}{\Delta_{uv}}\left(\Delta_{iv} t_i + \Delta_{iuv,v}\right) \tag{2.15}$$

$$\tau_v = \frac{1}{\Delta_{uv}}\left(\Delta_{iu} t_i + \Delta_{iuv,u}\right) \tag{2.16}$$

де

$$\Delta_{pq} = \begin{vmatrix} X_p & X_q \\ Y_p & Y_q \end{vmatrix}$$

$$\Delta_{ijk,k} = \begin{vmatrix} x_i^{(1)} - x_j^{(2)} + x_k^{(2)} & X_k^{(2)} \\ y_i^{(1)} - y_j^{(2)} + y_k^{(2)} & Y_k^{(2)} \end{vmatrix} \qquad \Delta_{iuv,v} = \begin{vmatrix} x_i^{(1)} - x_u^{(2)} + x_v^{(2)} & X_v^{(2)} \\ y_i^{(1)} - y_u^{(2)} + y_v^{(2)} & Y_v^{(2)} \end{vmatrix}$$

Підставивши вирази (1.28), (1.30) у вираз для функції цілі (1.27) отримаємо:

$$detW\left(\vec{r}_1(t_i), \vec{r}_2(\tau_j) - \vec{r}_2(\tau_u)\right) = t_i^2 C_{ijkuv}^{(0)} + t_i C_{ijkuv}^{(1)} + C_{ijkuv}^{(2)} \tag{2.17}$$

де

$$C_{ijkuv}^{(0)} = \frac{\Delta_{ik}}{\Delta_{jk}}\begin{vmatrix} X_i & X_j \\ Y_i & Y_j \end{vmatrix} - \frac{\Delta_{iv}}{\Delta_{uv}}\begin{vmatrix} X_i & X_u \\ Y_i & Y_u \end{vmatrix}$$

$$C_{ijkuv}^{(1)} = \frac{\Delta_{ik}}{\Delta_{jk}}\begin{vmatrix} x_i & X_j \\ y_i & Y_j \end{vmatrix} - \frac{\Delta_{iu}}{\Delta_{uv}}\begin{vmatrix} x_i & X_u \\ y_i & Y_u \end{vmatrix} + \begin{vmatrix} X_i & x_j - x_u \\ Y_i & y_j - y_u \end{vmatrix} + \frac{\Delta_{ijk,k}}{\Delta_{jk}}\begin{vmatrix} X_i & X_j \\ Y_i & Y_j \end{vmatrix} - \frac{\Delta_{iuv,v}}{\Delta_{uv}}\begin{vmatrix} X_i & X_u \\ Y_i & Y_u \end{vmatrix}$$

$$C_{ijkuv}^{(2)} = \begin{vmatrix} x_i & x_j - x_u \\ y_i & y_j - y_u \end{vmatrix} + \frac{\Delta_{ijk,k}}{\Delta_{jk}}\begin{vmatrix} x_i & X_j \\ y_i & Y_j \end{vmatrix} - \frac{\Delta_{iuv,v}}{\Delta_{uv}}\begin{vmatrix} x_i & X_u \\ y_i & Y_u \end{vmatrix}$$

Тобто у кожному класі K_{iju} найкращу решітку визначають такі t_i, при яких ця функція набуває мінімального значення на відрізку:

$$max\left(0;-\frac{\Delta_{ijk,k}}{\Delta_{ik}};-\frac{\Delta_{ijk,j}}{\Delta_{ij}};-\frac{\Delta_{iuv,v}}{\Delta_{iv}};-\frac{\Delta_{iuv,u}}{\Delta_{iu}}\right)\leq$$
$$\leq t_i \leq \tag{2.18}$$
$$\leq min\left(1;\frac{\Delta_{jk}-\Delta_{ijk,k}}{\Delta_{ik}};\frac{\Delta_{jk}-\Delta_{ijk,j}}{\Delta_{ij}};\frac{\Delta_{uv}-\Delta_{iuv,v}}{\Delta_{iv}};\frac{\Delta_{uv}-\Delta_{iuv,u}}{\Delta_{iu}};\right)$$

Оскільки функція цілі є квадратичною, то з її властивостей випливає, що на відрізку зміни t_i найменшого значення вона набуває або в кінцевих точках відрізка, або в його внутрішній точці, якщо вона є точкою мінімуму.

Підрахувавши значення в кінцевих точках відрізка зміни t_i та в точці мінімуму, якщо вона належить цьому відрізку, шляхом порівняння знаходимо найкращу решітку у класі K_{iju}.

2.2. Алгоритм пощуку щільних укладок

1. Перевіряємо на опуклість многокутник;
2. Обчислюємо площу многокутника;
3. Розраховуємо координати вершин годографа Γ_{11} вектор-функції щільного розміщення двох одинакових і одинаково орієнтованих многокутників S;
4. Розраховуємо координати вершин годографа Γ_{12} вектор-функції щільного розміщення двох одинакових многокутників $S(0)$ і $S(\pi)$;
5. Задаємо i=1, t_i=0, і обчислюємо стартові значення змінних j, k, u, v, а також стартові значення параметрів τ_j, τ_k, τ_u, τ_v.
6. Знаходимо інтервал допустимих значень параметрів t_i, τ_j, τ_k, τ_u, τ_v, в полі функціонування класу K_{iju}, де $t_i^0 \leq t_i \leq t_i^1$, $\tau_j^0 \leq \tau_j \leq \tau_j^1$, $\tau_k^0 \leq \tau_k \leq \tau_k^1$, $\tau_u^0 \leq \tau_u \leq \tau_u^1$, $\tau_v^0 \leq \tau_v \leq \tau_v^1$.

7. Обчислюємо коефіцієнти цільової функції $detW(\vec{a},\vec{p},\vec{g})$, яка є квадратичною функцією параметра $t_i \in (t_i^0, t_i^1)$, де $t_i^0, t_i^1 \in [0;1]$.
8. Обчислюємо значення параметру t_i^p, яке відповідає критичній точці цільової функції.
9. Обчислюємо значення цільової функції $detW(\vec{a},\vec{p},\vec{g})$ в точках t_i^0, t_i^1, та в точці t_i^p, якщо вона є внутрішньою точкою (t_i^0, t_i^1);
10. Визначаємо екстремум у класі K_{iju}, та заносимо у масив локальних екстремумів відповідні значення *i, j, u,* t_i, τ_j, τ_u, і $detW(\vec{a},\vec{p},\vec{g})$.
11. Перевіряємо $t_i^1 = 1, \tau_j^1 = 1, \tau_k^1 = 1, \tau_u^1 = 1, \tau_v^1 = 1$.
12. Ті з індексів *i, j, k, u, v* для яких відповідне значення параметру τ досягло одиниці, збільшуємо на одиницю, а параметр τ задаємо рівним нулю;
13. Якщо $i \le n$, то переходимо на крок 6, інакше на крок 14; n – кількість вершин многокутника S.
14. В масиві локальних оптимумів, шляхом порівняння значень $detW(\vec{a},\vec{p},\vec{g})$, знаходимо глобальний оптимум і строку параметрів та відповідне значення цільової функції видаємо як розв'язок задачі.

2.3. Визначення площі деталей взуття як площі апроксимуючого випукло-вгнутого многокутника

Аналіз існуючих методів апроксимації плоских геометричних об'єктів показав, що найбільш підходить кусково-лінійний метод апроксимації для опису геометрії взуттєвих деталей та деталей шкіргалантерейних виробів. Тоді інформацію про геометрію плоских геометричних об'єктів будемо подавати у вигляді масивів координат точок апроксимації зовнішньої границі деталей взуття $\{X_k, Y_k\}$, *k=1..n*, де точки $\{X_k, Y_k\}$ - є вершинами опукло-вгнутого многокутника,

апроксимуючого зовнішній контур плоского геометричного об’єкту. Отже площа апроксимуючого многокутника буде дорівнювати:

$$S_n = \left| \iint\limits_{\Omega} dXdY \right| = \frac{1}{2} \left| \oint \left(XdY - YdX \right) \right| = \frac{1}{2} \left| \sum_{k=1}^{n-1} \left(\int\limits_{Y_k}^{Y_{k+1}} XdY - \int\limits_{X_k}^{X_{k+1}} YdX \right) \right|, \qquad (2.19)$$

де $X_n = X_1; Y_n = Y_1$

Так як рівняння прямої, яка проходить через точки (X_k, Y_k) и (X_{k+1}, Y_{k+1}) має вигляд:

$$\frac{X - X_k}{X_{k+1} - X_k} = \frac{Y - Y_k}{Y_{k+1} - Y_k}, \text{ то } Y = \frac{Y_{k+1} - Y_k}{X_{k+1} - X_k} \cdot \left(X - X_k \right) + Y_k \text{ і } dY = \frac{Y_{k+1} - Y_k}{X_{k+1} - X_k} \cdot dX .$$

Підставивши вирази для Y і dY у вираз (2.1) і розрахувавши інтеграли

$$\int\limits_{Y_k}^{Y_{k+1}} XdY = \int\limits_{X_k}^{X_{k+1}} \frac{Y_{k+1} - Y_k}{X_{k+1} - X_k} \cdot dX = \frac{\left(X_{k+1} + X_k \right) \cdot \left(Y_{k+1} - Y_k \right)}{2}$$

$$\int\limits_{X_k}^{X_{k+1}} YdX = \int\limits_{X_k}^{X_{k+1}} \left[\frac{Y_{k+1} - Y_k}{X_{k+1} - X_k} \cdot \left(X - X_k \right) + Y_k \right] \cdot dX = \frac{\left(X_{k+1} - X_k \right) \cdot \left(Y_{k+1} + Y_k \right)}{2}$$

отримаємо

$$S_n = \frac{1}{2} \left| \sum_{k=1}^{n-1} \left(X_k Y_{k+1} - X_{k+1} Y_k \right) \right| \qquad (2.20)$$

Очевидно, що площа плоского геометричного об’єкту може бути наближено визначена як площа апроксимуючого опукло-вгнутого многокутника. При чому точність визначення площі плоского геометричного об’єкту залежить від кількості точок апроксимації. Чім більше точок апроксимації зовнішнього контуру плоского геометричного об’єкту, тим вища точність визначення площі плоского геометричного об’єкту і дійсна площа плоского геометричного об’єкту буде визначатися наступним чином:

$$S = \lim S_n = \lim \frac{1}{2} \left| \sum_{k=1}^{n-1} \left(X_k Y_{k+1} - X_{k+1} Y_k \right) \right| \qquad (2.21)$$

2.4. Контрольний приклад для перевірки працездатності запропонованих алгоритмів

Розрахуємо варіанти укладок для наступного многокутника

Вершини многокутника

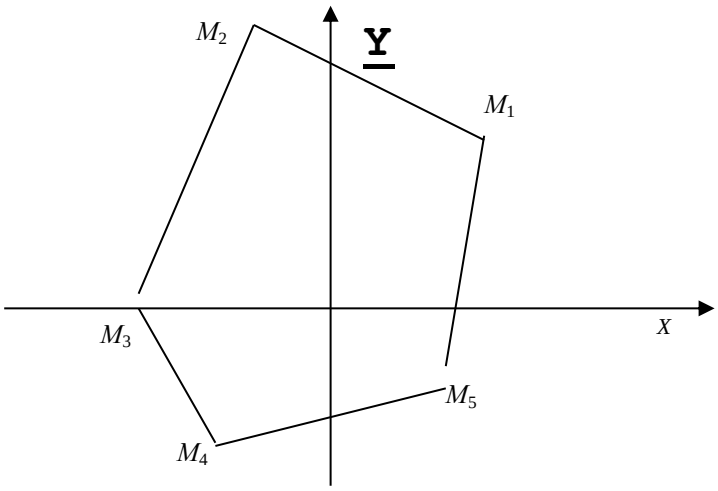
$M_1(4;4)$

$M_2(-2;7)$

$M_3(-5;0)$

$M_4(-3;-4)$

$M_5(3;-2)$



Повернувши заданий многокутник на кут 180°, отримаємо наступний:

Вершини многокутника

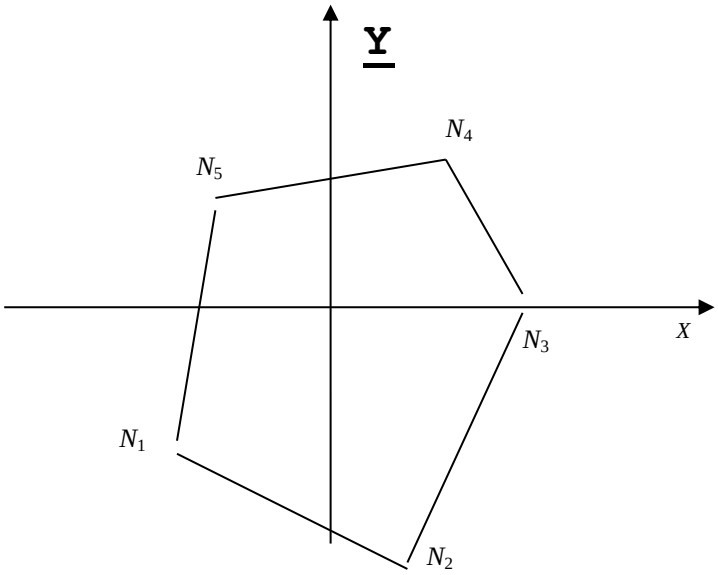
$N_1(-4;-4)$

$N_2(2;-7)$

$N_3(5;0)$

$N_4(3;4)$

$N_5(-3;2)$



На основі алгоритмів, описаних в розрахуємо координати годографів Γ_{11} і Γ_{12}.

Вершини годографа Γ_{11}	Вершини годографа Γ_{12}

$A_1(9;4)$	$B_1(8;8)$
$A_2(7;8)$	$B_2(-4;14)$
$A_3(1;11)$	$B_3(-10;0)$
$A_4(-5;9)$	$B_4(-6;-8)$
$A_5(-8;2)$	$B_5(6;-4)$
$A_6(-9;-4)$	
$A_7(-7;-8)$	
$A_8(-1;-11)$	
$A_9(5;-9)$	
$A_{10}(8;-2)$	

Далі, у відповідності до алгоритму, будемо крок за кроком обчислювати значення змінних *i, j, k, u, v,* та інтервал допустимих значень параметрів t_i, τ_j, τ_k, τ_u, τ_v.

Крок 1.

i=1, j=1, k=2, u=5, v=4

Обчислимо інтервал допустимих значень параметрів t_i, τ_j, τ_k, τ_u, τ_v, але спочатку знайдемо значення всіх необхідних Δ:

$$\Delta_{ij} = \begin{vmatrix} -2 & -12 \\ 4 & 6 \end{vmatrix} = 36 \qquad \Delta_{ik} = \begin{vmatrix} -2 & -6 \\ 4 & -14 \end{vmatrix} = 52 \qquad \Delta_{jk} = \begin{vmatrix} -12 & -6 \\ 6 & -14 \end{vmatrix} = 204$$

$$\Delta_{ijk,j} = \begin{vmatrix} 9-8-4 & -12 \\ 4-8+14 & 6 \end{vmatrix} = 102 \qquad \Delta_{ijk,k} = \begin{vmatrix} 9-8-4 & -6 \\ 4-8+14 & -14 \end{vmatrix} = 102$$

$$\Delta_{iu} = \begin{vmatrix} -2 & 2 \\ 4 & 12 \end{vmatrix} = -32 \qquad \Delta_{iv} = \begin{vmatrix} -2 & 12 \\ 4 & 4 \end{vmatrix} = -56 \qquad \Delta_{uv} = \begin{vmatrix} 2 & 12 \\ 12 & 4 \end{vmatrix} = -136$$

$$\Delta_{iuv,u} = \begin{vmatrix} 9-6-6 & 2 \\ 4+4-8 & 12 \end{vmatrix} = -36 \qquad \Delta_{iuv,v} = \begin{vmatrix} 9-6-6 & 12 \\ 4+4-8 & 4 \end{vmatrix} = -12$$

Підставивши обчислені значення у вирази (2.13) – (1.16) отримаємо:

$$\tau_j = \frac{1}{204}(52t_1 + 102) = \frac{13}{51}t_1 + \frac{1}{2}$$

$$\tau_k = \frac{1}{204}(36t_1 + 102) = \frac{3}{17}t_1 + \frac{1}{2}$$

$$\tau_u = \frac{1}{-136}(-56t_1 - 12) = \frac{7}{17}t_1 + \frac{3}{34}$$

$$\tau_v = \frac{1}{-136}(-32t_1 - 36) = \frac{4}{17}t_1 + \frac{9}{34}$$

Підставимо отримані значення у вираз 2.18, і розв'яжемо систему нерівностей:

$$\begin{cases} 0 \le t_1 \le 1 \\ 0 \le \frac{1}{2} + \frac{13}{51}t_1 \le 1 \\ 0 \le \frac{1}{2} + \frac{3}{17}t_1 \le 1 \\ 0 \le \frac{3}{34} + \frac{7}{17}t_1 \le 1 \\ 0 \le \frac{9}{34} + \frac{4}{17}t_1 \le 1 \end{cases} \Rightarrow \begin{cases} 0 \le t_1 \le 1 \\ -\frac{1}{2}\cdot\frac{51}{13} \le t_1 \le \frac{1}{2}\cdot\frac{51}{13} \\ -\frac{1}{2}\cdot\frac{17}{3} \le t_1 \le \frac{1}{2}\cdot\frac{17}{3} \\ -\frac{3}{34}\cdot\frac{17}{7} \le t_1 \le \frac{31}{34}\cdot\frac{17}{7} \\ -\frac{9}{34}\cdot\frac{17}{4} \le t_1 \le \frac{25}{34}\cdot\frac{17}{4} \end{cases} = \begin{cases} 0 \le t_1 \le 1 \\ -\frac{51}{26} \le t_1 \le \frac{51}{26} \\ -\frac{17}{6} \le t_1 \le \frac{17}{6} \\ -\frac{3}{14} \le t_1 \le \frac{31}{14} \\ -\frac{9}{8} \le t_1 \le \frac{25}{8} \end{cases}$$

$$0 \le t_i \le 1$$

Задавши обчислені границі зміни t_i, знайдемо інтервал значень τ_j, τ_k, τ_u, τ_v.

$$\frac{1}{2} \le \tau_1 \le \frac{77}{102} \quad \Rightarrow \quad 0.5 \le \tau_1 \le 0.7549$$

$$\frac{1}{2} \le \tau_2 \le \frac{23}{34} \quad \Rightarrow \quad 0.5 \le \tau_2 \le 0.6765$$

$$\frac{9}{34} \le \tau_4 \le \frac{1}{2} \quad \Rightarrow \quad 0.2647 \le \tau_4 \le 0.5$$

$$\frac{3}{34} \le \tau_5 \le \frac{1}{2} \quad \Rightarrow \quad 0.0882 \le \tau_5 \le 0.5$$

Тепер, у відповідності до виразу (2.17), обчислимо площу *det W*:

1.1 Обчислимо коефіцієнт укладаємості, задавши параметрам значення лівих границь інтервалу:

Оскільки t_i=0, то площа дорівнює вільномучленові цільової функції

$$detW = C_{ijkuv}^{(2)} = \begin{vmatrix} 9 & 8-6 \\ 4 & 8+4 \end{vmatrix} + \frac{1}{2}\begin{vmatrix} 9 & -12 \\ 4 & 6 \end{vmatrix} - \frac{3}{34}\begin{vmatrix} 9 & 2 \\ 4 & -12 \end{vmatrix} = 100 + 51 - 8.8235 = 142.18$$

$$K_{укл} = \frac{129}{142.18} = 0.9073$$

Всі дані заносимо в таблицю, що знаходиться нище.

1.2 Обчислимо коефіцієнт укладаємості, задавши параметрам значення правих границь інтервалу:

$$C_{ijkuv}^{0} = \frac{13}{51}\begin{vmatrix} -2 & -12 \\ 4 & 6 \end{vmatrix} - \frac{7}{17}\begin{vmatrix} -2 & 2 \\ 4 & 12 \end{vmatrix} = 9.1765 - 13.1765 = 22.353$$

$$C_{ijkuv}^{1} = \frac{13}{51}\begin{vmatrix} 9 & -12 \\ 4 & 6 \end{vmatrix} - \frac{7}{17}\begin{vmatrix} 9 & 2 \\ 4 & 12 \end{vmatrix} + \begin{vmatrix} -2 & 2 \\ 4 & 12 \end{vmatrix} + \frac{1}{2}\begin{vmatrix} -2 & -12 \\ 4 & 6 \end{vmatrix} - \frac{3}{34}\begin{vmatrix} -2 & 2 \\ 4 & 12 \end{vmatrix} = -26.353$$

$$C_{ijkuv}^{2} = \begin{vmatrix} 9 & 2 \\ 4 & 12 \end{vmatrix} + \frac{1}{2}\begin{vmatrix} 9 & -12 \\ 4 & 6 \end{vmatrix} - \frac{3}{34}\begin{vmatrix} 9 & 2 \\ 4 & 12 \end{vmatrix} = 142.1765$$

$$detW = 142.1765 + 1 \cdot (-26.353) + 1^2 \cdot 22.353 = 138.1765$$

$$K_{укл} = \frac{129}{138.1765} = 0.9336$$

Всі дані заносимо в таблицю, що знаходиться нище.

Оскільки t_i досягло значення 1, то збільшуємо *i* на одиницю.

Крок 2.

Проводимо обчислення аналогічні *Кроку*1.

i=2, j=1, k=2, u=5, v=4

$$\Delta_{ij}=\begin{vmatrix}-6 & -12\\ 3 & 6\end{vmatrix}=0 \qquad \Delta_{ik}=\begin{vmatrix}-6 & -6\\ 3 & -14\end{vmatrix}=102 \qquad \Delta_{jk}=\begin{vmatrix}-12 & -6\\ 6 & -14\end{vmatrix}=204$$

$$\Delta_{ijk,j}=\begin{vmatrix}7-8-4 & -12\\ 8-8+14 & 6\end{vmatrix}=138 \qquad \Delta_{ijk,k}=\begin{vmatrix}7-8-4 & -6\\ 8-8+14 & -14\end{vmatrix}=154$$

$$\Delta_{iu}=\begin{vmatrix}-6 & 2\\ 3 & 12\end{vmatrix}=-78 \qquad \Delta_{iv}=\begin{vmatrix}-6 & 12\\ 3 & 4\end{vmatrix}=-60$$

$$\Delta_{uv}=\begin{vmatrix}2 & 12\\ 12 & 4\end{vmatrix}=-136$$

$$\Delta_{iuv,u}=\begin{vmatrix}7-6-6 & 2\\ 8+4-8 & 12\end{vmatrix}=-68 \qquad \Delta_{iuv,v}=\begin{vmatrix}7-6-6 & 12\\ 8+4-8 & 4\end{vmatrix}=-68$$

$$\tau_j=\frac{1}{204}(102t_2+154)=\frac{1}{2}t_2+\frac{77}{102}$$

$$\tau_k=\frac{1}{204}(0t_2+138)=0t_2+\frac{23}{34}$$

$$\tau_u=\frac{1}{-136}(-60t_2-68)=\frac{15}{34}t_2+\frac{17}{34}$$

$$\tau_v=\frac{1}{-136}(-78t_2-68)=\frac{39}{68}t_2+\frac{17}{34}$$

$$\begin{cases}0\le t_1\le 1\\ 0\le \frac{77}{102}+\frac{1}{2}t_2\le 1\\ 0\le \frac{1}{2}+\frac{39}{68}t_2\le 1\\ 0\le \frac{1}{2}+\frac{15}{34}t_2\le 1\end{cases}\Rightarrow\begin{cases}0\le t_1\le 1\\ -\frac{77}{102}\cdot 2\le t_2\le \frac{25}{102}\cdot 2\\ -\frac{1}{2}\cdot\frac{68}{39}\le t_2\le \frac{1}{2}\cdot\frac{68}{39}\\ -\frac{1}{2}\cdot\frac{34}{15}\le t_2\le \frac{1}{2}\cdot\frac{34}{15}\end{cases}=\begin{cases}0\le t_1\le 1\\ -\frac{77}{51}\le t_2\le \frac{25}{51}\\ -\frac{34}{39}\le t_2\le \frac{34}{39}\\ -\frac{17}{15}\le t_2\le \frac{17}{15}\end{cases}$$

$$0 \le t_i \le \frac{25}{51}$$

$$\frac{77}{102} \le \tau_1 \le 1 \quad \Rightarrow \quad 0.7549 \le \tau_1 \le 1$$

$$\tau_2 = \frac{23}{34} = const$$

$$\frac{1}{2} \le \tau_4 \le \frac{2709}{3468} \quad \Rightarrow \quad 0.5 \le \tau_4 \le 0.7811$$

$$\frac{1}{2} \le \tau_5 \le \frac{621}{867} \quad \Rightarrow \quad 0.5 \le \tau_5 \le 0.7162$$

Як бачимо значення лівих границь зміни параметрів укладки на даному кроці дорівнюють значенням правих границь зміни параметрів укладки на попередньому кроці, тому будумо обчислювати коефіцієнт укладаємості тільки для параметрів, що дорівнюють правій границі. Ця закономірність буде виконуватись на всіх наступних кроках.

$$C^0_{ijkuv} = \frac{1}{2}\begin{vmatrix} -6 & -12 \\ 3 & 6 \end{vmatrix} - \frac{15}{34}\begin{vmatrix} -6 & 2 \\ 3 & 12 \end{vmatrix} = 34.4118$$

$$C^1_{ijkuv} = \frac{1}{2}\begin{vmatrix} 7 & -12 \\ 8 & 6 \end{vmatrix} - \frac{15}{34}\begin{vmatrix} 7 & 2 \\ 8 & 12 \end{vmatrix} + \begin{vmatrix} -6 & 2 \\ 3 & 12 \end{vmatrix} + \frac{77}{102}\begin{vmatrix} -6 & -12 \\ 3 & 6 \end{vmatrix} - \frac{1}{2}\begin{vmatrix} -6 & 2 \\ 3 & 12 \end{vmatrix} = 0$$

$$C^2_{ijkuv} = \begin{vmatrix} 7 & 2 \\ 8 & 12 \end{vmatrix} + \frac{77}{102}\begin{vmatrix} 7 & -12 \\ 8 & 6 \end{vmatrix} - \frac{1}{2}\begin{vmatrix} 7 & 2 \\ 8 & 12 \end{vmatrix} = 138.1765$$

$$detW = 138.1765 + \frac{25}{51} \cdot 0 + \left(\frac{25}{51}\right)^2 \cdot 34.4118 = 146.4454$$

$$K_{укл} = \frac{129}{146.4454} = 0.8794$$

Всі дані заносимо у зведену таблицю.

Крок	i	j	k	u	V	τ_i	τ_j	τ_k	τ_u	τ_v	$detW$	$K_{укл}\%$
1.1	1	1	2	5	4	0	0,5	0,5	0,1471	0,2647	142,18	90,73
1.2	1	1	2	5	4	1	0,7549	0,6765	0,5	0,5	138,18	90,36
2	2	1	2	5	4	0,4902	1	0,6765	0,7163	0,7811	146,44	87,94
3	2	2	2	5	4	0,4902	0÷0,32	0,68÷1	0.7163	0.7811	146,44	87,94
4	2	2	3	5	4	0,8718	0,4556	0,3743	0,8846	1	155,73	82,83
5	2	2	3	5	5	0,8718	0,4556	0,3743	0,88÷1	0÷0,12	155,73	82,83
6	2	2	3	1	5	1	0,5	0,5	0,1351	0,1757	145,35	88,75
7	3	2	3	1	5	0,7222	0,8889	1	0,3789	0,3932	151,24	85,29
8	3	2	4	1	5	1	0,8889	0,1388	0,5128	0,5	154,22	83,65
9	4	2	4	1	5	0,2222	1	0,1388	0,5313	0,6453	155,62	82,90
10	4	3	4	1	5	0,7647	0,3487	0,3908	0,6078	1	139,26	92,63
11	4	3	4	1	1	0,7647	0,3487	0,3908	0,61÷1	0÷39	139,26	92,63
12	4	3	4	2	1	1	0,5	0,5	0,1177	0,3922	137,41	93,88
13	5	3	4	2	1	0,8235	1	0,7353	0,4325	0,4810	142,44	90,56
14	5	4	4	2	1	0,8235	0÷0,26	0,74÷1	0,4325	0,4810	142,44	90,56
15	5	4	5	2	1	1	0,2647	0,0882	0,5	0,5	142,18	90,73

3. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ ІНТЕРАКТИВНОГО ПРОЄКТУВАННЯ ЩІЛЬНИХ УКЛАДОК ПЛОСКИХ ГЕОМЕТРИЧНИХ ОБ'ЄКТІВ ЗІ СКЛАДНОЮ КОНФІГУРАЦІЄЮ ЗОВНІШНЬОГО КОНТУРУ

3.1. Вимоги до програмного продукту

Програмний продукт повинен забезпечувати:

- надійну роботу як в локальному, так і в мережевому варіантах;
- високу точність. Будь-які обмеження на кількість внутрішніх контурів і число точок лекальних кривих в конструюванні ведуть, в кінцевому підсумку, до втрати точності при відтворенні складних деталей;
- гнучкість роботи. Як мінімум, повинні бути можливість скасування операцій на будь-яку кількість кроків, можливості введення та редагування будь-якої кількості додаткових точок і інших елементів креслення на будь-якому етапі конструювання. Дуже корисний механізм автоматичних прив'язок до характерних точок лекальні кривих;
- швидкість. Швидка змінюваність моделей, розширення асортиментної бази неможливі без потужного графічного редактора та конструкторського модуля (не плутати з «креслярськими засобами для конструювання»). Сучасний конструкторський модуль повинен забезпечувати виготовлення комплекту лекал для найскладнішої моделі протягом 2-3 годин;
- багато документальний інтерфейс. Сучасні системи дозволяють відкривати відразу кілька моделей при роботі. Вільно і наочно виділяти і переносити з моделі в модель будь-які елементи креслення - будь то лекала або окремі модельні лінії. Без обмеження комбінувати нові моделі на основі наявних;
- роботу з будь-яким серійним обладнанням. Вільний обмін даними з іншими програмами. Крім усього іншого, це полегшить створення єдиної мережі підприємства;
- вивід на друк в будь-якому масштабі на будь-якому етапі роботи.

3.2. Вибір мови програмування для практичної реалізації запропонованих методів та алгоритмів інтерактивної побудови щільних укладок

C++ - це мова програмування, яка мала великий вплив на формування більшості флагманських мов сучасності. Самі ж «плюси», незважаючи на свій вік, досі залишаються затребуваними та позиціонують себе у якості проміжної мови програмування. Це означає, що у C++ можна створювати як low-level (рівень hardware – робота з «залізом»), так і high-level код (створення програмної частини).

Цю мову успішно використовують у створенні високонавантаженого ПЗ, операційних систем, драйверів, комп'ютерних ігор, систем реального часу тощо. «Плюси» важливі скрізь, де на перший план виходять пам'ять та швидкодія.

C++ — мінімалістична мова програмування. Серед її головних цілей: можливість прямолінійної реалізації компіляції, використовуючи відносно простий компілятор, забезпечити низькорівневий доступ до оперативної пам'яті, формувати лише кілька інструкцій машинної мови для кожного елементу мови і не вимагати великої динамічної підтримки. У результаті код C придатний для більшості системного програмного забезпечення, яке традиційно писали асемблером.

Попри її низькорівневі можливості, мову проєктували для платформонезалежного програмування. Сумісна зі стандартами та платформонезалежна написана мовою C++ програма може легко компілюватися на великій кількості апаратних платформ та операційних систем з мінімальними змінами. Мова стала доступною для великої кількості платформ — від вбудованих мікроконтролерів до суперкомп'ютерів.

Переваги мови С++:

- **Швидкодія**. Швидкість роботи програм на С++ практично не поступається програмам на С, хоча програмісти отримали в свої руки нові можливості і нові засоби.
- **Масштабованість**. На мові С++ розробляють програми для найрізноманітніших платформ і систем.
- Можливість роботи на **низькому рівні** з пам'яттю, адресами, портами. (Що, при необережному використанні, може легко перетворитися на недолік.)
- Можливість створення **узагальнених алгоритмів** для різних типів даних, їхня спеціалізація, і обчислення на етапі компіляції, з використанням шаблонів.
- Підтримуються різні стилі та технології програмування, включаючи традиційне директивне програмування, ООП, узагальнене програмування, метапрограмування (шаблони, макроси).

3.3. Опис основних процедур розробленого програмного продукту, які забезпечують інтеративне проєктування щільних укладок для плоских об’єктів зі складною конфігурацією зовнішнього контуру

Процедура *TForm1.Open1Click* забезпечує введення інформації про зовнішні контури деталей моделі взуття для яких необхідно інтеративно побудовати щільні укладки:

Функція *void __fastcall TForm1::Open1Click* забезчує зчитується з файла **.bmp* малюнка. Визначає довжину DlIm та ширину ShIm малюнка в пікселях та роздільну здатність RzDl по вісі OX та RzSh по вісі OУ для цього малюнка.

```
void __fastcall TForm1::Open1Click(TObject *Sender)
{
unsigned int A,B,C,D;
```

```
int i,L,DlIm, ShIm,RzDl,RzSh;
float Xr[100],Yr[100];
char  Fname[100],T;
AnsiString NameF;

if (OpenPictureDialog1->Execute())
  Image1->Picture->LoadFromFile(OpenPictureDialog1->FileName);
  NameF=OpenPictureDialog1->FileName;
   for (i=0;i<99;i++)Fname[i]=NULL;
//   ifstream f;
   L=NameF.Length();
  for(i=1; i<=L;i++)
   {
    T=NameF[i];
    Fname[i-1]=T;
   }
   f.open(Fname,ios::in);
    DlIm=Val(18);
    ShIm=Val(22);
    RzDl=Val(38);
    RzSh=Val(42);
   f.close();
   mx=1000./RzDl;
   my=1000./RzSh;
   Image1->Width=DlIm;
   Image1->Height=ShIm;
}
 //---------------------------------------------------------------------------
```

Функція *int Val(int k)* забезпечує у файлі прямого доступу зчитування інфомації із послідовних чотирьох байтів, починаючи з *k* –го байту та представлення отриманого числа із цих чотирьох байтів у змінній *int Q*.

```
  int Val(int k)
{ int Q;
   unsigned int A,B,C,D;
   f.seekg(k);
   A=0;B=0;C=0;D=0;
   f.read((char*)&A,1);
   f.read((char*)&B,1);
   f.read((char*)&C,1);
   f.read((char*)&D,1);
   Q=A+256*B+256*256*C+256*256*256*D;
   return Q;
  }

 //---------------------------------------------------------------------------
```

Функція *void __fastcall TForm1::Image1MouseDown(TObject *Sender, TMouseButton Button, TShiftState Shift, int X, int Y)* забезпечує інтерактивне введення інформації про зовнішні контури плоских геометричних об\єктів у вигляді координат вершин многокутників *Xd[p][i], Yd[p][i],* де *p* - порядковий номер плоского об/єкту, *i* - порядковий номер вершини цього об’єкту. Ліва кнопка миші вводить координати многокутника, права - виключає, середня - замикає контур.

```
void __fastcall TForm1::Image1MouseDown(TObject *Sender,
      TMouseButton Button, TShiftState Shift, int X, int Y)
{
 int Xr,Yr;
```

```
Image1->Canvas->Pen->Mode=pmXor;
Image1->Canvas->Pen->Color=clGreen;
Image1->Canvas->Pen->Width=2;
if (Button==mbLeft)
 {
  Xd[p][i]=X*mx; Xt[i]=X;
  Yd[p][i]=-Y*my; Yt[i]=Y;
  if (i==0)Image1->Canvas->MoveTo(Xt[i],Yt[i]);
  else Image1->Canvas->LineTo(Xt[i],Yt[i]);
  i++;
 }
else
 if (Button==mbRight)
  {
   if (i>0)
    {
     i--;

     Image1->Canvas->LineTo(Xt[i],Yt[i]);
    }
  }
 else
  {
   // i++;
    Xd[p][i]=Xd[p][0];
    Yd[p][i]=Yd[p][0];
    Image1->Canvas->LineTo(Xt[0],Yt[0]);
    p=p+1;
```

```
if (p==1)  Model="AAAAA";
NameDet[p-1]=IntToStr(p);
KilksPointDet[p-1]=i+1;
KilDet=p;
i=0;
}
```

Функція *GraphIm3* забезпечує вивід плоского геометричного об'єкту з кількістю вершин *int n* та координатами вершин *float X[], float Y[],* координатами центру прямокутника описаного навколо цього об'єкту *float Xcf, float Ycf,* координатами центру *Image3 float Xce, float Yce,* в масштабі *float mxy,* кольором *int q,* товщиною лінії *int p* Параметри процедури:

Значення *q* відповідає наступним кольорам:

```
        switch(q)
    {
     case 1:Form1->Image3->Canvas->Pen->Color=clRed;break;
     case 2:Form1->Image3->Canvas->Pen->Color=clBlue;break;
     case 3:Form1->Image3->Canvas->Pen->Color=clGreen;break;
     case 4:Form1->Image3->Canvas->Pen->Color=clGray;break;
     default:Form1->Image3->Canvas->Pen->Color=clBlack;
    }
//---------------------------------------------------------------------------
  void GraphIm3(int  n, float X[], float Y[], float Xcf, float Ycf,
        float Xce, float Yce, float mxy,int q, int p)
  {
    int j,Xr[300],Yr[300];
    for(j=0;j<n;j++)
     {
```

```
   Xr[j]=floor((X[j]-Xcf)*mxy+Xce);
   Yr[j]=floor(-(Y[j]-Ycf)*mxy+Yce);
  }
 Form1->Image3->Canvas->Pen->Width=p;
 Form1->Image3->Canvas->Pen->Mode=pmCopy;
  switch(q)
  {
   case 1:Form1->Image3->Canvas->Pen->Color=clRed;break;
   case 2:Form1->Image3->Canvas->Pen->Color=clBlue;break;
   case 3:Form1->Image3->Canvas->Pen->Color=clGreen;break;
   case 4:Form1->Image3->Canvas->Pen->Color=clGray;break;
   default:Form1->Image3->Canvas->Pen->Color=clBlack;
  }
  for(j=0;j<n;j++)
   if (j==0)Form1->Image3->Canvas->MoveTo(Xr[j],Yr[j]);
   else Form1->Image3->Canvas->LineTo(Xr[j],Yr[j]);
 }
//---------------------------------------------------------------------------
```

Функція *void Obxod(int n, float x[], float y[])* забезпечує для плоского геометричного об’єкту з кількістю вершин *int n* та координатами вершин *float X[], float Y[]* обхід вершин проти годинникової стрілки.

```
void Obxod(int n, float x[], float y[])
{
 int i, nxi, nxe, nyi, nye;
 float xr[100], yr[100], xi, yi, xe, ye;
 bool pr;
 MaxMinNum(n,y,ye,nye,1);
 MaxMinNum(n,x,xe,nxe,1);
```

```
MaxMinNum(n,y,yi,nyi,-1);
MaxMinNum(n,x,xi,nxi,-1);
pr=(nxi<=nyi)&&(nyi<=nxe)&&(nxe<=nye);
if(pr)
{           for(i=n-1;i>=0;i--)
    {
        xr[n-i-1]=x[i];
        yr[n-i-1]=y[i];
    }
    for(i=0; i<n; i++)
    {   x[i]=xr[i];
        y[i]=yr[i];  }
}

}
//---------------------------------------------------------------------------
```

Функція *void ForDet(int n, int ni, float x[], float y[])*забезпечує для плоского геометричного об’єкту з кількістю вершин *int n* та координатами вершин *float X[], float Y[]* циклічну перестановку вершин таким чином, що першою вершиною плоского геометричного об’єкту становиться вершина з порядковим номером *int ni.*

```
void ForDet(int n, int ni, float x[], float y[])
{
    int i;
    float xr[100], yr[100];
    //k=n-ni-1;
    for(i=ni; i<n; i++)
    {
```

```
            xr[i-ni]=x[i];
            yr[i-ni]=y[i];
        }

        for(i=1; i<=ni; i++)
        {
            xr[n-ni-1+i]=x[i];
            yr[n-ni-1+i]=y[i];
        }

        for(i=0; i<n; i++)
        {
            x[i]=xr[i];
            y[i]=yr[i];
        }
    }
    //---------------------------------------------------------------------------
```

Функція *void VipDetNew(int n, float X[], float Y[], float Xob[], float Yob[], int &Nob)* забезпечує для плоского геометричного об'єкту з кількістю вершин *int n* та координатами вершин *float X[], float Y[]* побудову опуклої оболонки, де *int Nob* –кількість вершин оболонки та *float Xob[], float Yob[]* - координатами вершин оболонки.

```
    void VipDetNew(int n, float X[], float Y[], float Xob[], float Yob[], int &Nob)
    {
     int i,j,m,p;
     float A,B,C,Delta;
     bool Fl;
     i=0; p=0; m=1;
```

```
while(1)
{
 Fl = true;
 for(j=m+1; j<n-1;j++)
 {
  A=Y[m] - Y[i];
  B=X[i] - X[m];
  C=X[m]*Y[i] - X[i]*Y[m];
  Delta=A*X[j] + B*Y[j] + C;

  if(Delta <= 0)
  {
   m++;
   Fl = false;
   break;
  }
 }

 if(Fl)
 {
   Xob[p] = X[m];
   Yob[p] = Y[m];
   p++;
   i = m;
   m++;
 }
 if(m>n-1) break;
}
```

```
 Xob[p] = Xob[0];
 Yob[p] = Yob[0];
 p++;
 Nob = p;
}
//---------------------------------------------------------------------------
```

Функція *void GodVip(int nn, int nv, float xn[], float yn[], float xv[], float yv[], float xg[],float yg[], int &k)* забезпечує для нерухомого плоского геометричного об'єкту з кількістю вершин *int nn* та координатами вершин *float xn[], float yn[]* та рухомого плоского геометричного об'єкту з кількістю вершин *int nv* та координатами вершин *float xv[], float yv[]* побудову годографа вектор-функції щільного розміщення, де *int k* –кількість вершин годографа вектор-функції щільного розміщення та *float xg[],float yg[],* - координатами вершин годографа вектор-функції щільного розміщення.

```
void GodVip(int nn, int nv, float xn[], float yn[], float xv[], float
yv[], float xg[],float yg[], int &k)
{
  int i,j;
  float  s;
  i=0;  j=0;  k=-1;
  xn[nn]=xn[1];          yn[nn]=yn[1];
  xv[nv]=xv[1];          yv[nv]=yv[1];
  xn[nn+1]=xn[2];       yn[nn+1]=yn[2];
  xv[nv+1]=xv[2];       yv[nv+1]=yv[2];
 while(1)
 {
   k=k+1;
   //xg[k]=xv[i]-xn[j];   yg[k]=yv[i]-yn[j];
```

```
                xg[k]=-xv[i]+xn[j];   yg[k]=-yv[i]+yn[j];
               s=(xn[j+1]-xn[j])*(yv[i+1]-yv[i])-(yn[j+1]-yn[j])*(xv[i+1]-xv[i]);
             if  (((k>nn-1)  &&  (xg[0]==xg[k])  &&  (k>nv-1)  &&
(yg[0]==yg[k]))||(i>nv||j>nn)) break;
                /* if (s>0) j=j+1;
                   else  i=i+1;   */

                if (s>0) j=j+1;
                   else if (s<0)  i=i+1;
                      else {i=i+1; j=j+1;}
               }
               k=k+1;
             }
      //---------------------------------------------------------------------------
```

Функція *bool PointCrossTwoLine(float Xa,float Ya,float Xb,float Yb,float Xc,float Yc,float Xd,float Yd,float &X0,float &Y0)* визначає координати точки перетину двох відрізків *float X0,float Y0* відповідно з коодинатами кінців цих відрізків *float Xa,float Ya,float Xb,float Yb,float Xc* та *float Yc,float Xd,float Yd*,якщо ці відрізки перетинаються. Коли відрізки перетинаються функція повертає значення *True*, інакше вона повертає значення *False.*

```
bool PointCrossTwoLine(float Xa,float Ya,float Xb,float Yb,float Xc,float
Yc,float Xd,float Yd,float &X0,float &Y0)
{
    float A1,B1,C1,A2,B2,C2,Ra,Rb,Rd,Rc,Rab,Rcd,D1,D2,D0;
    int i;
    bool B;
    B=False;
    A1=Yb-Ya;
```

```
B1=Xa-Xb;
C1=Ya*Xb-Xa*Yb;
A2=Yd-Yc;
B2=Xc-Xd;
C2=Yc*Xd-Xc*Yd;
Ra=A2*Xa+B2*Ya+C2;
Rb=A2*Xb+B2*Yb+C2;
Rc=A1*Xc+B1*Yc+C1;
Rd=A1*Xd+B1*Yd+C1;
Rab=Ra*Rb;
Rcd=Rc*Rd;
if (Ra==0 && Rcd<0)
{X0=Xa;Y0=Ya;B=true;}
  else if (Rb==0 && Rcd<0)
  {X0=Xb;Y0=Yb;B=true;}
    else if (Rc==0 && Rab<0)
    {X0=Xc;Y0=Yc;B=true;}
      else if (Rd==0 && Rab<0)
      {X0=Xc;Y0=Yd;B=true;}
        else if (Ra==0 && Rc==0 && Rb!=0 && Rd!=0)
        {X0=Xc;Y0=Yc;B=true;}
          else if (Rb==0 && Rc==0 && Rb!=0 && Rd!=0)
          {X0=Xc;Y0=Yc;B=true;}
            else if (Ra==0 && Rd==0 && Rb!=0 && Rc!=0)
            {X0=Xd;Y0=Yd;B=true;}
              else if (Rb==0 && Rd==0 && Ra!=0 && Rc!=0)
              {X0=Xd;Y0=Yd;B=true;}
else
```

```
        {
        B=((Rcd<0)&&(Rab<0));
        if (B)
        {
        D0=A1*B2-A2*B1;
        D1=C2*B1-C1*B2;
        D2=C1*A2-C2*A1;
        X0=D1/D0;
        Y0=D2/D0;
        }
        }
        return B;
}
//---------------------------------------------------------------------------
```

Функція *float MaxMin(int n, float Z[], int p)* знаходить максимальний(мінімальний) ел-т q з масиву *float Z[]*.

Якщо int p = -1, то шукається мінімуми, якщо int p = 1, то максимум

```
 float MaxMin(int n,float Z[],int p)
    {
      int i;
      float q;
      q=Z[0];
      for (i=1;i<n;i++)
       if (p*q<p*Z[i]) q=Z[i];
      return q;
     }
//---------------------------------------------------------------------------
```

Функція *void __fastcall TForm1::Save1Click(TObject *Sender)* зберігає інформацію у файлі **.dgt* про контури плоских геометричних обєктів, які були введені в інтерактивному режимі.

```
void __fastcall TForm1::Save1Click(TObject *Sender)
{
 char Fname[120],T;
 AnsiString NameF;
 int i,j,L;
if (SaveDialog1->Execute())
  {
   NameF=SaveDialog1->FileName;
  for (i=0;i<119;i++)Fname[i]=NULL;
  L=NameF.Length();
 for(i=1; i<=L;i++)
  {
   T=NameF[i];
   Fname[i-1]=T;
  }
  }
ofstream output(Fname,ios::out);
output<<"Model"<<endl;
output<<"AAAAA"<<endl;
output<<KilDet<<endl;
for(i=0;i<p;i++)
 output<<"Detal"<<i+1<<endl;
NameDet[p-1]=IntToStr(p);
for(i=0;i<p;i++)
 output<<KilksPointDet[i]<<endl;
```

```
 for(i=0;i<p;i++)
  for(j=0;j<KilksPointDet[i];j++)
   if(j==0)
    output<<Xd[i][j]<<"    "<<Yd[i][j]<<"   Detal"<<i+1<<endl;
   else output<<Xd[i][j]<<"    "<<Yd[i][j]<<endl;
   output.close();
}
//---------------------------------------------------------------------------
```

Функція *void __fastcall TForm1::Opendgt1Click(TObject *Sender)* читаєає інформацію з файлу **.dgt* про контури плоских геометричних обєктів, для яких необхідно побудувати щільні укладки

```
void __fastcall TForm1::Opendgt1Click(TObject *Sender)
{
 char Fname[120],T,ModelT[40],Q[20],Det[30][20];
 AnsiString NameF;
 int i,j,L;
   for (i=1;i<41;i++)ModelT[i]=NULL;
   for (i=1;i<21;i++)Q[i]=NULL;
    for (j=0;j<30;j++)
     for(i=0;i<20;i++) Det[j][i]=NULL;
if (OpenDialog1->Execute())
   {
    NameF=OpenDialog1->FileName;
   for (i=0;i<119;i++)Fname[i]=NULL;
   L=NameF.Length();
  for(i=1; i<=L;i++)
   {
    T=NameF[i];
```

```
   Fname[i-1]=T;
  } }
 ifstream input(Fname,ios::in);
 input>>ModelT;
 input>>Q;
 input>>KilDet;
 for(i=0;i<KilDet;i++)
 input>>Det[i];
 //NameDet[p-1]=IntToStr(p);
 for(i=0;i<KilDet;i++)
  input>>KilksPointDet[i];
 for(i=0;i<KilDet;i++)
   for(j=0;j<KilksPointDet[i];j++)
  if (j==0)input>>Xd[i][j]>>Yd[i][j]>>Q;
   else
   input>>Xd[i][j]>>Yd[i][j];
   input.close();
  float XcE,YcE,XcIm3,YcIm3;
   p=KilDet;
   XcE=Image2->Width/2;
   YcE=Image2->Height/2;
  /* PageControl1->ActivePage=TabSheet2;
   for(i=0;i<p;i++) ParamDet(i);
   ParamModeli();
   BuildIm2(XcE,YcE);
   BuildIm3(XcIm3,YcIm3);  */
}
//---------------------------------------------------------------------------
```

Функція *float skv(int n,float X[],float Y[])* визначає площу *S* многокутника, який визначається наступними параметрами:

int n - кількість вершин многокутника;

float X[],float Y[]- масиви з координатами вершин многокутника.

```
float skv(int n, float X[],float Y[])
{       int i;
        float S;
        S=0;
        for(i=0;i<n-1;i++)
        S=S+X[i]*Y[i+1]-X[i+1]*Y[i];
        S=fabs(S)/2;
        return S;}
```

3.4. Інструкції по роботі з програмним продуктом

Початок роботи з програмою розпочинається з запуску файлу *PrOvcharov.exe*. При цьому на екрані з‘являється головне вікно програми прийме настуний вигляд(рис. 3.1).

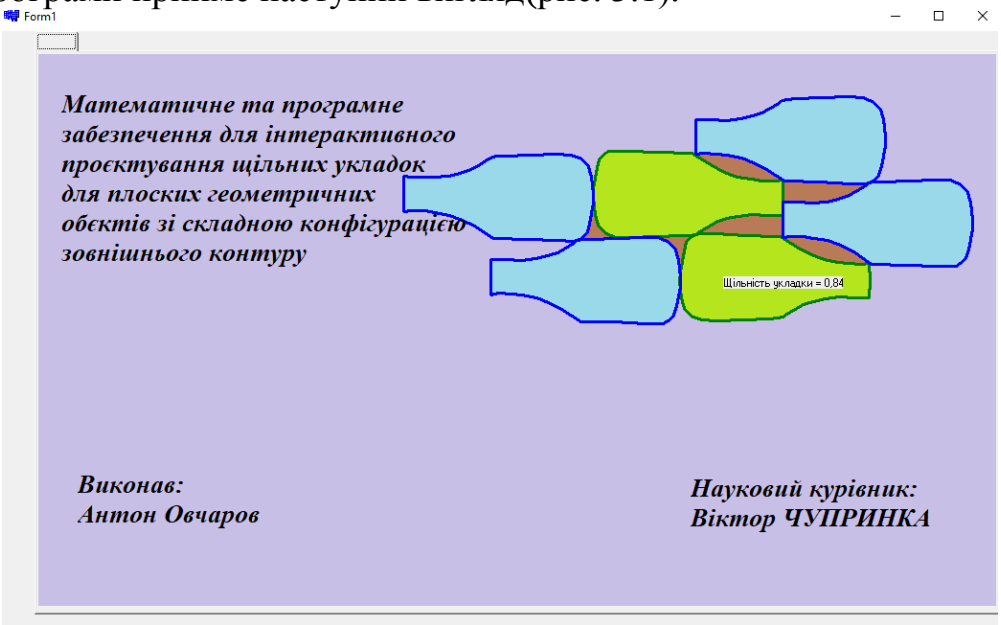


Рис. 3.1

Після натиску на кнопку «Робота» неохідно вибрати файл *.bmp з кресленнями необхідних плоских геометричних об’єктів та натиснути на кнопку «Ввести». (рис. 3.2).

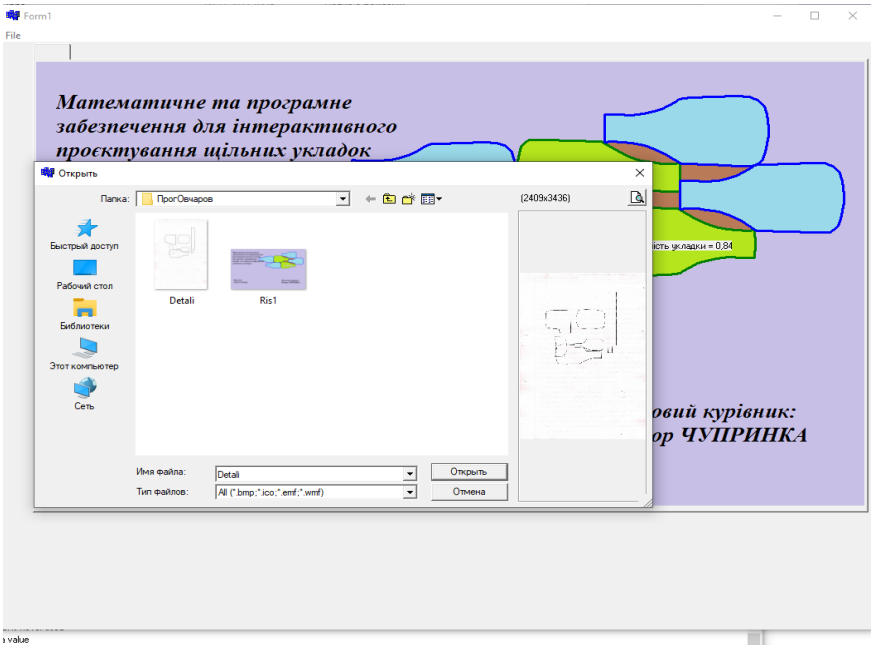


Рис. 3.2

Після введення цього головне вікно програми настуний вигляд (рис. 3.3).

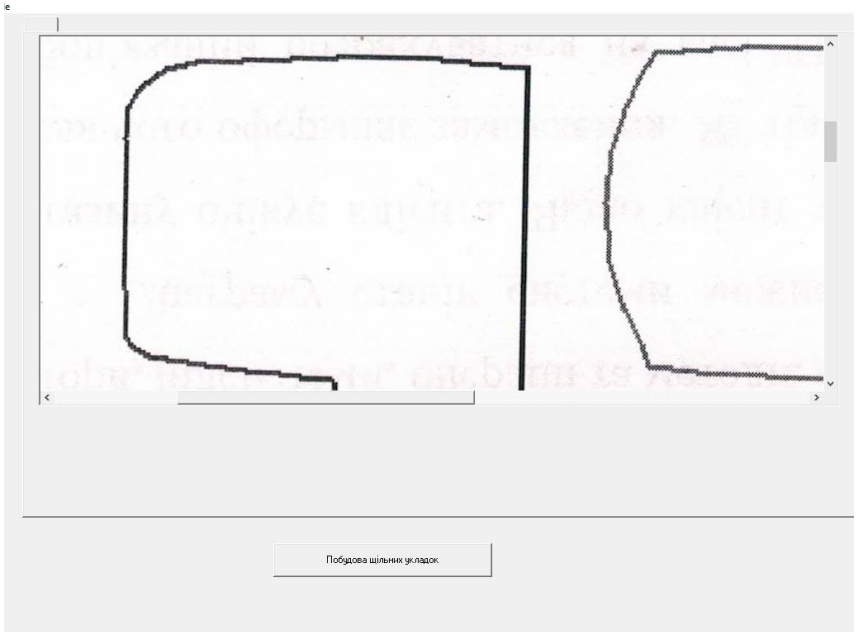


Рис. 3.3

Після інтерактивного введення інформації про зовнішні контури плоских геометричних об’єктів на натиску на кнопку «Побудова щільних укладок» головне вікно програми настуний вигляд (рис. 3.4).

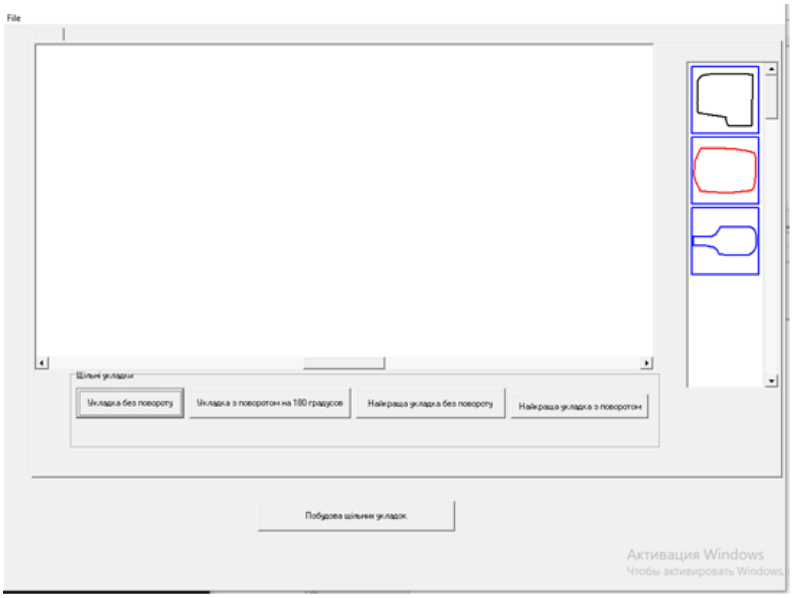


Рис. 3.4

Після вибору плоского геометричного об’єкту із меню, що знаходиться праворуч на формі, та натиску на кнопку «Щільні укладки без повороту» головне вікно програми настуний вигляд (рис. 3.5).

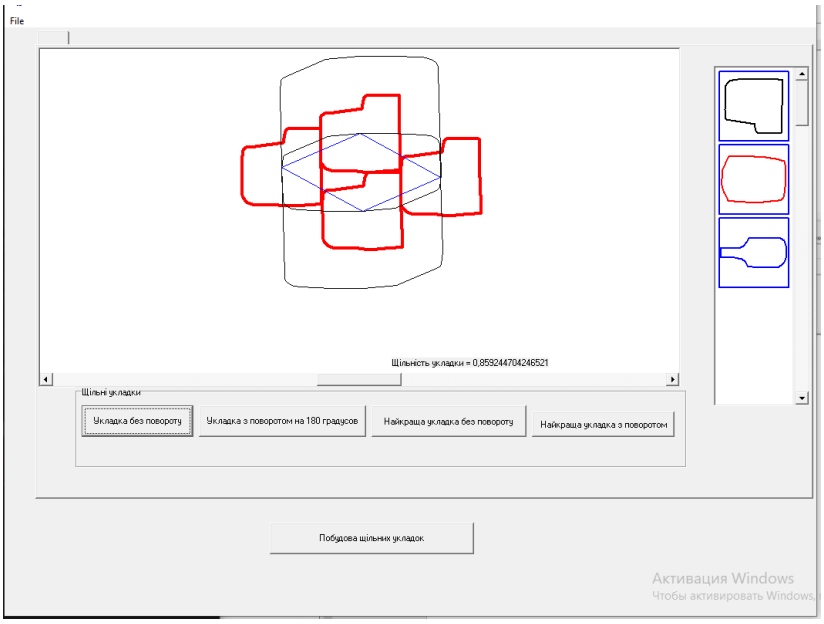


Рис. 3.5

Після натиску на кнопку «Найкраща укладки без повороту» головне вікно програми настуний вигляд (рис. 3.6), де буде виведена найкраща укладки без поворота для вибраного плоского геометричного об’єкту.

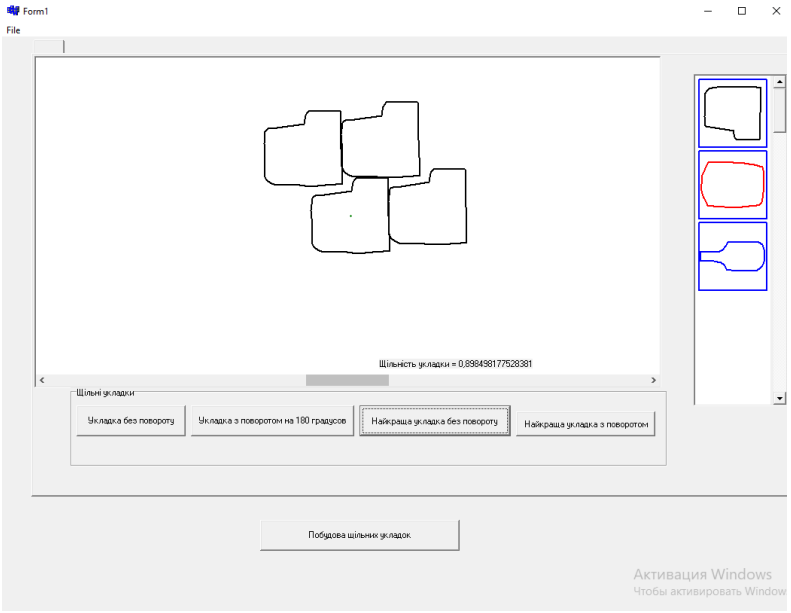


Рис. 3.6

Після вибору плоского геометричного об’єкту із меню, що знаходиться праворуч на формі, та натиску на кнопку «Щільні укладки з поворотом на 180^{o}» головне вікно програми настуний вигляд (рис. 3.7).

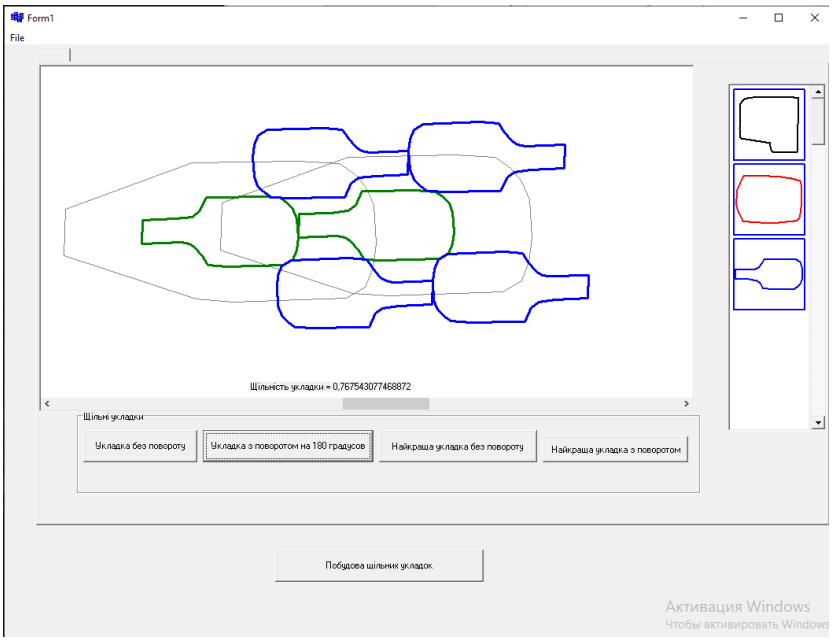


Рис. 3.7

Після натиску на кнопку «Найкраща укладки з поворотом на 180^{o}» головне вікно програми настуний вигляд (рис. 3.8), де буде виведена найкраща укладки ,з поворотом 180^{o} для вибраного плоского геометричного об’єкту.

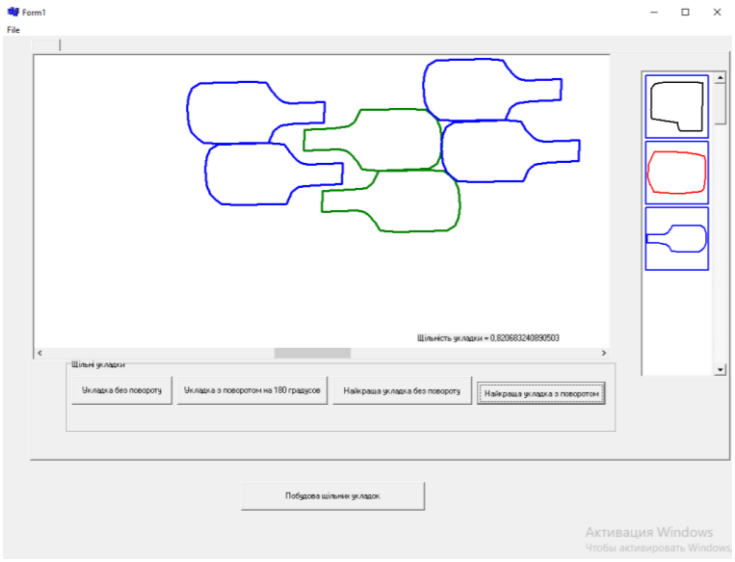


Рис. 3.8

Координати вершин плоских геометричних обєктів, які отримані в результаті інтерактивного введення їх по зображенню їх креслень із файлу *.bmp можна зберегти у файлі *.dgt.. Для цього треба натиснути на кнопку «Зберегти». (рис. 3.9).

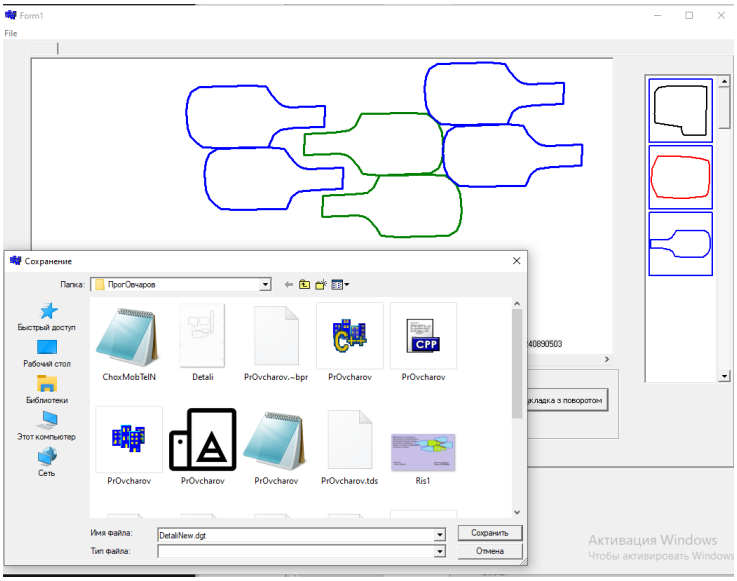


Рис. 3.9

Крім того є можливість загрузити інформацію про координати вершин плоских геометричних обєктів файлі *.dgt.. Для цього треба натиснути на кнопку «Відкрити». (рис. 3.10).

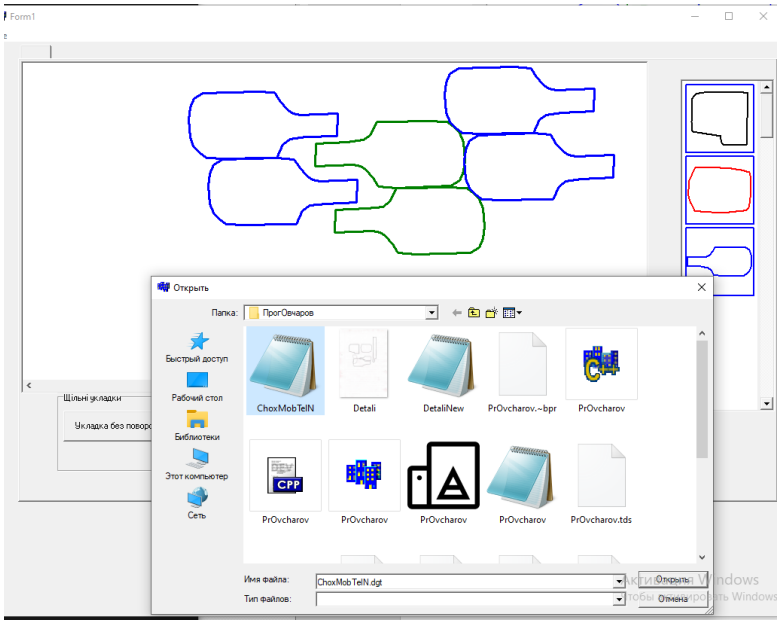


Рис.3.10

Після цього головне вікно програми настуний вигляд (рис. 3.11).

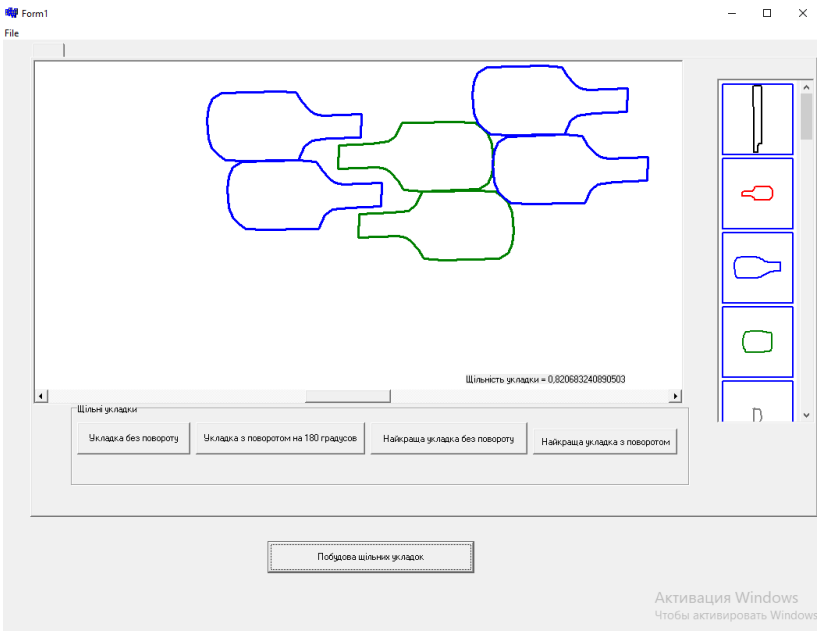


Рис. 3.11

Структура файлу *.dgt для деталей чохла до мобільного телефону представлена нижче:

ЧохолМобТел – *назва моделі*
555– *службова інформація*
6 – *кількість деталей у моделі*
Ботан – *назва першої деталі у файлі з інформацією про зовнішні контури деталей*
Накладка – *назва другої деталі у файлі з інформацією про зовнішні контури деталей*
Передня_стінка – *назва третьої деталі у файлі з інформацією про зовнішні контури деталей*
Задня_стінка– *назва четвертої деталі у файлі з інформацією про зовнішні контури деталей*
NK– *назва п'ятої деталі у файлі з інформацією про зовнішні контури деталей*
Плівка– *назва шостої деталі у файлі з інформацією про зовнішні контури деталей*

12 – *кількість вершин на зовнішньому контурі ботана*
36 – *кількість вершин на зовнішньому контурі _накладки*
41 – *кількість вершин на зовнішньому контурі передньої_стінки*
34 –*кількість вершин на зовнішньому контурі*
24 –*кількість вершин на зовнішньому контурі NK*
26 –*кількість вершин на зовнішньому контурі плівки*

190.82 75.41 Ботан
200.66 75.28
201.04 93.56
201.55 94.96
202.58 95.86
203.73 95.86
210.76 96.37
207.69 240.28
188.26 240.15
189.03 189.67
190.69 82.44
190.82 75.41

Відповідно координати вершин X та У на зовнішньому контурі ботана

153.50 111.70 Накладка
169.73 111.45
172.03 111.32
174.46 110.30
176.25 108.38
177.65 105.19
178.68 100.33
178.80 95.34
178.68 90.74
177.27 86.40
174.46 82.31
171.13 80.01
169.09 79.50
161.04 79.37
153.88 78.99
141.87 78.99
135.09 78.60
133.94 80.65
131.90 84.48
129.73 86.40
127.68 87.93
125.64 88.06
123.33 88.44
116.05 88.70
104.55 88.95
104.42 93.56
104.29 98.67
104.67 99.31
123.21 99.82
126.66 100.33
129.34 102.12
131.90 105.06
135.09 111.07
142.38 111.45
142.38 111.32
153.50 111.70

Відповідно координати вершин X та У на зовнішньому контурі накладки

57.98	65.39 Передня_стінка
57.85	71.26
59.00	81.22
60.79	87.86
64.88	92.33
71.39	95.14
80.97	95.40
112.38	94.12
118.00	93.99
121.07	93.36
123.24	91.70
128.35	88.63
133.33	85.82
139.71	83.78
146.74	82.12
155.55	81.10
165.00	81.61
169.85	81.61
174.07	81.73
174.20	71.13
174.07	60.28
169.47	60.92
161.17	60.53
154.91	59.77
146.23	58.11
138.69	55.17
130.65	51.21
120.56	44.57
119.15	43.80
117.62	43.80
98.21	43.29
79.69	42.40
70.24	42.27
67.81	42.53
66.54	43.29
65.26	44.06
63.47	45.72
61.81	47.64
60.28	52.74
60.28	52.74

Відповідно координати вершин X та $У$ на зовнішньому контурі передньої стінки

57.98 65.39

178.55 187.37 Задня_стінка
165.89 190.18
155.93 191.20
144.42 191.84
135.73 191.97
129.09 191.71
123.97 191.20
114.90 190.43
112.60 186.47
110.68 181.36
108.64 174.71
107.61 166.53
107.87 162.44
108.51 156.95
110.81 149.79
113.62 144.30
115.03 140.59
126.53 139.44
133.81 139.31
141.61 139.05
150.69 139.44
160.53 140.46
169.22 141.61
174.97 142.76
177.01 143.91
178.42 145.06
178.93 147.75
180.21 154.39
180.85 161.93
180.59 168.45
180.46 176.50
179.83 181.49
179.83 181.36
178.55 187.37

Відповідно координати вершин X та Y на зовнішньому контурі задньої стінки

-9.95	17.53 NK
-9.61	11.18
-8.67835	8.80536
-8.25502	6.94269
-8.33968	-9.05934
-9.18635	-11.43
-9.44035	-17.4414
-6.05368	-17.4414
-2.159	-17.78
0.804337	-17.3567
4.61434	-15.1553
7.32368	-12.446
9.01702	-9.48268
9.94836	-5.588
9.86369	-1.26999
9.60969	2.45535
8.84769	7.19669
7.83168	11.5994
6.39235	14.3087
4.61434	16.51
1.397	16.9334
-2.159	17.1874
-6.22301	17.78
-9.95	17.53

Відповідно координати вершин X та У на зовнішньому контурі NK

50.49	138.69	Плівка
37.46	140.07	
34.58	140.57	
32.45	141.70	
31.07	143.45	
30.69	151.22	
30.44	175.40	
30.82	180.28	
32.45	182.92	
36.21	186.30	
41.22	187.43	
56.63	188.55	
71.91	189.43	
80.81	188.93	
94.84	187.43	
94.97	126.91	
72.16	126.54	
69.53	126.79	
68.15	127.16	
66.78	128.42	
65.65	130.67	
64.77	135.68	
64.40	137.19	
62.89	137.44	
55.75	138.19	
50.49	138.69	

Відповідно координати вершин X та У на зовнішньому контурі плівки

ВИСНОВКИ

Проведено дослідження та розроблені алгоритми для інтерактивного проєктування щільних укладок для плоских геометричних об’єктів зі складною конфігурацією зовнішнього контуру в основном положенні та з поворотом на 180^0.

Запропоноване математичне забезпечення для інтерактивного проєктування плоских геометричних об’єктів зі складною конфігурацією зовнішнього контуру реалізоване в прогрограмний продукт.

Програмне забезпечення дозволяє введення інформації про зовнішні контури плоских геометричних об’єктів із файлу з координатами вершин цих плоских геометричних об’єктів формату *.dgt та із файлу *.bmp із відсканованим зображенням цих плоских геометричних об’єктів.

Цей програмний продукт легкий в користуванні, має дружній та привабливий інтерфейс. Може бути застосована на підприємствах в взуттєвої галузі легкої промисловості.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Дякон В. М. Моделі і методи теорії прийняття рішень : підручник / В. М. Дякон, Л. Є. Ковальов. – К.: АНФ ГРУП, 2013. – 604 с.
2. Зайченко Ю.П. Дослідження операцій: Підручник. – К.: ВІПОЛ, 2010.
3. K. Weiler. An incremental angle point in polygon test, in: P. Heckbert (Ed.), Graphic Gems IV, Academic Press, Boston, MA, 1994, pp. 16—23.
4. Hormann K., Agathos A. The point in polygon problem for arbitrary polygons (англ.) // Comput. Geom. Theory Appl.. — 2001. — Vol. 20. — P. 131—144.
5. J. Melissen. Packing 16, 17 or 18 circles in an equilateral triangle // Discrete Mathematics. — 1995. — Т. 145. — С. 333–342. — doi:10.1016/0012-365X(95)90139-C.
6. Y. G. Stoyan, G. N. Yaskov. Packing identical spheres into a cylinder // International Transactions in Operational Research. — 2010. — Т. 17. — С. 51–70. — doi:10.1111/j.1475-3995.2009.00733.x.
7. P. Erdős, R. L. Graham. On packing squares with equal squares // Journal of Combinatorial Theory, Series A. — 1975. — Т. 19. — С. 119-123. — doi:10.1016/0097-3165(75)90099-0.
8. Lodi, S. Martello, M. Monaci. Two-dimensional packing problems: A survey // European Journal of Operational Research. — Elsevier, 2002. — Т. 141. — С. 241–252. — doi:10.1016/s0377-2217(02)00123-6.
9. A. Haji-Akbari, M. Engel, A. S. Keys, X. Zheng, R. G. Petschek, P. Palffy-Muhoray, S. C. Glotzer. Disordered, quasicrystalline and crystalline phases of densely packed tetrahedra // Nature. — 2009. — Т. 462, вип. 7274. — С. 773–777 — doi:10.1038/nature08641. — Bibcode: 2009Natur.462..773H. — arXiv:1012.5138. — PMID 20010683.

10. E. R. Chen, M. Engel, S. C. Glotzer. Dense Crystalline Dimer Packings of Regular Tetrahedra // Discrete & Computational Geometry. — 2010. — Т. 44, вып. 2. — С. 253–280. — doi:10.1007/s00454-010-9273-0.
11. Cherri, L. H., Cherri, A. C., and Soler, E. M. (2018). Mixed integer quadratically-constrained programming model to solve the irregular strip packing problem with continuous rotations. *J. Glob. Optim.* 72 (1), 89–107. doi:10.1007/s10898-018-0638-x. https://link.springer.com/article/10.1007/s10898-018-0638-x
12. Peralta, J., Andretta, M., and Oliveira, J. F. (2018). Solving irregular strip packing problems with free rotations using separation lines. *Pesqui. Oper.* 38 (2), 195–214. doi:10.1590/0101-7438.2018.038.02.0195. https://www.scielo.br/j/pope/a/RcXzqWKwBnL7QhcgkgNyZPv/?lang=en
13. Stoyan, Y., Pankratov, A., and Romanova, T. (2016). Cutting and packing problems for irregular objects with continuous rotations: Mathematical modelling and non-linear optimization. *J. Operational Res. Soc.* 67 (5), 786–800. doi:10.1057/jors.2015.94. https://www.tandfonline.com/doi/abs/10.1057/jors.2015.94?journalCode=tjor20
14. Wang, A., Hanselman, C. L., and Gounaris, C. E. (2018). A customized branch-and-bound approach for irregular shape nesting. *J. Glob. Optim.* 71 (4), 935–955. doi:10.1007/s10898-018-0637-y. https://link.springer.com/article/10.1007/s10898-018-0637-y
15. Guo, B., Ji, Y., Hu, J., Wu, F., and Peng, Q. (2019). Efficient free-form contour packing based on code matching strategy. *IEEE Access* 7, 57917–57926. doi:10.1109/ACCESS.2019.2914248. https://ieeexplore.ieee.org/abstract/document/8704207
16. Havrylov T.M. Model′ avtomatychnoho proektuvannya skhem rozkroyu lystovykh materialiv na detali vzuttya /T.M. Havrylov, V.I. Chuprynka //Visnyk KNUTD.- 2011, №6. – S. 83-88. https://knutd.edu.ua/files/Visnyk/Visnuk_6_2011.pdf

17. Chuprynka V.I. Metod avtomatyzovanoho proektuvannya shchil′nykh ukladok pry pryamokutno-hnizdoviy skhemi rozkroyu / V.I. Chuprynka, V.S. Murzhenko//Visnyk KNUTD.- 2011, №6. – S. 72-77. https://knutd.edu.ua/files/Visnyk/Visnuk_6_2011.pdf
18. Chuprynka V.I., Naumenko B.V., Vasylenko O.L. Heneruvannya ratsional′nykh skhem rozkroyu rulonnykh materialiv na detali shkirhalantereyi. Mekhatronni systemy: innovatsiyi ta inzhynirynh tezy dopovidey VI mizhnar. Nauk.-prakt. konf., 24 lystopada 2022 r. Kyiv: KNUTD, S. 157-158. https://er.knutd.edu.ua/bitstream/123456789/20956/1/MSIE_2022_P157-158.pdf
19. Kolysko O.Z. Modyfikatsiya henetychnoho alhorytmu dlya heneratsiyi sektsiy rozkriynykh skhem/ O.Z. Kolysko // Visnyk KNUTD. –2009.-№1. – S. 54-56. https://er.knutd.edu.ua/bitstream/123456789/6983/1/V45_P014-017.pdf
20. Кондаков А.І. САПР технологічних процесів і виробництв. ACADEMA, 2007
21. Березовський В.С., Потієнко, В.О. та Завадський, І.О., 2009. Основи комп’ютерної графіки. Київ: BHV.
22. Ванін В.В., Перевертун, В.В. та Надкернична, Т.М., 2006. Комп’ютерна інженерна графіка в середовищі AutoCAD. Київ: Каравела.
23. Веселовська Г.В. та Ходакова, В.Є., 2015. Комп’ютерна графіка. Київ: Кондор.
24. Горобець С.М., 2006. Основи комп’ютерної графіки. Київ: Центр навчальної літератури.
25. Романюк О.Н., 2001. Комп’ютерна графіка. Вінниця: Вінницький державний технічний університет.
26. E. G. Birgin, R. D. Lobato, R. Morabito. An effective recursive partitioning approach for the packing of identical rectangles in a rectangle // Journal of the Operational Research Society. — 2010. — Т. 61. — С. 306–320. — doi:10.1057/jors.2008.141.

27. D.A. Klarner, M.L.J Hautus. Uniformly coloured stained glass windows // Proceedings of the London Mathematical Society. — 1971. — T. 23. — doi:10.1112/plms/s3-23.4.613.
28. Eckard Specht. The best known packings of equal circles in an isosceles right triangle (11 березня 2011).
29. U. Betke, M. Henk. Densest lattice packings of 3-polytopes // Comput. Geom.. — 2000. — T. 16. — С. 157–186.
30. H. Minkowski. Dichteste gitterförmige Lagerung kongruenter Körper // Nachr. Akad. Wiss. Göttingen Math. Phys. KI. II. — 1904. — С. 311–355.
31. Роберт Мартін Чиста архітектура. Мистецтво розроблення програмного забезпечення.Видавництво — Фабула Жанр, ISBN — 978-617-09-5286-Рік видання — 2019, 416 с,
32. Ерік Фрімен , Елізабет Робсон Патерни проєктування, Видавництво — Фабула Жанр, Видавництво — Фабула Жанр, ISBN — 978-617-09-6159-4, Рік видання — 2020, 672 с.
33. Роберт Мартін Чистий Agile, Видавництво — Фабула Жанр, ISBN — 978-617-09-6760-2 , Рік видання — 2021, 224 с.
34. Олексій Васильєв Програмування мовою PYTHON Видавництво — Навчальна книга Богдан Жанр, ISBN — 978-966-10-5611-3, 504 с.
35. Берт Бейтс , Кеті Сьєрра JAVA, Видавництво — Фабула Жанр, ISBN — 978-617-522-033-7, Рік видання — 2022, 720 с.
36. Пол Беррі Head First Python, Видавництво — Фабула Жанр, ISBN — 978-617-522-019-1 , Рік видання — 2021 ,624 с.
37. Дж. Генк Рейнвотер Як пасти котів. Посібник для програмістів, які мають керувати іншими програмістами, Видавництво — Фабула Жанр, ISBN — 978-617-09-6155-6, Рік видання — 2020, 320 с.

38. Томас Г. Кормен, Чарлз Е. Лейзерсон, Роналд Л. Рівест, Кліфорд Стайн Вступ до алгоритмів, Видавництво : К.І.С., ISBN : 9786176842392,Рік видання : 2019, 1288 с.

39. Роберт Бош Opt Art. Від математичної оптимізації до візуального дизайну Видавництво : Фабула, ISBN : 9786175220795, 200 с.10.

40. Джин Кім, Джез Хамбл, Патрік Дебуа, Джон Вілліс DevOps. Посібник. Як домогтися гнучкості, надійності і безпеки світового рівня в технічних компаніях Видавництво : Фабула, ISBN : 9786170979841, Рік видання : 2023, 384 с.