МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ТЕХНОЛОГІЙ ТА ДИЗАЙНУ
(повне найменування університету)

ФАКУЛЬТЕТ МЕХАТРОНІКИ ТА КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
(назва факультету)

КАФЕДРА КОМП'ЮТЕРНИХ НАУК
(повна назва кафедри)

Дипломна магістерська робота

на тему

«Розробка математичного та програмного забезпечення для проєктування декоративних елементів на деталях взуття»

Виконав: студент групи ЗМгІТ-22
спеціальності 122 Комп'ютерні науки
Роман ЩЕРБАТЮК

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

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

Київ 2023

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ТЕХНОЛОГІЙ ТА ДИЗАЙНУ
(повне найменування університету)

ФАКУЛЬТЕТ МЕХАТРОНІКИ ТА КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
(назва факультету)

КАФЕДРА КОМП'ЮТЕРНИХ НАУК
(повна назва кафедри)

Спеціальність 122 Комп'ютерні науки
(шифр і назва)

Освітня програма Комп'ютерні науки

ЗАТВЕРДЖУЮ

Завідувач кафедри КН

Володимир ШЕРБАНЬ

"____" _____ 2023 р.

З А В Д А Н Н Я

НА ДИПЛОМНУ МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Щербатюку Роману Васильовичу

1. Тема роботи «Розробка математичного та програмного забезпечення для проєктування декоративних елементів на деталях взуття»
науковий керівник роботи д.т.н., професор Чупринка Віктор Іванович**,**
затверджені наказом вищого навчального закладу від «12» вересня 2023 року, № 210-уч.
2. Строк подання студентом роботи - 10 листопада 2023 р.
3. Вихідні дані до роботи: розробка кафедри комп'ютерних наук, літературні джерела з конструювання та технології одягу
4. Зміст дипломної роботи (перелік питань, які потрібно розробити)
Вступ, РОЗДІЛ 1(Сапр у взуттєвій галузі легкої промисловості); РОЗДІЛ 2(Математичне забезпечення для автоматизованого проєктування декоративних елементів на деталях взуття); РОЗДІЛ 3(Програмне забезпечення автоматизованого проєктування декоративних елементів на деталях взуття)

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

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної магістерської роботи	Терміни виконання етапів	Примітка про виконання
1	Вступ	05.09. 2023	
2	Розділ 1. Сапр у взуттєвій галузі легкої промисловості	20.09.2023	
3	Розділ 2. Математичне забезпечення для автоматизованого проєктування декоративних елементів на деталях взуття	03.10.2023	
4	Розділ 3. Програмне забезпечення автоматизованого проєктування декоративних елементів на деталях взуття	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)	31.10.2023	
8	Перевірка дипломної магістерської роботи на наявність ознак плагіату (за 10 днів до захисту)	01.10.2023	
9	Подання дипломної магістерської роботи у відділ магістратури для перевірки виконання додатку до індивідуального навчального плану (за 10 днів до захисту)	07.11.2023	
10	Подання дипломної магістерської роботи на затвердження завідувачу кафедри (з 7 днів до захисту)	9.12.2023	

Студент _____ **Роман ЩЕРБАТЮК**
(підпис) (ім'я та прізвище)

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

Керівник відділу магістратури _____ **Олена ГРИГОРЕВСЬКА**
(підпис) (ім'я та прізвище)

АНОТАЦІЯ

Щербатюк Р.В. Розробка математичного та програмного забезпечення для проєктування декоративних елементів на деталях взуття – 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 проєктування декоративних елементів на деталях взуття. Запропоновані алгоритми реалізовані в програмний продукт для автоматизованого проєктування декоративних елементів на деталях взуття.

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, взуття, декоративні елементи

ABSTRACT

Shcherbatiuk R.V. Development of mathematical and software for designing decorative elements on shoe details - 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 designing decorative elements on shoe details. The proposed algorithms are implemented in a software product for the automated design of decorative elements on shoe details.

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, shoes, decorative elements.

Зміст

ВСТУП

Ринок товарів на замовлення постійно зростає, все більше у всьому світі стає людей зацікавлених у придбанні продукції на замовлення, яка б відповідала їхнім індивідуальним потребам, смакам, соціальному статусу чи рангу. Взуття служить для полегшення пересування та запобігання травмам стопи.

Проте, вибір доступного взуття має деякі потенційно суперечливі критерії для споживачів, де доводиться часто вибирати між комфортом і естетикою. Більшість, як правило, віддають перевагу візуально-естетичним показникам над комфортним взуттям, але нажаль цей вибір в результаті призводить до різноманітних паталогій стопи в процесі носіння виробу та після. Крім того, більшість виробників взуття зараз імітують сучасні модні стилі у виробах, але йдуть на компроміс з якістю, щоб знизити вартість товару.

Взуття повинно відповідати низці вимог, виконувати захисну, утилітарну та естетично-ергономічні функції, оскільки це дозволить створити оптимальні умови для нормальної діяльності стопи та організму в цілому. Естетично-ергономічні функції дозволяють створити природне середовище для стопи без її деформацій, тобто, за формою та окремими абрисами взуття має відповідати сформованій стопі. Певною мірою забезпечення естетично-ергономічних функцій взуття залежить і від якості проектування, тому що рационально спроектоване взуття з врахуванням анатомо-морфологічних властивостей стопи забазпечує раціональність даного виробу у подальшій його експлуатації.

Протягом тисячоліть ремісництво взуття завжди має свій попит та відіграє важливу роль у суспільстві. Виробники взуттєвих підприємств, студії та ательє пропонують пошив індивідуального взуття для клієнта. Майстри взуття задовольняють конкретні запити персоналізованої продукції споживача.

При класичному методі проектування взуття на колодку наносять допоміжні і стильові лінії, а потім за цими контурами проектують деталі. Таким

чином ми визначаємо частину тривимірної поверхні колодки, потім «розплющуємо» її для отримання плоских шаблонів і викроювання деталей, а потім заготовку, зібрану з плоских деталей, затягуємо на колодку, перетворюючи їй знову в просторовий об’єкт. Така подвійна трансформація часто призводить до того, що отриманий виріб не зовсім відповідає задуму модельєра.

Постійно зростаючі вимоги до дизайну взуття, його якості при одночасній необхідності скорочення термінів розробки нових моделей і їхнього запуску в серійне виробництво, а також утримання цін на конкурентоздатному рівні в буквальному значенні, змушують виробників впроваджувати новітні технології на всіх етапах проектування та виготовлення взуття. Безпосереднє перенесення зарубіжного досвіду, технологій, устаткування не завжди прийнятне, оскільки, вимагає значних капіталовкладень, практично недоступних для сучасного вітчизняного підприємства. Розробка власних систем автоматизованого проектування в першу чергу вимагає попередньої розробки їх наукової аналітичної основи, спираючись при цьому на концепцію виробництва, характерного для масового, але враховуючи при цьому анатомо-морфологічні властивості стоп.

В сучасних умовах швидкої зміни моди і смаків споживача та наявності великого асортименту взуття, прискорюється процес морального старіння виробів, тому важливо, щоб темпи їх проектування були якнайшвидшими. Для конкурентоспроможності виробів взуттєвого виробництва і підвищення його ефективності першочергове значення має рівень конструкторсько-технологічної підготовки, якої сприяє застосування систем автоматизованого проектування (САПР). За останній час у нас в країні, як і в усьому світі, здійснюється комп'ютеризація взуттєвого виробництва, активно впроваджуються нові інформаційні технології. Звідси очевидна актульність розробки математичного та програмного забезпечення для проєктування декоративних елементів на деталях взуття

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 проєктування декоративних елементів на деталях взуття.

Наукова новизна полягає у розробці математичного та програмного забезпечення для проєктування декоративних елементів на деталях взуття.

. 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 проєктування декоративних елементів на деталях взуття» опубліковано одна наукових робота.

Структура та обсяг магістерської роботи. Магістерська робота містить вступ, три розділи, загальні висновки досліджень, список використаних джерел, додатки.

1.САПР У ВЗУТТЄВІЙ ГАЛУЗІ ЛЕГКОЇ ПРОМИСЛОВОСТІ

1.1. Автоматизовані системи проектування взуття

Автоматизована система управління взуттєвим підприємством являє собою управління із застосуванням сучасних високоефективних автоматичних засобів обробки даних, економіко-математичних методів для регулярного рішення задач управління виробниче-господарською діяльністю підприємства, автоматизованого проектування технологічних процесів і різноманітності конструкцій взуття, що випускається.

Основні переваги застосування САПР:

- підвищення точності побудови;
- зниження трудомісткості.

Обчислювальна техніка і сучасні методи управління дозволяють вирішувати виробничі задачі за порівняно короткий час і значно підвищити продуктивність труда.

Використання автоматизованих систем управління є неодмінною умовою ефективного функціонування технологічного підрозділу.

Оглядова характеристика САПР взуття

Система автоматизованого проектування (САПР) взуття являє собою організаційно-технічну систему, що складається з комплексу коштів автоматизованого проектування взаємодіючого з розробниками проектно-конструкторської документації.

САПР реалізовується в різних варіантах пространственности:

1. Просторове проектування. У його основу встановлені алгоритми, що забезпечують роботу пристроїв знімання параметрів поверхні взуттєвої колодки і отримання умовної розгортки для розробки плоских деталей конструкції (система 3D). Алгоритми просторового проектування на вітчизняних і

зарубіжних підприємствах не знаходять широкого практичного застосування через складність і високу вартість таких пристроїв.

2. Площинне проектування. Ведучою є концепція розробки і вдосконалення алгоритмів функціонування автоматизованих систем площинного проектування (система 2D). Структура побудови такого програмного комплексу базується на двох основних принципах:

- спадкоємність і органічна єдність з діючими структурами підготовки виробництва;
- об'єднання і рішення максимально можливого числа задач конструкторської підготовки з урахуванням можливостей комп'ютерної техніки.

Аналіз чого склався структури процесу підготовки проекту і потреби виробництва передбачає наявність в системі проектування наступних базових програмних модулів:

- введення початкової інформації;
- проектування моделі і її деталей;
- градирование контурів деталей;
- виведення на графічний або друкуючий пристрій;
- контроль укладиваемости деталей;
- розрахунок трудомісткості зборки моделі;
- формування паспорта моделі.

Пакет прикладних програм являє собою комплекс програм, працюючий під управлінням головної програми і призначений для рішення певного класу задач, як правило, близьких за змістом або по вживаних математичних методах. Пакет прикладних програм є найбільш довершеною формою програмного забезпечення САПР.

Системи і програмні комплекси автоматизованого проектування взуття: Уперше САПР взуття успішно демонструвалися в Пірмазенсе (ФРН), а потім на виставці "Тиждень шкіри" в Парижі в 1985 році. Розробки САПР знайшли

підтримку у ведучих взуттєвих фірм, що надали свої підприємства для випрбовування експериментальних систем.

Двухкоординатние системи автоматизованого проектування на всіх підприємствах зарекомендували себе добре, хоч і з різним рівнем продуктивності. Проте, сам вибір системи для конкретного підприємства залишається досить складним, оскільки необхідний, щоб системи відповідали технічним умовам виробництва

Трехкоордінатние системи автоматизованого проектування (об'ємного) ще знаходяться на стадії становлення. При створенні моделі взуття основний акцент робиться на отриманні малюнка, що відображає задум художника-модельєра. Конструювання ж вимагає геометрично точного зображення реальної моделі. Отримують зображальну інформацію і геометричні дані в трьохмірному уявленні. Якщо САПР здатна представляти зображальну інформацію, то систему називають ескізною, зображальною, графічною. Якщо САПР дозволяє отримувати геометричні дані, то її можна використати для управління роботою обладнання.

Так, отримані з допомогою САПР геометричні дані, необхідні для розрахунку траєкторії руху робочих органів взуттєвих машин, використовуються в системі автоматизованого виробництва. За рубежем використовується багато систем CAD, серед них: Gradamatic, Apex - фірми Camsco (США). У останніх варіантах цих систем виконують наступні операції:

- конструювання деталей;
- градирование деталей;
- оцінка моделей, аналіз економічності (вартість і трудомісткість);
- визначення площ;
- калькуляція витрати матеріалів і заробітної плати.

До числа найбільш відомих і поширених відносяться:

Система FDS Microdynamics (США)[1]

Система проектування взуття FDS розроблена як серія модулів. Завдяки своїй конструкції FDS може охоплювати 1000 робочих місць або АРМ. У FDS кожне робоче місце має свій комп'ютер і запам'ятовуючий пристрій. Все це тісно пов'язане з іншими робочими місцями і центром. Існує модель системи FDS -50 для двухразмерного моделювання нових фасонів взуття швидких зарисовок. Модель FDS -1500 дозволяє проектувати, конструювати, градировать і робити укладиваемость деталей в двомірних координатах. Модель FDS -300 і FDS -350 - трьохмірні проектуючі системи.

Система фірми Clarks

Система базується на автоматичному описі складної поверхні колодки з використанням графічних пристроїв з числовим управлінням. Роботи, проведені на фірмі, показали, що прийнята сьогодні система двухразмерного градирования приводить до значних витрат матеріалу. Тому фахівці фірми розробили систему, яка градирует заготівлі в трьох напрямах. Цей метод використовується у Франції. Фірма USM (Англія) створила систему CAD-CAM, яка дозволяє по числовій інформації за допомогою термінала отримувати креслення деталей. Фірма розробила три методи підготовки зразка.

У одному модельєр вичерчивает модель зразка в двухкоординатной сітці на моніторі ЕОМ, що включає телевізійний екран, креслярський програматор, що має в своїй пам'яті 16,7 млн різних колірних відтінків, ліній, дуг.

У другому методі дані про взуття вводяться в ЕОМ за допомогою телевізійної камери. Оператор може задати колір, комбінацію матеріалів і т. д. У третьому методі обробляється інформація в трьохмірному вимірюванні. Трьохмірний координатор знімає координати з колодки, ЕОМ тут же дає точну копію. На цьому трьохмірному зображенні колодки конструктор на екрані дисплея зображає деталі взуття. Далі конструктор використовує інформацію про матеріали. Комп'ютер розраховує площі деталей, визначає економічність і т. д.

Система фірми Lectra[2]

З середини 80-х років більше за 300 систем фірми Lectra працюють у взуттєвій промисловості Франції, Англії, ФРН, Італії, США. Таким же чином фірма Dic & Jordan спільно з Bata Engineering створила систему підготовки трьохмірних зразків, які використовуються для контролю виробів при виготовленні різаків і в інших операціях САМ. Система BATA конструює колодку і взуття також в трьохмірному вимірюванні.

З останніх розробок потрібно виділити Digiton (Канада), Crispin (Австрія), Сіхі (Франція).

Системи виконують наступні функції:

- введення даних про колодку;
- моделювання взуття в інтерактивному режимі;
- отримання розгортки і деталировка верху;
- градирование деталей;
- розміщення деталей взуття на шкірі;
- виготовлення шаблонів деталей верху взуття.

Процес закінчується виготовленням лекал деталей взуття за допомогою лазерного автомата.

Система Apex

Розроблена фірмою Camsco (США).

Система передбачає наступні операції:

- знімання інформації про перетин колодки;
- запис в цифровій формі відповідних ліній моделі, нанесених на колодці;
- розрахунок і отримання на екрані графічного дисплея умовних розгорток зовнішніх і внутрішніх бічних поверхонь колодки;
- розмноження деталей (градирование);
- виготовлення креслень і підготовка документації;
- вирізування шаблонів для моделювання.

Крім того, система може виконувати перехідні операції від антропометричних даних до розмірів деталей взуття. Вона забезпечує проектування взуття на основі способу жорсткої оболонки. Система по певному коду викликає з пам'яті машини УРК потрібного фасону, а також типові деталі верху. Потім на екрані графічного дисплея будується система координат, відкладаються по завданню конструктора-програміста необхідні кути розведення крил, в які автоматично вписується УРК, будуються контури деталей верху і відкладаються величини затяжної кромки і інші параметри. Система забезпечує проектування окремих деталей верху з припуском на обробку, а також креслення внутрішніх і проміжних деталей, серійну градирование деталей.

Універсальна автоматизована графічна система "КОМПАС-ГРАФІК" російської компанії "АСКОН" Система "КОМПАСА-ГРАФІК" розроблена спеціально для операційної середи MS Windows і в повній мірі використовує всі її можливості і переваги, надаючи користувачу максимальну ефективність і зручності в роботі. По функціях побудови і редагування креслення, виведення на друк, а також за основними інтерфейсними рішеннями практично повністю співпадає з сучасними пакетами САПР:

- підключені параметричні функції і функції асоціативності моделі креслення;
- вбудований текстовий редактор, що дозволяє створювати текстові документи з розміщенням в них графічних зображень;
- підтримуються бібліотеки фрагментів і вставки зовнішніх фрагментів в документ;
- підтримуються прикладні бібліотеки;
- підтримуються атрибути об'єктів;
- активізовані призначені для користувача панелі команд;
- підтримуються призначені для користувача стилі ліній, текстів, штриховок, основних написів креслення і т. д.;
- розширені сервісні можливості;

- є утиліти швидкого перегляду і створення драйверів векторних пристроїв;
- підтримується експорт документів в форматах DXF і IGES.

2. Структура і характеристика модульної САПР "Ірис"[3-4]

Програмний комплекс розроблений на кафедрі КТИК КНУТД в 1990-2005 рр. і за цей час впроваджений на багатьох взуттєвих підприємствах України різної форми власності.

По своєму характеру система є системою площинного проектування 2D.

Програмний комплекс складається з декількох програмних модулів, які об'єднуються під егідою головного меню:

- оцифровка контурів;
- проектування;
- градирование подетальное;
- градирование грунтів;
- вичерчивание картинок;
- розміщення шаблонів;
- каталог;
- інструкція користувача.

Структура комплексу

Структура побудови комплексу такого роду базується на двох основних принципах:

- спадкоємність і єдність з діючими структурами конструкторської підготовки взуттєвого виробництва;
- об'єднання як можна більшого числа задач в конструкторській підготовці виробництва.

Програмний комплекс призначений для проетирования деталей верху і низу взуття різних конструкцій. Аналіз чого склався структури процесу виробництва і підготовки проекту передбачає наявність в системі наступних базових програмних модулів:

- введення початкової інформації;
- проектування моделі і її деталей;
- серійне градирование контура деталей;
- контроль укладиваемости контура деталей;
- формування паспорта моделі.

Функції автоматизованого проектування на програмному комплексі "ІРИС"

Широке застосування методів автоматизованого проектування взуття і кожгалантерейних виробів передбачає використання послідовності цілеспрямованих функцій. Під функцією автоматизованого проектування в цьому випадку розуміємо специфічний вплив системи на об'єкт проектування, направлений на приведення його до вигляду, відповідного вимогам конкретного етапу розробки. У результаті реалізації необхідних функцій отримують комплект конструкторської документації, необхідний для запуску виробу у виробництво. Основу автоматизованого проектування складають прийоми звичайних (креслярських) методів, що отримали свій природний розвиток.

Жодна з сучасних систем не може вважатися завершеною, якщо вона не в змозі виконати хоч би одна вимога повсякденної практики розробки конструкцій взуття. Якісно новий рівень процесу проектування повинен включати досить широкий спектр додаткових можливостей, що надаються модельєру-конструктору. Тому цілком закономірним є розділення функцій автоматизованого проектування на традиційні і специфічні (машинні).

Традиційні функції формуються з набору функцій, необхідних для підготовки графічних і текстових документів. Графічні поділяються на функції прямої і послідовної дії. Перші характеризуються досягненням результату безпосередньо після їх ініціалізації. Функції послідовної дії вимагають введення додаткових даних, вказуючих на інтенсивність і область поширення. У

залежності від масштабу додатку функції прямої дії поділяються на параметричні, одиничні і структурних.

Використання параметричних функцій дозволяє визначати габарити об'єкта{G}(деталі, декількох деталей або вузлів, грунд-моделі), довжину окремих контурів{x}, периметр {AX}, площа {z}, відстань {X} між заданими точками, кут нахилу вибраних ліній {y}. Результати висвічуються на екрані або зберігаються в файлі. Одиничні функції прямої дії дають можливість перемістити будь-яку точку в нове положення {}, встановити першу точку {T}, видалити {U}, сдублировать {w} або вставити {W} нову точку на заданій відстані. Одночасно можна задати потрібний напрям {h} точок по замкненому контуру.

Структурні функції виконують дії над вказаними об'єктами (замкненими або розімкненими контурами, окремими лініями, вставками і т. п.). Зсув {F6} - переміщення об'єкта у вказаному напрямі відносно всього зображення.

Функції прямої дії ініціалізувалися однократним натисненням відповідної клавіші або переміщенням курсора в задану область екранного меню.

Функції послідовної дії передбачають багаторазове натиснення клавіш з вказівкою необхідних параметрів і поділяються на елементарні, об'єктні і функції формообразования. Елементарні функції впливають на окремі дільниці (елементи) контура. Згладжування {L} дозволяє вирівнювати значення координатних пар методом найменших квадратів у вказаному інтервалі.

Функція інтерполяції {I} дозволяє визначити бракуючі точки плавного контура за допомогою інтерполяційних многочленів (наприклад, сплайнов), зберігаючи при цьому координати вузлових (початкових) точок.

Функції припусків {e} і {v} призначені для проведення постійних (під строчку, загибку і т. п.) і змінних (затяжна кромка) припусков.

Об'єктні функції виконують перетворення з готовими фрагментами деталей. Функція з'єднання {s} дозволяє з'єднати два різних контури в одну деталь з привласненням імені.

Площа при цьому визначається по зовнішньому контуру.

Функція вставки {F} є однією з найбільш могутніх в програмному комплексі і дозволяє сформувати деталь з окремих дільниць декількох контурів. Корисно застосовувати цю функцію при введенні початкової інформації на дигитайзере, оскільки вельми скрутно ввести в комп'ютер абсолютно однаково співпадаючі контури двох і більш деталей. Для підготовки складального креслення моделі взуття із зовнішніми і внутрішніми берцами і задинками включають функцію {r}, що дозволяє розвертати деталь навколо заданої осі (лінії перегину).

Особливу групу складають функції формообразования, що дозволяють створювати окремі елементи моделі, наприклад, за допомогою малювання замкнених або розімкнених контурів {q}. Мітки для зборки або накол {i}, мітки для бло-чек {до}, для перфорації можуть виконуватися як окремі деталі {I},{До},{P} з привласненням імені. За допомогою функції геометричних фігур {F7} можна створювати кола, овали, багатокутники і лінії з різними геометричними параметрами (радіус, довжина, ширина) і під різними кутами. Машинні функції дозволяють створити особливу оболонку і надають певний сервіс модельєру, який може підібрати зручний режим роботи.

Функціями управління екраном масштабують зображення {F10}, зсувають його в зручну позицію, очищають екран {ESC}, викликають поточну підказку і меню {F1}. Сюди ж відноситься функція {M} вибору певної кількості деталей, що одночасно висвічуються на екрані (від однієї до всіх). Вона також міняє порядок виклику окремих деталей.

Функції управління файлами можна створювати або перейменовувати існуючий файл {F2}. Функція {F3} призначена для перегляду каталога моделей або файлів і з її допомогою можна з'єднувати в один, два і більше за файли, отримуючи необхідний набір деталей.

Функції управління друком дозволяють підібрати шрифт для виведення текстових, визначати розмір тексту і форматувати його по правому або лівому полю, будувати рамки таблиць. Функції управління графопостроителем (плоттер) дають можливість підбору кольору (пера) або типу лінії, вибирати масштаб і месторасположение тексту.

Практичний досвід експлуатації програмного комплексу "ІРИС" показує, що найбільш ефективні результати досягаються при роботі в певній послідовності:

коректування початкових контурів моделі. Контури повинні бути представлений:

- плавними замкненими кривими. Точки розташовуються за годинниковою стрілкою, частота їх залежить від кривизни дільниці. Здвоєні, строєні і т. д. точки не рекомендуються, їх кількість повинна бути мінімальною;
- побудова відрізних деталей. Початкові контури розрізаються на частині по зазделегідь відмічених точках;
- побудова суцільних деталей. Зовнішню сторону деталей грунд-моделі рекомендується розташовувати знизу, внутрішню - зверху;
- побудова технологічних припусков міток. Після установки припусков за допомогою вказаних функцій бажане невелике коректування в кінцевих точках;
- побудова внутрішніх і проміжних деталей. Їх будують, виходячи з контурів зовнішніх деталей. Після побудови віддаляються зайві або здвоєні точки;
- остаточна доробка контура. Перевіряються всі розроблені контури. Проставляються мітки для блочек, перфорації і т. п. Грунд-модель розвертається у вісь градирования;
- виведення на плоттер і збереження файлів. Розроблена модель записується в файл і може бути виведена на плоттер або принтер.

1.2. Сучасний стан впровадження автоматизованих систем проектування у взуттєве виробництво

Розвиток тривимірної графіки дозволив перевести пакети автоматизованого проектування в середовище, яке дозволяє оцінювати продукт, що проектується, до виготовлення дослідного зразка за допомогою фотореалістичної візуалізації.

На етапі розробки конкретної моделі взуття з'являється можливість розглянути її з усіх боків, при цьому колір і фактура деталей можуть бути перенесені з відповідних фотографій матеріалу. Тривимірні моделі дають можливість раціональніше використовувати трудові та фінансові ресурси, економити матеріали та пришвидшити процес проектування нових моделей.

Вже не перший рік взуттєва промисловість активно використовує останні досягнення 3D технологій для створення нової продукції. Просторове проектування взуття та шкіряних аксесуарів стало обов’язковим етапом при створенні колекцій. Набуває популярності 3D друк окремих деталей та виробів.

Серед САПР, які підтримують 3D-формат, найпопулярнішими є розробки Crispin фірми Delcam (Англія), ShoeMaster англійської фірми Clarks, Lectra systems від Romans Cad та ShoeMaker фірми Gerber System (США).

Комплекси автоматизованого проектування, що підтримують функцію тривимірної візуалізації надають можливість проектувати всі частини взуття і покращувати його ще до виготовлення макету.

Фірма Delcam є провідним постачальником CAD / CAM-рішень для взуттєвої промисловості в усіх країнах світу.

Програми сімейства Power Solution дозволяють вирішувати завдання з опрацювання дизайну, декорування і виготовлення всіх типів взуття. Але ціна програм сімейства Power Solution настільки висока, що придбати їх можуть тільки гіганти-виробники взуття та спеціалізовані фірми з виготовлення оснастки

для взуттєвої промисловості такі, як Nike (США), Clarks (Великобританія), Ecco (Данія), Eram (Франція), Feng Tay і Pou Chen (Тайвань), Azaleia (Бразилія), Apego і STM Meccanica (Італія) та ін..

Ціни на програмні продукти інших розробників спеціалізованих САПР теж дуже високі і недосяжні для середнього і, тим паче, малого бізнесу.

Щоб долучити нашого вітчизняного виробника до сучасних технологій просторового моделювання та проектування взуття необхідно розробити методику використання менш коштовних і більш доступних універсальних програм 3Д графіки, але які не мають специфічних функцій пов’язаних з проектуванням взуття.

За останні роки широкого розвитку досягли універсальні програми просторового моделюванню такі як - Zbrush від Pixologic; 3DS MAX від Autodesk; Cinema 4D від MAXON, які активно використовуються для створення спецефектів в кінематографії і рекламних роликах, а також дизайнерами по всьому світу.

Вони мають багато функцій і дозволяють робити не тільки реалістичні 3D моделі, но і їх анімацію, але вони досить коштовні і складні в освоюванні. Наряду з ними є більш доступні і простіші графічні редактори такі як Wings 3D, Daz Studio, AutoDesk 123D, Meshmixer 3.0, PTC Creo, Netfabb, MeshMagic або Blender від The Blender Foundation, які поширюються безкоштовно або за помірну плату.

Але всі ці вище наведені редактори базуються на створенні полігональних моделей, в яких просторова поверхня апроксимується плоскими трикутними (іноді чотирикутними) поверхнями. Такий спосіб моделювання підходить тільки для візуалізації об'єктів, але не для їх проектування, де необхідна точність і можливість отримати просторові координати будь-якої точок на поверхні.

Для проектування тривимірних об’єктів потрібен сплайновий метод просторового моделювання, який використовують такі програми як: AutoCAD

від Autodesk, SolidWorks від Dassault Systèmes та NX (раніше «Unigraphics») від Siemens PLM Software.

Всі ці програми дозволяють автоматизувати процес 2D та 3D моделювання і проектування для машинобудування, архітектури та промислового дизайну але не дуже зручні при моделюванні об'єктів, що мають складну, органічну форму, таких як взуттєва колодка чи взуття.

Для просторового моделювання виробів індустрії моди, і в тому числі взуття, доречно застосовувати доступний по ціні програмний продукт Rhinoceros фірми Robert McNeel &Associates, який використовує для моделювання сплайновий метод, а точніше NURBS технологію (NURBS від Non-Uniform Rational B-Spline, що означає неоднорідні раціональні B-сплайни).

Технологія NURBS-моделювання дозволяє досить просто створити просторову модель об'єктів, що мають складну форму і з достатньо великою точністю розрахувати геометричне положення кожної точки поверхні майбутньої моделі взуття і потім використовувати цю модель для проектування, виготовлення прототипу за допомогою 3D-принтерів чи пристроїв з ЧПУ.

Порівнявши методи просторового моделювання і характеристики сучасних комп'ютерних програм, було встановлено, що найбільш відповідає нашим потребам (а також користувачами малого і середнього бізнесу) графічний редактор Rhinoceros 3D, який успішно використовується в промисловому, ювелірному та автомобільному дизайні, швидкому прототипуванні, а також в мультимедіа та графічному дизайні.

Застосування комп'ютерної 3D графіки дає можливість побачити виріб ще до виготовлення прототипу з матеріалу, повернути і роздивитися зі всіх боків і підібрати для нього найбільш відповідні матеріали, колір і фактуру, а у разі необхідності відкоригувати контури і форму деталей. Для 3D моделювання взуття нами запропонована методика із застосуванням технологій NURBS моделювання у програмі Rhinoceros, яка включає наступні етапи:

• вивчення напрямку моди та аргументований вибір властивостей об'єкту (призначення, форма, габарити, основні матеріали, поділ на деталі, конструктивні особливості та ін.);

• поділ об'єкта по шарам на деталі або частини з загальними властивостями або ті, що виготовляються з одного матеріалу;

• побудова або імпорт вже готової моделі колодки, яка визначає форму і габарити основних деталей;

• креслення на моделі колодки стильових і, якщо потрібно, допоміжних ліній; • попередня побудова поверхонь і форм основних деталей;

• уточнення конфігурації деталей і форми об'єкта (в разі необхідності, деформація, обрізка чи об’єднання тривимірних поверхонь, які утворюють деталі); • моделювання та додавання фурнітури або дрібних і додаткових деталей; • додавання кольору і текстури шарам і деталям;

• розгортання 3D деталей на поверхні, щоб одержати плоскі шаблони; • додавання технологічних припусків;

• остаточна презентація моделі - створення мізансцени, додавання оточуючих або додаткових об'єктів, освітлення.

Висновки до першого розділу

1. Постійно зростаючі вимоги до дизайну взуття, його якості при одночасній необхідності скорочення термінів розробки нових моделей і їхнього запуску в серійне виробництво, а також утримання цін на конкурентоздатному рівні в буквальному значенні, змушують виробників впроваджувати новітні інформаційні технології на всіх етапах проєктування та виготовлення взуття.
2. Розробка математичного та програмного забезпечення для проєктування декоративних елементів на деталях взуття є однією із задач інформаційних технологій на етапі проєктування взуття.

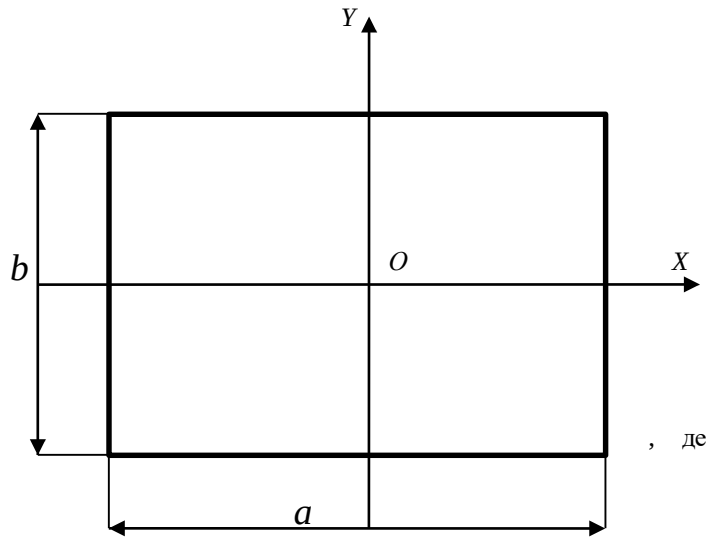
2. РОЗРОБКА МАТЕМАТИЧНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ПРОЄКТУВАННЯ ДЕКОРАТИВНИХ ЕЛЕМЕНТІВ НА ДЕТАЛЯХ ВЗУТТЯ

2.1. Генерування одинарних декоративних елементів

Розглянемо наступні деталі простої конфігурації: прямокутник, ромб, хрест, *N*-кутник, коло, крапля, еліпс, зірка, трикутник, паралелограм, трапеція[5- 7].

Прямокутник. Для однозначного відображення прямокутника необхідно задати координати його вершин.

N	X_i	Y_i
1	a/2	a/2
2	-a/2	a/2
3	-a/2	-a/2
4	a/2	-a/2
5	a/2	a/2



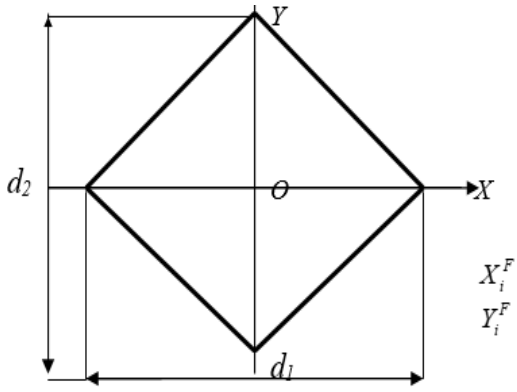
$$X_i^F = X_i Cos\theta - Y_i Sin\theta + Xp$$
$$Y_i^F = X_i Sin\theta + Y_i Cos\theta + Xp$$

, де $i = 1,2,...5$

Рис. 2.1. Розрахунок координат вершин прямокутника

Для визначення координат вершин необхідно знати координати полюса прямокутника *Xp,Yp,* довжину a та ширину b прямокутника та кут повороту прямокутника θ (рис. 2.1).

Ромб. Для однозначного відображення ромба необхідно задати координати його вершин. Для визначення координат вершин необхідно знати координати полюса ромба *Xp,Yp,* довжину діагоналей d_1 та d_2 ромба та кут повороту ромба θ (рис.2).

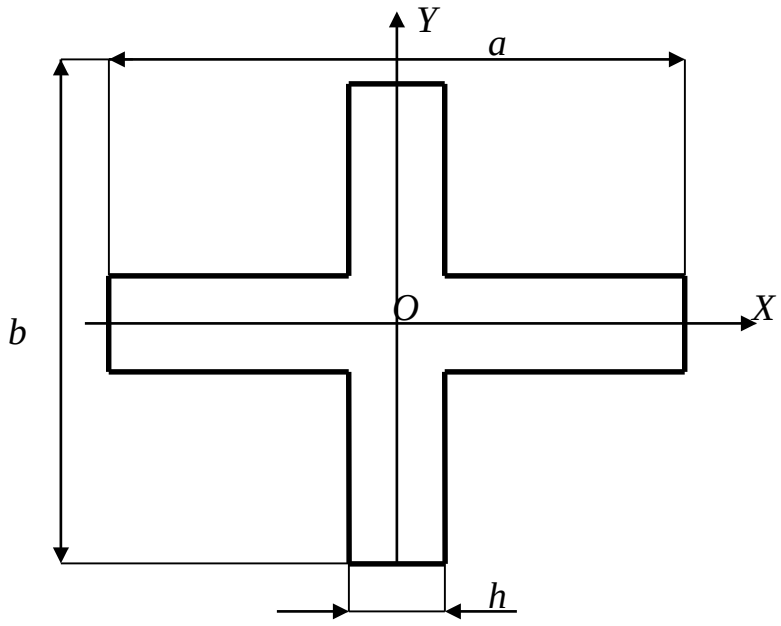


N	X_i	Y_i
1	$d_1/2$	0
2	0	$D_2/2$
3	$-d_1/2$	0
4	0	$-d_2/2$
5	$d_1/2$	0

$$X_i^F = X_i Cos\theta - Y_i Sin\theta + Xp$$
$$Y_i^F = X_i Sin\theta + Y_i Cos\theta + Xp, \quad \text{де } i = 1,2...5$$

Рис. 2.2. Розрахунок координат вершин ромба

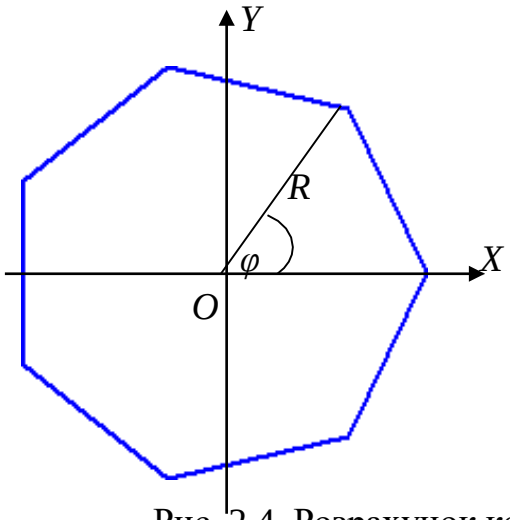
Хрест. Для однозначного відображення хреста необхідно задати координати його вершин. Для визначення координат вершин необхідно знати координати полюса хреста *Xp*,*Yp*, довжину *a*, ширину *b*, товщину хреста *h* та кут повороту хреста θ (рис. 2.3).



N	X_i	Y_i
1	h/2	h/2
2	h/2	b/2
3	-h/2	b/2
4	-h/2	h/2
5	-a/2	h/2
6	-a/2	-h/2
7	-h/2	-h/2
8	-h/2	-b/2
9	h/2	-b/2
10	h/2	-h/2
11	a/2	-h/2
12	a/2	h/2
13	h/2	h/2

Рис. 2.3. Розрахунок координат вершин хреста

N-кутник. Для однозначного відображення *N*-кутника необхідно задати координати його вершин. Для визначення координат вершин необхідно знати координати полюса *N*-кутника *Xp,Yp*, *N*-кількість сторін многокутника, радіус описаного кола *R* та кут повороту *N*-кутника θ[9] (рис. 2.4).



$$\varphi_i = 2i\pi / N$$
$$X_i = R \cdot Cos\varphi_i \quad \text{де } i = 0,1,2...N$$
$$Y_i = R \cdot Sin\varphi_i$$

$$\varphi = 2\pi / N$$

$$X_i^F = X_i Cos\varphi_i - Y_i Sin\varphi_i + Xp$$
$$Y_i^F = X_i Sin\varphi_i + Y_i Cos\varphi_i + Yp \quad \text{, де } i=0,1,2..N$$

Рис. 2.4. Розрахунок координат вершин N-кутника

Коло. Для однозначного відображення кола необхідно задати координати вершин *N*-кутника, що апроксимує коло із заданою точністю ε. Для визначення координат вершин апроксимуючого *N*-кутника необхідно знати координати полюса кола *Xp,Yp*, *N*-кількість сторін апроксимуючого многокутника, радіус кола *R* (рис.5.). Кількість сторін *N* апроксимуючого многокутника залежить від радіуса кола *R* та допустимої точності ε (рис. 6).

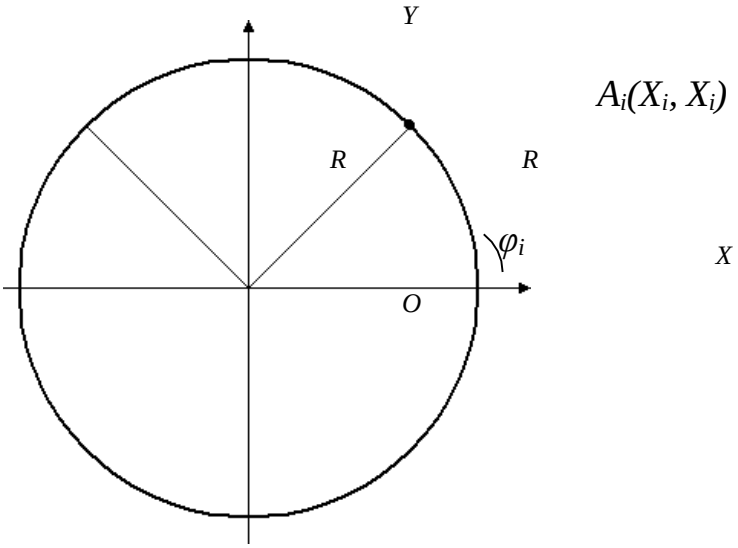
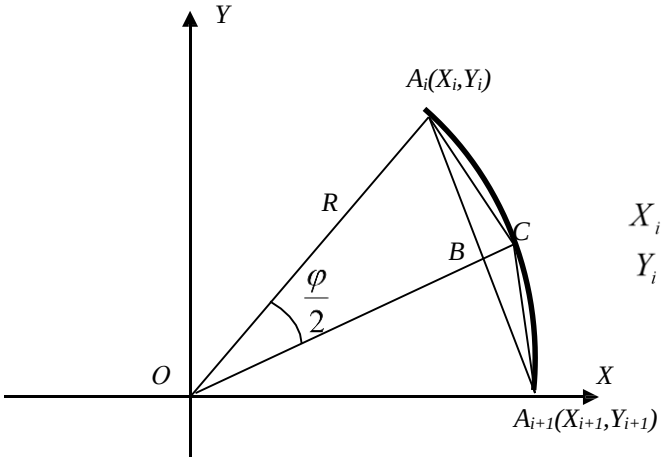


Рис. 2.5. Коло

Розглянемо ΔOA_iB. Так як $OA_i=R$ та ΔOA_iB -прямокутний, то $|OB| = R \cdot Cos\frac{\varphi}{2}$.



$$\varphi = 2\pi / N$$

$$\varphi_i = i \cdot \varphi$$

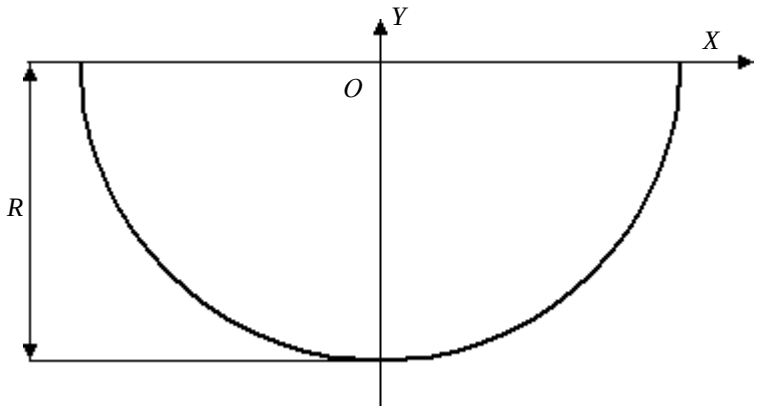
$$X_i = R \cdot Cos\varphi_i$$
$$Y_i = R \cdot Sin\varphi_i$$, де $i= 0,1,2..N$

Рис. 6. Розрахунок кількості сторін *N* апроксимуючого многокутника

Тоді $|BC| = R - R \cdot Cos\frac{\varphi}{2} \le \varepsilon$, або $R - R \cdot Cos\frac{2\pi}{2N} \le \varepsilon$, або $Cos\frac{\pi}{N} \ge 1 - \frac{\varepsilon}{R}$. Звідси маємо[10-12]:

$$\frac{\pi}{N} \le arcCos(1 - \frac{\varepsilon}{R}) \text{ або } N \ge \frac{\pi}{arcCos(1 - \frac{\varepsilon}{R})} \qquad (2.1).$$

Півколо. Для однозначного відображення півкола необхідно задати координати вершин *N*-кутника, що апроксимує півколо (рис.7.) із заданою точністю *ε*. Для визначення координат вершин апроксимуючого *N*-кутника необхідно знати координати полюса півкола *Xp,Yp*, *N*-кількість сторін апроксимуючого многокутника, радіус півкола *R*. Кількість сторін *N*



Координати вершин:

$$X_i = R \cdot Cos\varphi_i$$
$$Y_i = R \cdot Sin\varphi_i$$, де $i= 0,1,2..N$

$\varphi = \pi / N$ та $\varphi_i = \pi + i \cdot \varphi$

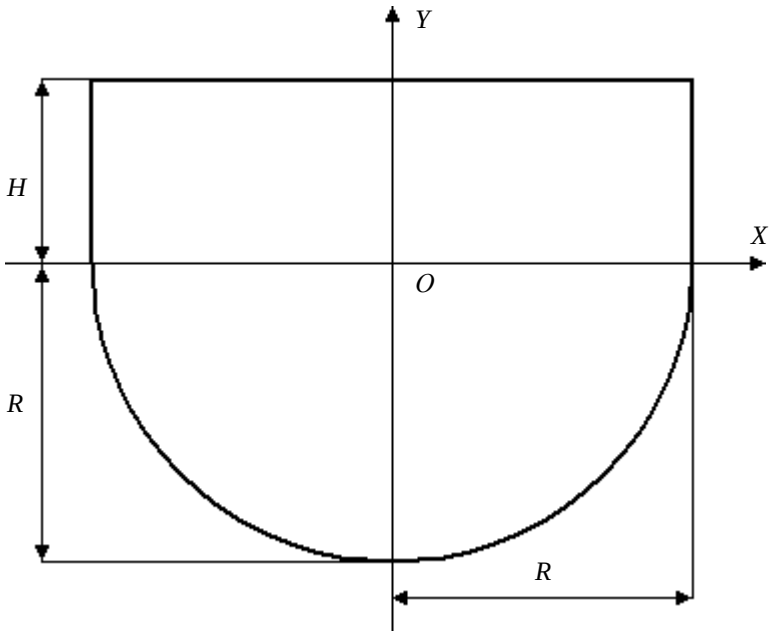
$X_{N+1} = X_0$

$Y_{N+1} = Y_0$

апроксимуючого многокутника залежить від радіуса кола *R* та допустимій точності ε та визначається наступним чином

$$N \geq \frac{\pi}{2arcCos(1-\frac{\varepsilon}{R})}. \qquad (2.2)$$

Півколо+Прямокутник. Для однозначного відображення півкола+прямокутника необхідно задати координати вершин *N*-кутника, що апроксимує півколо+прямокутник (рис.8) із заданою точністю ε. Для визначення координат вершин апроксимуючого *N*-кутника необхідно знати координати полюса півкола *Xp*,*Yp*, *N*-кількість сторін апроксимуючого многокутника, радіус півкола *R* та висоту прямокутника *H*. Кількість сторін *N* апроксимуючого многокутника залежить від радіуса кола *R* та допустимої точності ε та визначається виразом (2.2)[13-15].



Координати вершин:

$$\begin{aligned} X_i &= R \cdot Cos\varphi_i \\ Y_i &= R \cdot Sin\varphi_i \end{aligned}, \text{ де } i = 0,1,2..N$$

$$\varphi = \pi / N \text{ та } \varphi_i = \pi + i \cdot \varphi$$

$$\begin{aligned} X_{N+1} &= X_N \\ Y_{N+1} &= Y_N + H \end{aligned}$$

$$\begin{aligned} X_{N+2} &= X_0 \\ Y_{N+2} &= Y_0 + H \end{aligned}$$

$$\begin{aligned} X_{N+3} &= X_0 \\ Y_{N+3} &= Y_0 \end{aligned}.$$

Рис. 2.8. Півколо+Прямокутник

Еліпс. Для однозначного відображення еліпса (рис.9) необхідно задати координати вершин *N*-кутника, що апроксимує еліпс із заданою точністю *ε*. Для визначення координат вершин апроксимуючого *N*-кутника необхідно знати координати полюса еліпса *Xp,Yp*, *N*-кількість сторін апроксимуючого многокутника, піввісі еліпса *a* та *b*. Кількість сторін *N* апроксимуючого многокутника залежить від $R=\max(a,b)$ та точності *ε* (2). Координати вершин апроксимуючого многокутника для еліпса із піввісями *a* та *b* та координатами полюса *Xp,Yp* обчислюються наступним чином[16]:

$$X_i^F = X_i + Xp = a \cdot Cos\varphi_i + Xp \text{ , де } \varphi = \frac{\pi}{N} i,\ i = 0,1,2...N \qquad (2.3)$$

$$Y_i^F = Y_i + Yp = b \cdot Sin\varphi_i + Yp$$

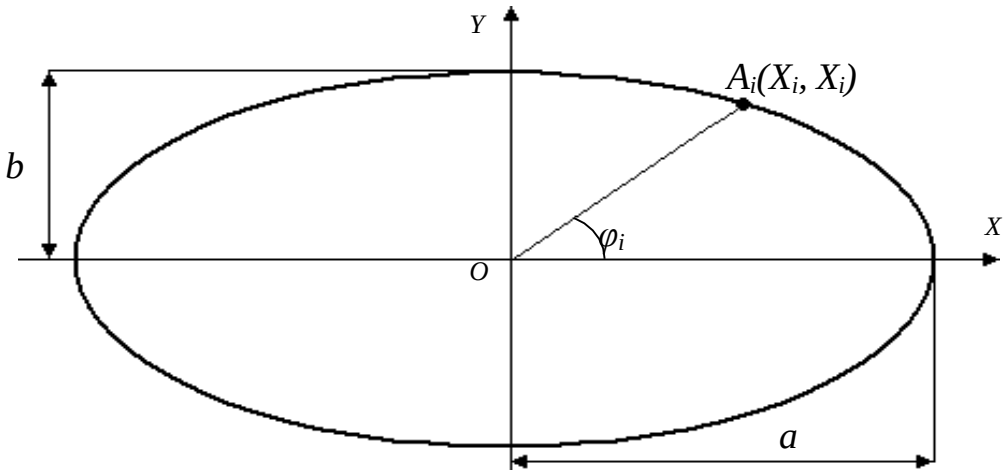
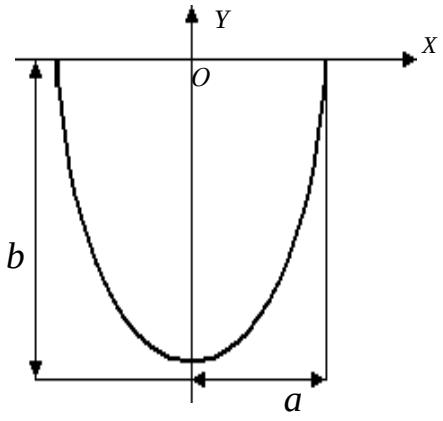


Рис. 2.9. Еліпс.

Півеліпс. Для однозначного відображення півеліпса (рис.2.10) необхідно задати координати вершин *N*-кутника, що апроксимує півеліпс із заданою точністю *ε*. Для визначення координат вершин апроксимуючого *N*-кутника необхідно знати координати полюса півеліпса *Xp,Yp*, *N*-кількість сторін апроксимуючого многокутника, піввісі еліпса *a* та *b*. Кількість сторін *N* апроксимуючого многокутника для півеліпса залежить від $R=\max(a,b)$ та точності *ε* (2.2). Координати вершин апроксимуючого многокутника для півеліпса із

піввісями a та b та координатами полюса Xp,Yp обчислюються наступним чином(4):

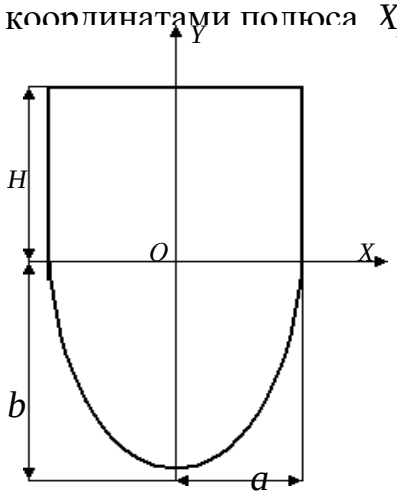


Координати вершин:

$$X_i^F = X_i + Xp = a \cdot Cos\varphi_i + Xp$$
$$Y_i^F = Y_i + Yp = b \cdot Sin\varphi_i + Yp$$
$$\text{де } \varphi_i = \pi + \frac{\pi}{N} i \text{ та } i = 0,1,2...N \quad (2.4)$$
$$X_{N+1}^F = X_{N+1} + Xp = X_0 + Xp$$
$$Y_{N+1}^F = Y_{N+1} + Yp = Y_0 + Yp$$

Рис. 2.10. Півеліпс.

Півеліпс+Прямокутник. Для однозначного відображення півеліпса+прямокутника (рис.11) необхідно задати координати вершин N-кутника, що апроксимує півеліпс із заданою точністю ε. Для визначення координат вершин апроксимуючого N-кутника необхідно знати координати полюса півеліпса Xp,Yp, N-кількість сторін апроксимуючого многокутника, піввісі еліпса a та b та висоти прямокутника H. Кількість сторін N апроксимуючого многокутника для півеліпса+прямокутника залежить від $R=\max(a,b)$ та точності ε (2.2). Координати вершин апроксимуючого многокутника для півеліпса+прямокутника із піввісями a та b та координатами полюса Xp,Yp обчислюються наступним чином (2.5) :

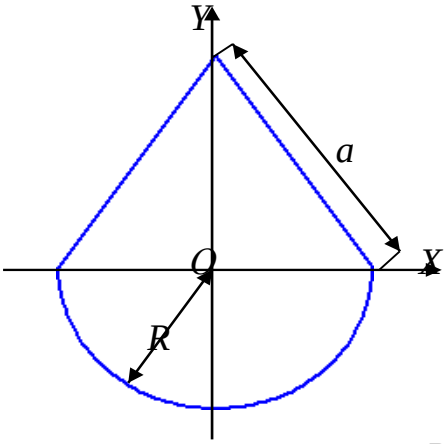


Координати вершин:

$$X_i^F = X_i + Xp = a \cdot Cos\varphi_i + Xp$$
$$Y_i^F = Y_i + Yp = b \cdot Sin\varphi_i + Yp$$
$$\text{де } i = 0,1,2...N \text{ та } \begin{matrix} X_i^F = X_i + Xp = a \cdot Cos\varphi_i + Xp \\ Y_i^F = Y_i + Yp = b \cdot Sin\varphi_i + Yp \end{matrix}$$
$$X_{N+1}^F = X_{N+1} + Xp = X_N + Xp$$
$$Y_{N+1}^F = Y_{N+1} + Yp = Y_N + H + Yp$$
$$X_{N+2}^F = X_{N+2} + Xp = X_0 + Xp \quad (2.5)$$
$$Y_{N+2}^F = Y_{N+2} + Yp = Y_0 + H + Yp$$
$$X_{N+3}^F = X_{N+3} + Xp = X_1 + Xp$$

Рис. 2.11. Півеліпс+Прямокутник

Крапля. Крапля буде складатися із двох фігур: півкола із радіусом *R*, та рівнобедреного трикутника із боковою стороною *a* та основою *2R*. Для однозначного відображення краплі необхідно задати координати вершин *N*-кутника, що апроксимує півколо із заданою точністю *ε*. Кількість сторін *N* апроксимуючого многокутника залежить від радіуса півкола *R* та допустимої точності *ε*. В нашому випадку *N* визначається виразом(2.2)[17-18],



Координати вершин:

$$X_i^F = R \cdot Cos(\pi + i\varphi) + Xp$$
$$Y_i^F = R \cdot Sin(\pi + i\varphi) + Yp \qquad (2.6)$$

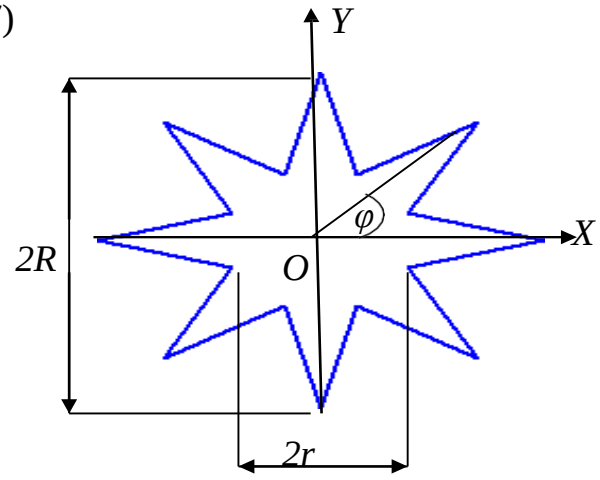
де $\varphi = \pi / N$ та $i = 0,1,2..N$

$$X_{N+1}^F = Xp; \qquad Y_{N+1}^F = Xp + \sqrt{a^2 - R}$$
$$X_{N+2}^F = Xp + X_0^F; \quad Y_{N+2}^F = Xp + Y_0^F$$

Рис. 2.12. Крапля

Зірка. Для однозначного відображення зірки необхідно задати координати її вершин. Для визначення координат вершин зірки необхідно знати кількість вершин зірки *N*, координати полюса зірки *Xp*,*Yp*, радіус кола *R*, що описане навколо зірки, радіус кола *r*, на якому лежать впадини зірки (рис. 2.13).

Тоді координати вершин зірки можуть бути представлені наступним чином (7)

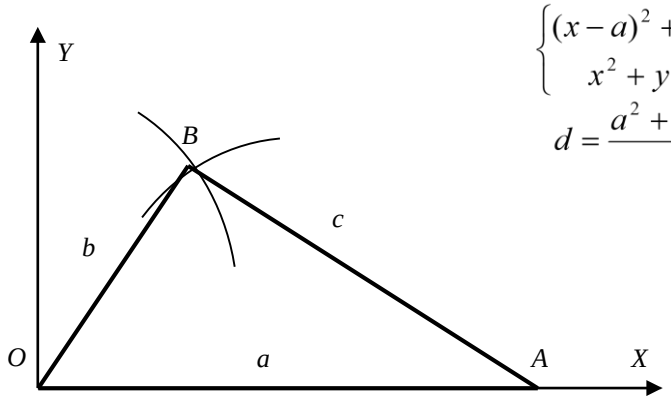


Координати вершин:

$$X_{2i}^F = R \cdot Cos(i \cdot \varphi) + Xp$$
$$Y_{2i}^F = R \cdot Sin(i \cdot \varphi) + Yp$$
$$X_{2i+1}^F = R \cdot Cos(i \cdot \varphi + \varphi/2 + Xp$$
$$Y_{2i+1}^F = R \cdot Sin(i \cdot \varphi + \varphi/2) + Yp \qquad (2.7)$$
$$X_{2N+2}^F = X_0^F$$
$$Y_{2N+2}^F = Y_0^F \ ,$$

Рис. 2.13. Основні параметри зірки

Трикутник: Основні параметри трикутника представлені на рис. 2.14, де координати точки $B(X_в, Y_в)$ знаходимо розв'язавши систему рівнянь: знаходимо розв'язавши систему рівнянь:



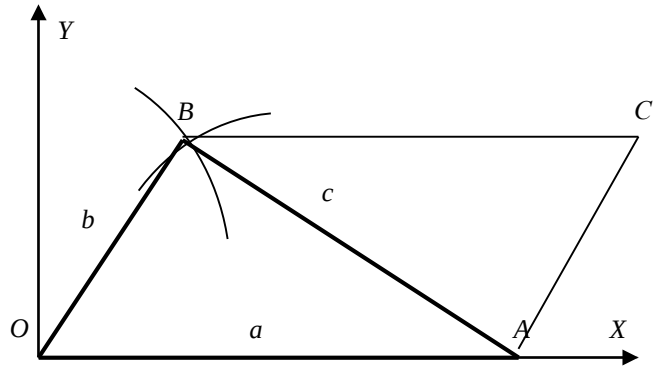
$$\begin{cases} (x-a)^2 + y^2 = a^2 \\ x^2 + y^2 = b^2 \end{cases}$$

$$d = \frac{a^2 + b^2 - c^2}{2a}$$

N	X_i	Y_i
1	0	0
2	a	0
3	d	$\sqrt{b^2 - d^2}$
4	0	0

Рис. 2.14. Основні параметри трикутника

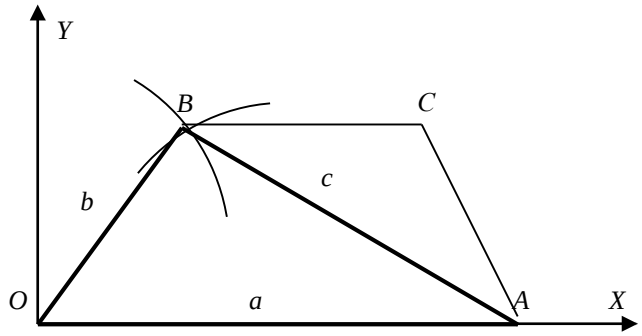
Паралелограм: Основні параметри паралелограма представлені на рис. 2.15. Координати вершин паралелограма представлені в таблиці, де координати точки $B(X_в, Y_в)$ знаходимо аналогічно попередньому випадку.



N	X_i	Y_i
1	0	0
2	a	0
3	d+a	$\sqrt{b^2 - d^2}$
4	d	$\sqrt{b^2 - d^2}$
5	0	0

Рис. 2.15. Основні параметри паралелограма

Рівнобедрена трапеція: Основні параметри рівнобедреної трапеції представлені на рис. 2.16. Координати вершин рівнобедреної трапеції представлені в таблиці:



N	X_i	Y_i
1	0	0
2	a	0
3	a-d	$\sqrt{b^2-d^2}$
4	d	$\sqrt{b^2-d^2}$
5	0	0

Рис. 2.16. Основні параметри рівнобедреної трапеції

На деталі взуття може бути нанесено багато декоративних елементів складної форми. Тому зберігати зовнішні контури декоративних елементів не раціонально, краще зберігати параметри цих елементів, по яким однозначно можна відтворити декоративні елементи, Такі параметри можна розбити на два типи:

- параметри, що відповідають за місце розташування декоративного елемента (вони стандартні для кожного декоративного елемента) та параметри декоративного елементу, які однозначно визначають зовнішній контур декоративного елементу.

До параметрів, що відповідають за місце розташування декоративного елемента можна віднести наступні:

PD - признак деталі, на який наноситься декоративний елемент;

PE - признак декоративного елемента;

Xp,Yp - координати полюса декоративного елементу відносно деталі;

θ – кут повороту декоративного елементу відносно його основного положення.

Параметри декоративних елементів представлені в таблиці 2.1.

Таблиця 2.1

Параметри декоративних елементів

PE	Назва декоративного елементу	Параметри декоративного елементу
1	Прямокутник	Сторони прямокутника *a* та *b*
2	Ромб	Діагоналі ромба d_1 та d_2
3	Хрест	Довжина *a*, ширина *b*, товщина хреста *h*
4	*N* - кутник	*К*ількість сторін многокутника *N*, радіус описаного кола *R*
5	Коло	Радіус кола *R*
6	Еліпс	Піввісі еліпса *a* та *b*
7	Крапля	Сторона *a* та радіус півкола *R*
8	Зірка	Кількість сторін зірки *N*, радіус описаного кола *R* та радіус внутрішнього кола *r*

2.2. Генерування групових декоративних елементів

Під груповими декоративними елементами ми будемо мати на увазі однакові декоративні елементи розміщені на колі радіуса *R* таким чином, що відстань між сусідніми груповими елементами по дузі цього кола однакові. Тоді $A_iA_{i+1}=A_{i+1}A_{i+2}=...=A_pA_{p+1}=2r+\Delta$, де Δ- відстань між зовнішніми границями двох кіл, що описані навколо двох сусідніх групових декоративних елементів[17-19].

Задачу про побудову групових елементів можна сформулювати наступним чином (рис. 2.17).

Дано: точка $O(X_o,Y_o)$, радіус кола $R=OQ$, до якого дотикаються групові елементи, кількість *N* групових елементів у групі.

Знайти: координати полюса $A_i(X_i,Y_i), i=0,1,2..N-1$ для кожного із групових елементів, радіус кола $r=QA_p$, описаного навколо групового елемента, кут між двома сусідніми елементами $\angle A_iOA_{i+1} = \angle\beta$.

Розглянемо ΔA_iOA_{i+1} (рис. 2.17). Він рівнобедрений, т.я. $OA_i=OA_{i+1}=R_o$ Тоді $A_iA_{i+1}=R_o/sin(\beta/2)$ та $\angle\beta = 2\pi/N$. Звідси маємо $2r+\Delta=R_o/sin(\beta)$. Але $QO=R=R_o+r$, або $R_o=R-r$. Тоді $2r+\Delta= (R-r)\ sin(\beta/2)$, або $r=(R-\Delta)/(2+sin(\beta/2))$.

Координати полюсів $A_i(X_i,Y_i), i=0,1,2..N-1$ групових декоративних елементів будуть визначатись наступним чином:

$$X_i=X_o+R_o cos(\beta/2+i\beta)$$

$$Y_i =Y_o+R_o sin(\beta/2+i\beta).$$

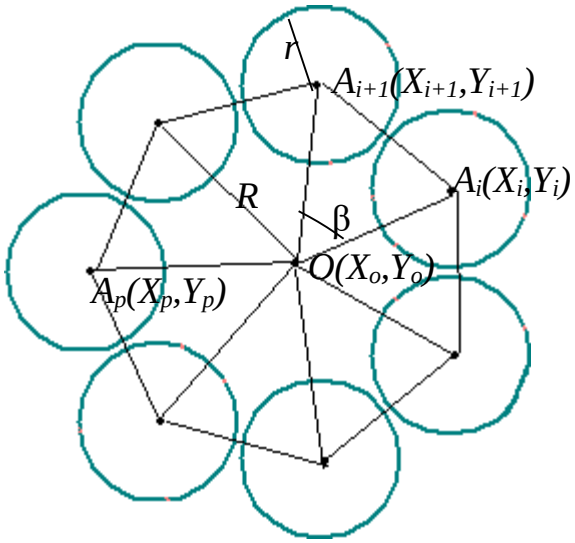


Рис. 2.17. Модель побудови групових декоративних елементів

Нехай базовий елемент для заданої групи елементів має координати: $\{Xв_j,Yb_j\}$, де $j=1,2,...Nb$. Тоді координати $\{Xq_i^j,Yq_i^j\}$ зовнішнього контуру i-го групового елементу будуть визначатись наступним чином:

$$\begin{cases} Xq_i^j = X_i + Xb_j \cos(\beta/2 + i\cdot\beta) - Yb_j \sin(\beta/2 + i\cdot\beta) \\ Xq^j = X_i + Xb_j \cos(\beta/2 + i\cdot\beta) - Yb_j \sin(\beta/2 + i\cdot\beta) \end{cases}, \qquad (2.8)$$

де $i=0,1..N, j=1,2,...Nb$.

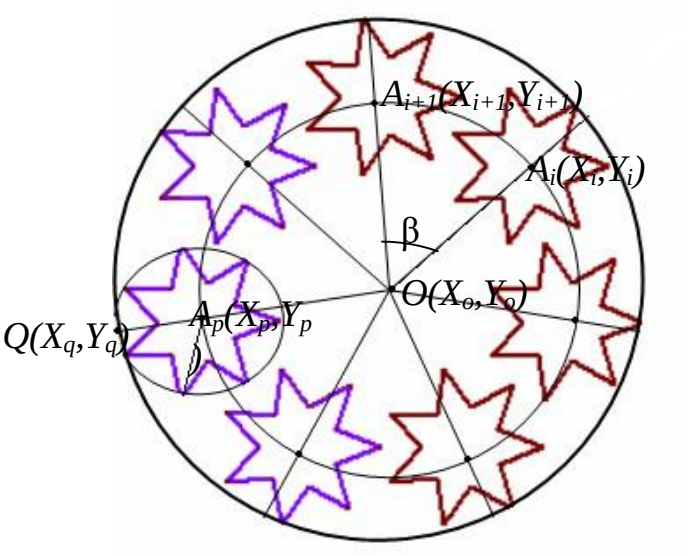
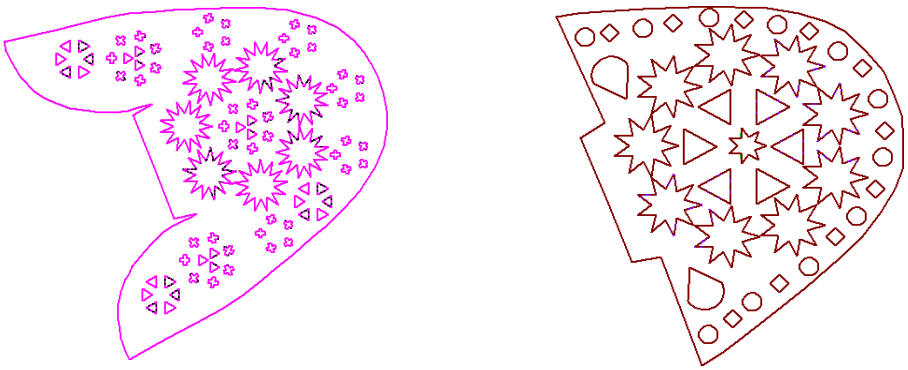


Рис. 2.18. Побудова групових декоративних

Всі розглянуті задачі реалізовані в програмному продукті в середовищі програмування Delphi для операційної системи Windows (додаток Г). Розроблене програмне забезпечення, дозволяє автоматизувати побудову декоративних елементів на деталях взуття. Приклад побудованих групових декоративних елементів на деталях взуття представлені на рис. 2.19.а, приклад комбінації групових та одиночних елементів представлені на рис. 2.19.б.



а) *б)*

Рис.2.19. Комбінації групових та одиночних елементів

а) Групові декоративні елементи; *б) Комбінація групових та одиночних елементів*

При проектуванні схем розкрою для деталей з декоративними елементами ми не будемо враховувати декоративні елементи, а враховувати їх ми будемо тільки на етапі візуалізації розкрійної схеми.

Для візуалізації декоративних елементів на деталях, що входять у спроектовану схему розкрою, ми згенеруємо ці елементи та відобразимо їх аналогічно деталям. Інформацію про декоративні елементи на деталі беремо у файлі *.Res, що був створений при проектуванні декоративних елементів на деталях взуття.

Так як декоративні елементи в більшості випадків представляють собою технологічні отвори на деталях взуття, то розв’язок такої задачі дозволяє впровадити розкрій таких деталей за допомогою роботизованих комплексів.

Висновки до другого розділу

1. В роботі запропоновані параметричні моделі одинарних декоративних елементів на деталях взуття.
2. В роботі запропоновані параметричні моделі групових декоративних елементів на деталях взуття.
3. На основі параметричних моделей одинарних та групових декоративних елементів на деталях взуття розроблені алноритми цих генерування на деталях взуття враховуючи параметрами цих елементів.

3. РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ПРОЄКТУВАННЯ ДЕКОРАТИВНИХ ЕЛЕМЕНТІВ НА ДЕТАЛЯХ ВЗУТТЯ

3.1. Вимоги до програмного продукту

Програмний продукт повинен забезпечувати[20]:

- 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. Програмні засоби

Для реалізації даного програмного продукту необхідні наступні програмні засоби: Delphi 7.0[21-22].

Цей вибір обґрунтовано тим, що використання даних інструментальних засобів дозволяє створити інформаційну систему для обробки і аналізу необхідного взаємопов’язаного набору даних.

Як операційна система була обрана ітерактивна оболонка Microsoft Windows – 2010, що дозволяє підтримувати режим багатозадачності, а також на відзнаку від відомих операційних систем Unix, OS/2 та ін. Windows забезпечує високу якість зображення. Вона володіє розширеними можливостями при паралельній роботі декількох програм, спрощеною передачею результатів від програми до програми, підвищеною ємністю системи, дозволяє водночас виконувати велике число прикладних програм і системних компонентів, ефективними засобами управління оперативною пам'яттю і іншими можливостями, що відрізняють її від вищезгаданих операційних систем.

Delphi 7.0 – це програмний інструмент для створення прикладних програм, які працюють в середовищі операційної системи Microsoft Windows – 2010. Пакет включає не тільки мову програмування, а й ефективне діалогове середовище для розробки форм та вікон, що дозволяє програмісту створити програмний продукт достатньо високої якості за порівняно невеликий проміжок часу. Delphi 7.0 дає можливість використовувати ресурси інших прикладних програм Windows та взаємодіяти з зовнішньою базою даних.

3.3. Опис основних процедур розробленого програмного продукту, які забезпечують генерування декоративних елементів на деталях взуття

Процедура *TForm1.Open1Click* забезпечує введення інформації про зовнішні контури деталей моделі взуття для нанесення декоративних елементів на них[23-26].

```
procedure TForm1.Open1Click(Sender: TObject);
Var x1,x2,y1,y2:integer;
St:string[10];
nn,Rost:array[1..Kold]of word;
i,p,l:integer;

Procedure Vvod;
Var F,Fr:System.Text;
NameF:string;
xi,yi,xe,ye,Xc,Yc,x1,y1,x2,y2:Int64;
i,j,l:word;
nxi:integer;
Xrab,Yrab,A:real;
begin
{  Form1.Left:=0; Form1.Top:=0;}
if OpenDialog1.Execute then
begin
FileName:=OpenDialog1.FileName;
System.Assign(F,FileName);
Reset(F);
Readln(F,Model);
Readln(F,A);
Readln(F,KolDet);
For i:=1 to KolDet do
Readln(F,NameDet[i]);
For i:=1 to KolDet do
begin
Readln(F,KolPointDet[i]);
nn[i]:=KolPointDet[i];
end;
For i:=1 to KolDet do
begin
For j:=0 to nn[i]-1 do
begin
readln(f,Xrab,Yrab);
If KolPointDet[i]<>nn[i] then continue;
if (Xrab=999)and(Yrab=999) then
begin
KolPointDet[i]:=j;
continue
end;
xd[i,j]:=Round(Xrab*100);
```

```
yd[i,j]:=Round(Yrab*100);
{    BitBtn1.Caption:=IntToStr(i)+' '+IntToStr(j);}
end;
//nn[i]:=KolPointDet[i];
MaxMin(nxi,KolPointDet[i],xd[i],xi,-1);
MaxMin(nxi,KolPointDet[i],yd[i],yi,-1);
MaxMin(nxi,KolPointDet[i],xd[i],xe,1);
MaxMin(nxi,KolPointDet[i],yd[i],ye,1);
Xc:=Round((xe+xi)/2);
Yc:=Round((ye+yi)/2);
Delta_X[i]:=Round((Xe-Xi)/2);
Delta_Y[i]:=Round((Ye-Yi)/2);
for j:=0 to KolPointDet[i]-1 do
begin
xd[i,j]:=xd[i,j]-Xc;
yd[i,j]:=yd[i,j]-Yc;
end;
Skv(KolPointDet[i],xd[i],yd[i],S_det[i]);
end;
end;
System.Close(F);
end;
Begin
ScrollBox1.Visible:=True;
GroupBox1.Visible:=True;
GroupBox3.Visible:=True;
Vvod;

P:=0;
L:=Length(FileName);
For i:=L downto 1 do
If FileName[i]='.' then
begin
P:=i;
break;
end;
If P=0 then  FName:=FileName
else
For i:=1 to P-1 do FName:=FName+FileName[i];
FName:=FName+'.Res';
System.Assign(f,FName);
Repeat
```

```
{$I-}
Reset(f);
If IOResult<>0 then
begin
Nres:=0;
break
end;
Readln(f,Nres);
For i:=1 to NRes do
With Res[i] do
Readln(f,Nd,Ndek,A,B,D,Alf,Nr,Xcur,Ycur);
System.Close(f);
Until true;
{$I+}
Image2.Height:=55*KolDet+10;
X1:=5;
X2:=55;
Scala(50,50,Dxy);
With Form1.Image2.Canvas do
For i:=1 to koldet do
begin
Y1:=5+55*(i-1);
Y2:=55*i;
Pen.Color:=ClBlack;
Brush.Color:=ClWhite;
Rectangle(x1,y1,x2,y2);
Grd(KolPointDet[i],Xd[i],Yd[i],0,0,30,55*i-25,Dxy,i);
end;
end;
```

Процедура *TForm1.N1Click* забезпечує введення інформації про нанесені декоративні елементи для кожної із деталей моделі взуття у *.res.

```
rocedure TForm1.N1Click(Sender: TObject);
Var FName:string;
Var i,l,p:integer;
begin
P:=0;
L:=Length(FileName);
For i:=L downto 1 do
If FileName[i]='.' then
begin
P:=i;
```

```
break;
end;
If P=0 then  FName:=FileName
else
For i:=1 to P-1 do FName:=FName+FileName[i];
FName:=FName+'.Res';
System.Assign(f,FName);
Rewrite(f);
Writeln(f,NRes);
For i:=1 to NRes do
With Res[i] do
Writeln(f,Nd:6,Ndek:6,A:12,B:12,D:12,Alf:6,Nr:6,Xcur:12,Ycur:12);
System.Close(f);
end;
```

Процедура *Graph_im1* забезпечує вивід зображення многокутника на область *Image1*.

Параметри процедури:

- *col*-номер кольору:

```
1:  pen.Color:=ClRed;
2:  pen.Color:=ClBlack;
3:  pen.Color:=ClGray;
4:  pen.Color:=ClYellow;
5:  pen.Color:=ClPurple;
6:  pen.Color:=ClSilver;
7:  pen.Color:=ClGreen;
8:  pen.Color:=ClAqua;
9:  pen.Color:=ClBlue;
10:  pen.Color:=ClFuchsia;
11:  pen.Color:=ClTeal;
12:  pen.Color:=ClNavy;
13:  pen.Color:=ClMaroon;
14:  pen.Color:=ClOlive;
```

mmxy - масштаб, в якому виводиться зображення многокутника вобласть *Image1;*

n - *кількість* вершин многокутника*;*

xm,ym - масиви з координатами вершин многокутника*;*

xc,yc – координати центру прямокутника, описаного навколо многокутника, зображення якого виводиться в область *Image1;*

xcd,ycd – координати центру області *Image1;*

```
procedure Graph_im1(col:integer;mmxy:real; n:word; xm,ym:array of int64;
xc,yc:integer; xcd,ycd:int64);
var Xr,Yr:mas1; i:Integer ;
Begin
for i:=0 to n-1
begin
xr[i]:=round((xm[i]-xcd)*mmxy+xc);
yr[i]:=round((-ym[i]+ycd)*mmxy+yc);
end;
with form1.image1.canvas do  begin
Pen.width:=2;
Pen.Mode:=pmCopy;
case col of
1:  pen.Color:=ClRed;
2:  pen.Color:=ClBlack;
3:  pen.Color:=ClGray;
4:  pen.Color:=ClYellow;
5:  pen.Color:=ClPurple;
6:  pen.Color:=ClSilver;
7:  pen.Color:=ClGreen;
8:  pen.Color:=ClAqua;
9:  pen.Color:=ClBlue;
10:  pen.Color:=ClFuchsia;
11:  pen.Color:=ClTeal;
12:  pen.Color:=ClNavy;
13:  pen.Color:=ClMaroon;
14:  pen.Color:=ClOlive
else pen.Color:=TColor(RGB(255,128,64));
end;
for i:=0 to n-1 do
if i=0 then moveto(xr[i],yr[i]) else lineto(xr[i],yr[i]); {Ðèñóåì êîíòóðû äåòàëè}
end;
End;
```

Процедура *Skv* визначає площу *s* многокутника, який визначається наступними параметрами:

n - *кількість* вершин многокутника*;*

x,y - масиви з координатами вершин многокутника.

```
Procedure Skv(n:integer; Var x,y:array of Int64; Var s:Int64);
Var i:integer;
Begin
s:=0;
For i:=0 to n-2 do
s:=s+x[i]*y[i+1]-x[i+1]*y[i];
s:=Round(abs(s)/2)
End;
```

Процедура *PrKoor* забезпечує перерахування координат вершин многокутника при повороті його на кут *alf*, який визначається наступними параметрами[27-29]:

n - *кількість* вершин многокутника*;*

x,y - масиви з координатами вершин многокутника;

xr,yr - масиви з координатами вершин многокутника при повороті його на кут *alf*.

```
Procedure PrKoor(n:integer; var x,y,xr,yr:array of Int64;
alf:real;xr1,yr1:Int64);
Var i:integer;
begin
for i:=0 to n-1 do
begin
xr[i]:=round(x[i]*cos(alf)-y[i]*sin(alf)+xr1);
yr[i]:=round(x[i]*sin(alf)+y[i]*cos(alf)+yr1);
end;
end;
```

Процедура *polig* генерує мнокутник з координатами вершин *x,y* вписаного в коло радіусу *r* та повернутого на кут *alf*.

Параметри процедури:

n - *кількість* вершин многокутника*;*

x,y - масиви з координатами вершин многокутника;

*Xc,Yc –ко*ординати центру многокутника.

```
procedure polig(var n:integer;var x,y:array of Int64;r,alf,Xc,Yc:int64);
var
i:integer;
begin
for i:=0 to n-1 do
begin
x[i]:=round(r*cos(pi*(alf/180+2*i/n))+xc);
y[i]:=round(r*sin(pi*(alf/180+2*i/n))+yc);
end;
n:=n+1;
x[n-1]:=x[0];
y[n-1]:=y[0];
end;
```

Процедура *krest* генерує мнокутник з координатами вершин *x,y*, відображає хрест повернутий на кут *alf*.

Параметри процедури:

n - кількість вершин многокутника*;*

x,y - масиви з координатами вершин многокутника;

*Xc,Yc –ко*ординати центру многокутника;

a,b,d- відповідно довжина сторін a,b та ширина сторін.

```
procedure krest(var n:integer;var x,y:array of Int64;a,b,d,alf:integer;Xc,Yc:Int64);
var
i:integer;
xr,yr:mas1;
begin
n:=13;
xr[0]:=-round(a/2);
yr[0]:=-round(d/2);
xr[1]:=-round(a/2);
yr[1]:=round(d/2);;
xr[2]:=-round(d/2);
yr[2]:=round(d/2);
xr[3]:=-round(d/2);
yr[3]:=round(b/2);
xr[4]:=round(d/2);
```

```
yr[4]:=round(b/2);
xr[5]:=round(d/2);
yr[5]:=round(d/2);
xr[6]:=round(a/2);
yr[6]:=round(d/2);
xr[7]:=round(a/2);
yr[7]:=-round(d/2);
xr[8]:=round(d/2);
yr[8]:=-round(d/2);
xr[9]:=round(d/2);
yr[9]:=-round(b/2);
xr[10]:=-round(d/2);
yr[10]:=-round(b/2);
xr[11]:=-round(d/2);
yr[11]:=-round(d/2);
xr[12]:=-round(a/2);
yr[12]:=-round(d/2);
PrKoor(n,xr,yr,x,y,alf/180*pi,xc,yc);
end;
```

Процедура *star* генерує мнокутник з координатами вершин *x,y,* відображає зірку, повернуту на кут *alf*[30-31].

Параметри процедури:

n - кількість вершин зірки*;*

x,y - масиви з координатами вершин многокутника;

*Xc,Yc –ко*ординати центру многокутника;

br,mr - відповідно зовнішній та внутрішній радіуси зірки.

```
procedure star(var n:integer;var x,y:array of Int64;br,mr,alf:integer;Xc,Yc:Int64);
var
i:integer;
xr,yr:mas1;
begin
for i:=0 to n-1 do
begin
xr[2*i]:=round(br*cos(2*pi*i/n));
yr[2*i]:=round(br*sin(2*pi*i/n));
xr[2*i+1]:=round(mr*cos(2*pi*i/n+pi/n));
yr[2*i+1]:=round(mr*sin(2*pi*i/n+pi/n));
```

```
end;
xr[2*n]:=xr[0];
yr[2*n]:=yr[0];
n:=2*n+1;
PrKoor(n,xr,yr,x,y,alf/180*pi,xc,yc);
end;
```

Процедура *pryam* генерує мнокутник з координатами вершин *x,y*, відображає прямокутник, повернуту на кут *alf*.

Параметри процедури:

n - кількість вершин мнокутника;

x,y - масиви з координатами вершин многокутника;

*Xc,Yc –ко*ординати центру многокутника;

ar, br - відповідно дожина та ширина прямокутника.

```
procedure pryam(var n:integer;var x,y:array of Int64;ar,br,alf:integer;
Xc,Yc:Int64);
var
xr,yr:mas1;
begin
xr[0]:=round(ar/2);
yr[0]:=round(br/2);
xr[1]:=-round(ar/2);
yr[1]:=round(br/2);
xr[2]:=-round(ar/2);
yr[2]:=-round(br/2);
xr[3]:=round(ar/2);
yr[3]:=-round(br/2);
n:=5;
xr[n-1]:=xr[0];
yr[n-1]:=yr[0];
PrKoor(n,xr,yr,x,y,alf/180*pi,xc,yc);
end;
```

Процедура *romb* генерує мнокутник з координатами вершин *x,y*, відображає ромб, що повернутий на кут *alf*.

Параметри процедури:

n - кількість вершин мнокутника;

x,y - масиви з координатами вершин многокутника;

*Xc,Yc –ко*ординати центру многокутника;

ar, br - відповідно дожини діагоналей ромбу.

```
procedure romb(var n:integer; var x,y:array of Int64;ar,br,alf:integer;Xc,Yc:Int64);
var
xr,yr:mas1;
begin
xr[0]:=round(br/2);
yr[0]:=0;
xr[1]:=0;
yr[1]:=round(ar/2);
xr[2]:=-round(br/2);
yr[2]:=0;
xr[3]:=0;
yr[3]:=-round(ar/2);
n:=5;
xr[n-1]:=xr[0];
yr[n-1]:=yr[0];
PrKoor(n,xr,yr,x,y,alf/180*pi,xc,yc);
end;
```

Процедура *oval* генерує мнокутник з координатами вершин *x,y,* відображає еліпс, що повернутий на кут *alf*[32].

Параметри процедури:

n - кількість вершин мнокутника*;*

x,y - масиви з координатами вершин многокутника;

*Xc,Yc –ко*ординати центру многокутника;

ar, br – відповідно піввісі *a, r* еліпсу.

```
procedure oval(var n:integer;var x,y:array of Int64;ar,br,alf:integer;Xc,Yc:Int64);
var
i:integer;
alfa:real;
xr,yr:mas1;
begin
for i:=0 to 19 do
begin
```

```
alfa:=2*pi*i/20;
xr[i]:=round(ar*cos(alfa+pi*alf/180));
yr[i]:=round(br*sin(alfa+pi*alf/180));
end;
n:=21;
xr[n-1]:=xr[0];
yr[n-1]:=yr[0];
PrKoor(n,xr,yr,x,y,alf/180*pi,xc,yc);
end;
```

Процедура *Krug* генерує мнокутник з координатами вершин *x,y,* відображає коло.

Параметри процедури:

n - кількість вершин мнокутника*;*

x,y - масиви з координатами вершин многокутника;

*Xc,Yc –ко*ординати центру многокутника;

ar – радіус кола.

```
procedure Krug(var n:integer;var x,y:array of Int64;ar:integer;Xc,Yc:Int64);
var
i:integer;
alfa:real;
xr,yr:mas1;
begin
for i:=0 to 19 do
begin
alfa:=2*pi*i/20;
x[i]:=round(ar*cos(alfa)+Xc);
y[i]:=round(ar*sin(alfa)+Yc);
end;
n:=21;
x[n-1]:=x[0];
y[n-1]:=y[0];
end;
```

Процедура *oval* генерує мнокутник з координатами вершин *x,y,* відображає краплю, що повернута на кут *alf.*

Параметри процедури:

n - кількість вершин мнокутника*;*

x,y - масиви з координатами вершин многокутника;

*Xc,Yc –ко*ординати центру многокутника;

r0, r – відповідно радіуси овалу та основи краплі.

```
procedure kaplya(var n:integer;var x,y:array of Int64;
                 r,r0,alf:Integer;Xc,Yc:Int64);
                  var
                 i:integer;
                 xr,yr:mas2;
                 eps,ug,v,t,du,c:real;
                  begin
                 eps:=0.5;
                   c:=R*pi;
                   ug:=pi;
                   t:=1-eps/r;
              v:=Pi/2-ArcTan(t/(sqrt(1-t*t)));
              n:=round(ug/2/v);
              if n<=1 then n:=4;
               du:=ug/(n-1);
              for i:=0 to n-1 do
                  begin
              xr[i]:=round(r*cos(i*du));
              yr[i]:=round(r*sin(i*du));
                   end;
              n:=n+1;
              xr[n-1]:=0;
             yr[n-1]:=-round(r0);
              n:=n+1;
             xr[n-1]:=xr[0];
             yr[n-1]:=yr[0];
          PrKoor(n,xr,yr,x,y,alf/180*pi,xc,yc);
              end;
```

3.4. Інструкції по роботі з програмним продуктом

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

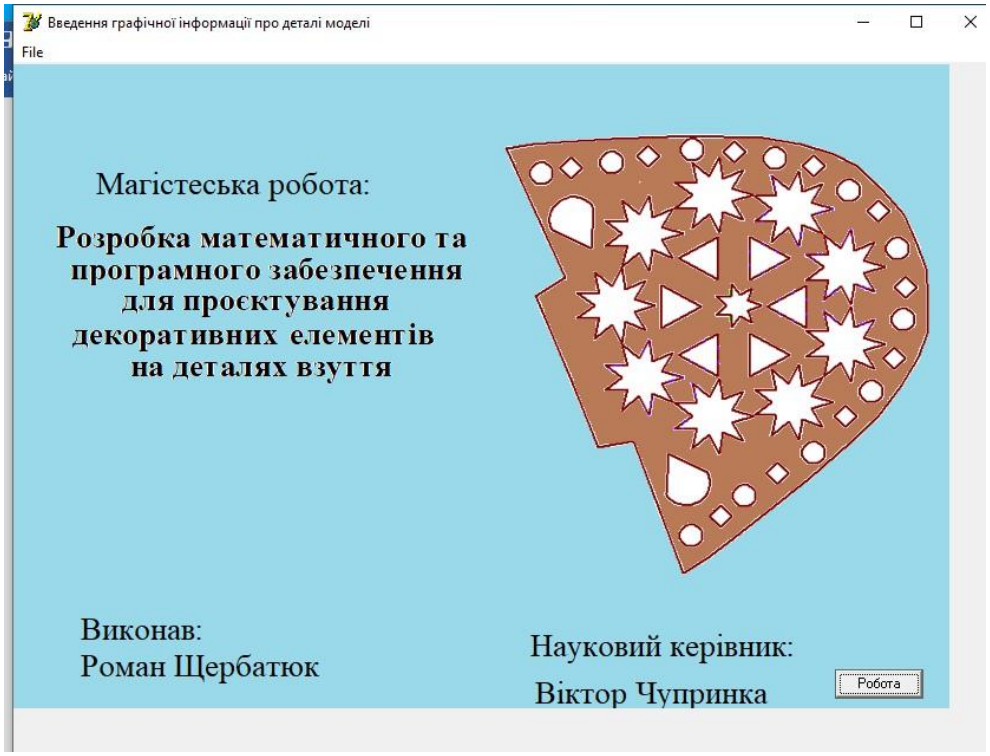


Рис. 3.1

Після натиску на кнопку «Робота» головне вікно програми наступний вигляд (рис. 3.2).

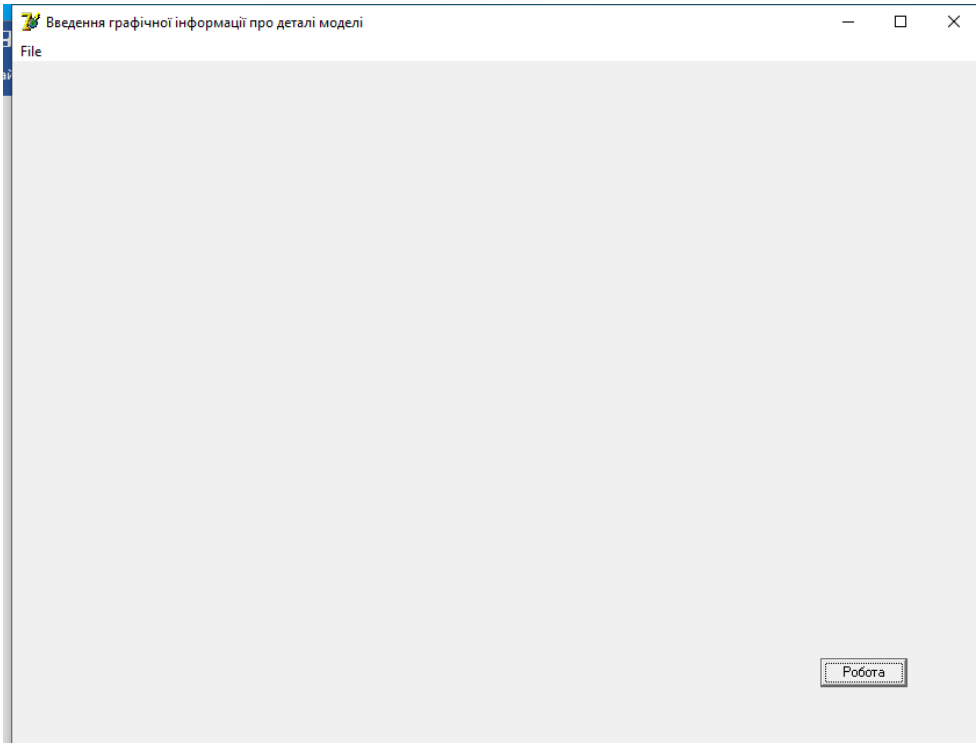


Рис. 3.2

Тепер вводимо інформацію про зовнішні деталей моделі. Після цього головне вікно програми настуний вигляд (рис. 3.3).

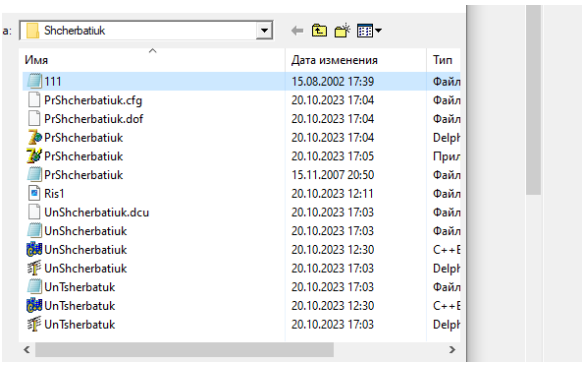


Рис. 3.3

Після вводимо інформацію про зовнішні деталей моделі головне вікно програми настуний вигляд (рис. 3.4).

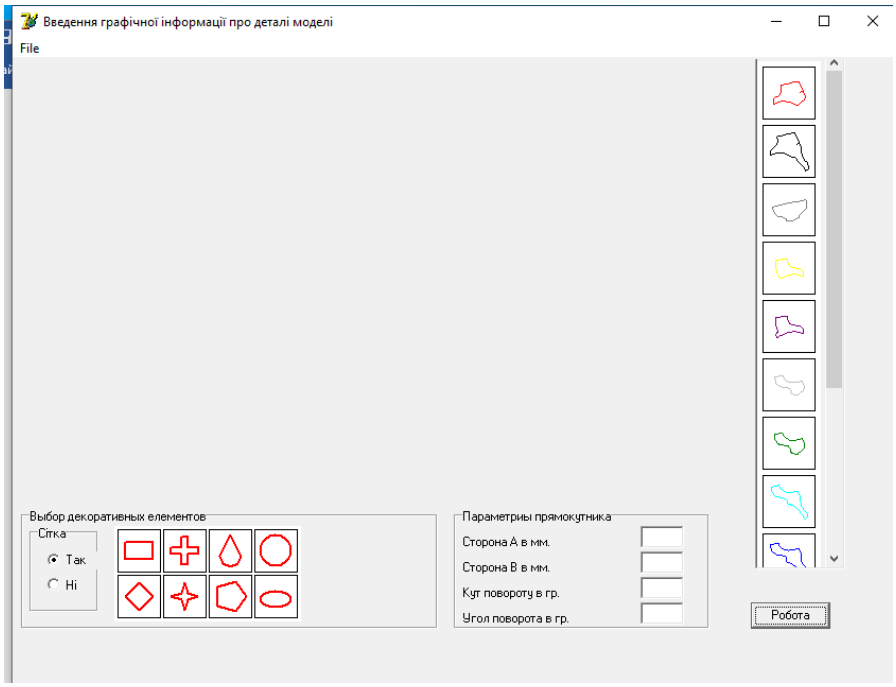


Рис. 3.4

Після вибору деталі із меню, що знаходиться праворуч на формі, головне вікно програми настуний вигляд (рис. 3.5).

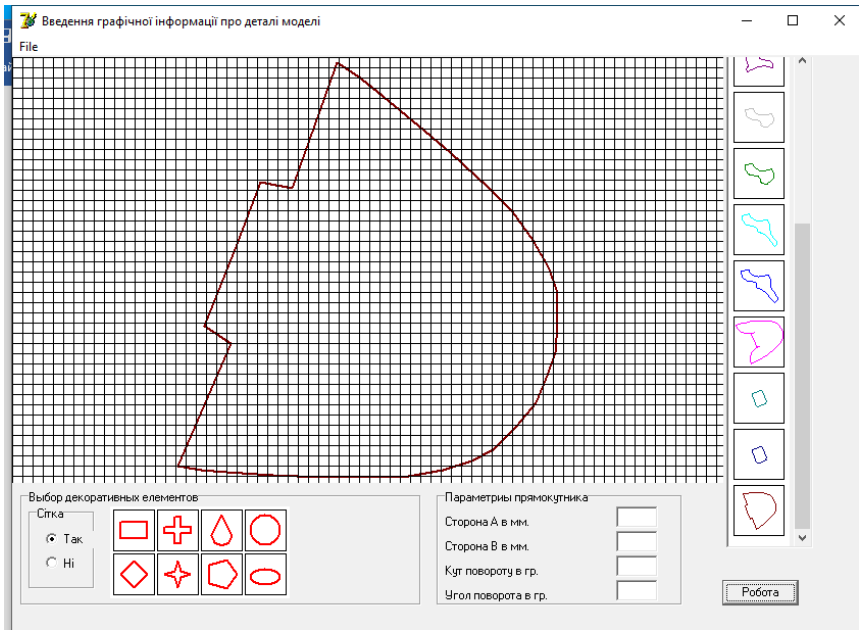


Рис. 3.5

Після вибору декоративного елементу із меню, що знаходиться знизу на формі та введення параметрів вибраного декоративного елементу, головне вікно програми настуний вигляд (рис. 3.6).

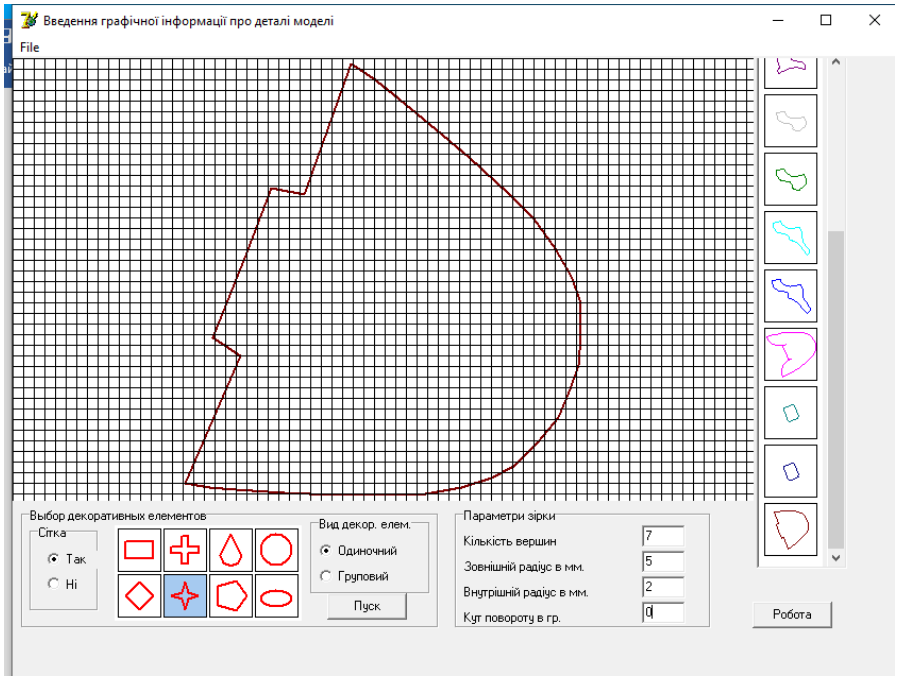


Рис. 3.6

Після введення інформації про вибраний одинарний декоративний елемент та розміщення його на деталі головне вікно програми настуний вигляд (рис. 3.7).

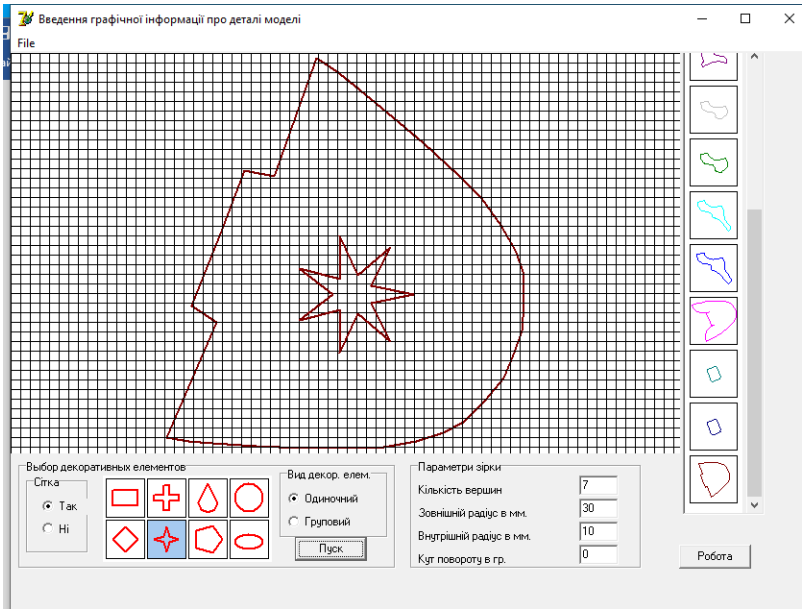


Рис. 3.7

Програма має молжливість розміщати групові декоративні елементи на деталях(рис. 3.8)

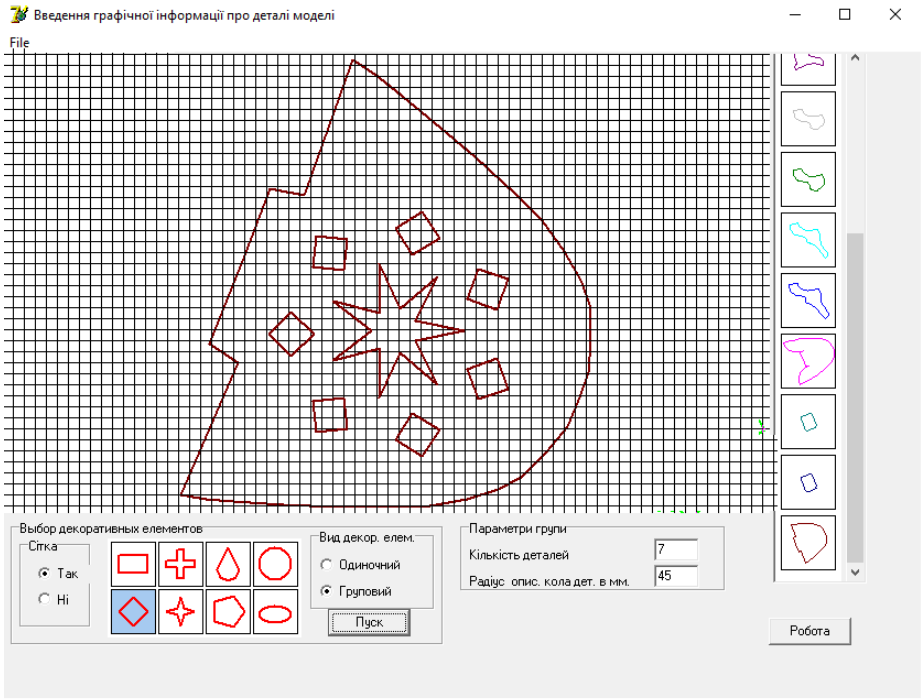


Рис. 3.8

Висновки до третього розділу

1. Використавши розроблені алноритми генерування одинарних та групових декоративних елементів на деталях взуття розроблений відповідний програмний продукт для генерування декоративних елементів на деталях взуття враховуючи параметрами цих елементів
2. Програмний продукт має дружній інтерфейс та не потребує спеціальних знань з комп'ютерних наук для роботи з ним..

ВИСНОВКИ

Проведено дослідження та розроблені такі алгоритми для автоматизованого проектування системних декоративних елементів на деталях виробів шкіргалантереї:

1. Генерування на деталі декоративних елементів із одного примітиву, а саме: трикутника, прямокутника, трапеції, многокутника, пів еліпса, еліпса хреста, краплі та інших;
2. Генерування генерування на деталі комбінованих декоративних елементів
3. Збереження інформації про спроєктованих декоративних елементів деталей у файлі.

Запропоноване математичне забезпечення для генерування одинарних та групових декоративних елементів на деталях взуття реалізоване в програмний продукт для генерування одинарних та групових декоративних елементів на деталях взуття

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

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Gerber Hiera mit Partner//Schuh-Techn. Int. Schuh-Techn. + ABC. 1997.- 91, № 1 -2.-P. 6
2. Lectra-Marktfiihrer in Brasilien//Schuh-Techn. Int.1995.-89, № 1 - 2. P.24-25.
3. Каменець С.Є., Васютинська В.В. Використання сучасних методів візуального дизайну та просторового моделювання для створення шкіргалантерейних виробів// Вісник Хмельницького національного університету. – 2020. - № 4. с. 199-205.
4. Столярова В. В., Каменець С. Є., Борщевська Н. М. Просторове моделювання та проектування аксесуарів і фурнітури виробів індустрії моди / Технології та дизайн № 3(36) 2020р. [Електронний ресурс] - Режим доступу:https://drive.google.com/file/d/1DEkBLCjWLsRG23OkK0hgnKNDO-uZTZ6i/ view?usp=sharing.
5. Омельченко П.В. Алгоритм автоматизованої підготовки інформації про деталі шкіргалантерейних виробів / П.В. Омельченко, В.П. Коновал, В.І. Чупринка // Вісник КНУТД. – 2004. – № 1. – С. 85-89.
6. Кременок В.В. Автоматизоване проектування деталей взуття / В.В. Кременок, А.В. Пінчук, В.І. Чупринка. Тези доповідей VII Всеукраїнської наукової конференції молодих вчених та студентів. – Київ, 2008. – С. 123
7. Чупринка В.І., Чебанюк О.В. Алгоритм автоматичної підготовки вихідної інформації для побудови раціональних схем розкрою / В.І. Чупринка, О.В. Чебанюк // Вісник КНУТД. – 2006. – № 5. – С. 18-22.
8. Чупринка В.І. Підготовка інформації для автоматичного розкрою / В.І. Чупринка, Волошин, Піпа Т.А. // Вісник ДАЛПУ. – 2000. - №1. – С. 91-83.

9. Є. П. Зайцев: Вища математика. Лінійна та векторна алгебра, аналітична геометрія, вступ до математичного аналізу Алерта 574 с.

10. Чупринка В.І. Підготовка інформації для побудови раціональних схем розкрою рулонних матеріалів на деталі взуття та графічна візуалізація цих схем / В.І. Чупринка, О.О. Шишкіна, Л.Т. Свістунова // Вісник Технологічного університету Поділля. – 2003. - №5. - Ч1. – С. 38-41
11. Чупринка В.І. Система підготовки інформації для автоматичного розкрою спроектованих схем розкрою./ В.І. Чупринка, В.Ю. Щербань, І.І. Тонн // Вісник КНУТД. – 2003. - №2 . – С. 171-173.
12. Омельченко П.В. Автоматизована підготовка інформації про контури деталей шкіргалантерейних виробів / П.В. Омельченко, В.П. Коновал, В.І. Чупринка // Вісник КНУТД. – 2004. - № 4. – С. 138-142
13. Чупринка В.І. Алгоритм підготовки інформації для побудови розкрійних схем рулонних матеріалів на деталі взуття та шкіргалантерейних виробів / В.І. Чупринка, О.З Колиско // Вісник КНУТД. – 2005. - №3 – С. 19-24.
14. Чупринка В.І. Програмні методи підготовки інформації для автоматизованого розкрою матеріалу прямокутної форми / В.І. Чупринка, О.О. Хоменко, М.М. Шкоденко // Проблеми програмування. – 2008. -№2-3. - С. 665-668
15. Аналітична геометрія Підручник для вищих навчальних закладів Б. В. Гринев, І. К . Кириченко Гімназія 344 с.
16. Рудавський Ю. К. Лінійна алгебра та аналітична геометрія [Текст]: навч. посібн. / Ю. К. Рудавський, П. П. Костробій, Х. П. Луник, Д. В. Уханська, ДУ «Львівська політехніка», 1999. — 262 с.
17. Чупринка Н. В. Програмне забезпечення для генерування декоративних елементів на деталях жіночих сумок / Н. В. Чупринка // Тези доповідей XIV Всеукраїнської наукової конференції молодих вчених та студентів, 23-24

квітня 2015 р. „Наукові розробки молоді на сучасному етапі”. – Т1 - К.:КНУТД, 2015. – С. 72.

18. Чупринка В.І. Алгоритм підготовки інформації для побудови розкрійних схем рулонних матеріалів на деталі взуття та шкіргалантерейних виробів / В.І. Чупринка, О.З Колиско // Вісник КНУТД. – 2005. - №3 – С. 19-24.
19. Чупринка В.І. Програмні методи підготовки інформації для автоматизованого розкрою матеріалу прямокутної форми / В.І. Чупринка, О.О. Хоменко, М.М. Шкоденко // Проблеми програмування. – 2008. -№2-3. - С. 665-668
20. Караванова Т. П. Основи алгоритмізації та програмування. Форум. – К.: 2002. – 286 с.
21. .Ковалюк Т. В. Алгоритмізація та програмування: підручник з грифом МОН України / Т.В. Ковалюк. – Львів : Магнолія-2006, 2013. – 400 с.
22. Шаховська, Н. Б. Алгоритми і структури даних [Текст]: посібник / Н.Б. Шаховська, Р.О. Голощук; - Львів: Магнолія, 2010 – 215с.
23. Веселовська Г.В. та Ходакова, В.Є., 2015. Комп’ютерна графіка. Київ: Кондор.
24. . Михайленко В. Є. Інженерна та комп’ютерна графіка. К. : Вища школа, 2002. 332 с.
25. Горобець С.М., 2006. Основи комп’ютерної графіки. Київ: Центр навчальної літератури. Manual of Shoemaking. /Under edition of R.G. Miller.Produced By the Trading Department Clarks, 1989. - 337 p.
26. Березовський В.С., Потієнко, В.О. та Завадський, І.О., 2009. Основи комп’ютерної графіки. Київ: BHV.
27. Веселовська Г.В. та Ходакова, В.Є., 2015. Комп’ютерна графіка. Київ: Кондор.
28. . Михайленко В. Є. Інженерна та комп’ютерна графіка. К. : Вища школа, 2002. 332 с.
29. Горобець С.М., 2006. Основи комп’ютерної графіки. Київ: Центр навчальної літератури. Manual of Shoemaking. /Under edition of R.G. Miller.Produced By the Trading Department Clarks, 1989. - 337 p.

30. Березовський В.С., Потієнко, В.О. та Завадський, І.О., 2009. Основи комп’ютерної графіки. Київ: BHV.

31. Ванін В.В., Перевертун, В.В. та Надкернична, Т.М., 2006. Комп’ютерна інженерна графіка в середовищі AutoCAD. Київ: Каравела. 2023, 384 с.

32. Романюк О.Н., 2001. Комп’ютерна графіка. Вінниця: Вінницький державний технічний університет.

Додаток А

Публікації по темі випускної кваліфікаційної магістерської роботи

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Київський національний університет технологій та дизайну

Кафедра комп'ютерних наук

ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ В НАУЦІ, ВИРОБНИЦТВІ ТА ПІДПРИЄМНИЦТВІ

Збірник наукових праць молодих вчених, аспірантів, магістрів кафедри комп'ютерних наук

Київ – 2023

$$A_1 \cdot Y^2 + B_1 \cdot Y + C_1 = 0 \quad (9)$$

Якщо дискримінант D рівняння (9) більший або дорівнює нулю, то коло та еліпс перетинаються.

$$D = B_1{}^2 - 4A_1 \cdot C_1 \geq 0 \quad (10)$$

Тоді

$$Y_{1,2} = \frac{-B_1 \pm \sqrt{D}}{2A_0}, \quad (11)$$

$$X_{1,,2} = -\frac{B_0}{A_0} Y_{1,2} - \frac{C_0}{A_0}. \quad (12)$$

Розроблені параметричні моделі деталей чоловічих на не типову фігуру були реалізовані в програмне забезпечення для автоматизованого проектування цих виробів. Програмне забезпечення має дружній інтерфейс та не потребує спеціальних знань з комп'ютерних наук при роботі з ним та може бути використаним в ательє індивідуального пошиву чоловічих штанів. Розроблений програмний продукт дозволяє запам'ятати інформацію про деталі спроектованих штанів, вивести креслення цих деталей в натуральну величину або в масштабі.

Висновки

Запропоноване математичне та програмне забезпечення для автоматизованого проектування чоловічих штанів має практичну значимість, так як воно направлене на підвищення конкурентоспроможності вітчизняного малого виробництва.

ЩЕРБАТЮК Р.В. А.В, ЧУПРИНКА В.І.

РОЗРОБКА МАТЕМАТИЧНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ПРОЄКТУВАННЯ ДЕКОРАТИВНИХ ЕЛЕМЕНТІВ НА ДЕТАЛЯХ ВЗУТТЯ

SHCHERBATYUK R.V., CHUPRYNKA V.I.

DEVELOPMENT OF MATHEMATICAL AND SOFTWARE FOR DESIGNING DECORATIVE ELEMENTS ON FOOTWEAR DETAILS

The article is devoted to the development of mathematical and software for designing decorative elements on shoe details. The software has a user-friendly interface and does not require additional computer science knowledge when working with it.
Key words: design, decorative elements, shoes, software

Вступ

Часта зміна моделей взуття потребує значного підвищення підготовчих робіт. Скорочення термінів цих робіт, зменшення вартості та підвищення якості технологічних рішень повинно бути досягнуто шляхом підвищення продуктивності за рахунок впровадження у виробництво

математичних методів, обчислювальної техніки та розробки програмних засобів технологічної підготовки виробництва. Це зумовлює необхідність створювати у легкій промисловості гнучкі виробничі системи, які швидко і з мінімальними затратами могли переналагоджуватись на випуск нової продукції.

На більшості виробництв існуючі в нинішній час процеси проектування взуття не забезпечують необхідної мобільності виробництва. Цикл робіт від створення нової моделі до запуску в технологічний потік залишається тривалим. Зростання об'єму проектних робіт в умовах частої зміни моделей особливо гостро ставить задачу скорочення часу та підвищення якості процесу проектування. Також ціллю автоматизації проектування є, зниження матеріальних затрат, скорочення термінів проектування та ліквідація тенденції до збільшення кількості інженерно-технічних робітників, які зайняті проектуванням, підвищення продуктивності їх праці.

Це дозволить, по-перше, підвищити якість і скоротити терміни рішення проектних задач за рахунок можливості розглядати як весь об'єкт у цілому, так і взаємозв'язку його елементів. По-друге, розробка структурно-графічних моделей технологічних об'єктів є формалізованим їхнім описом, що дозволяє здійснити перехід до математичних моделей — як основи алгоритмізації інтелектуальних процесів у технологічному проектуванні.

Основна частина

За декоративні елементи приймемо геометричні примітиви (деталі з простою конфігурацією зовнішнього контуру), проектування яких було детально розглянуто у другому розділі. Тому на цьому питанні ми зупинятись не будемо.

Так як при побудові розкрійних схем нам необхідний тільки зовнішній контур деталі, то інформацію про побудовані декоративні елементи на деталях взуття будемо зберігати в окремому файлі *.res, який пов'язаний із файлом із деталями моделі *.dgt. На деталі взуття може бути нанесено багато декоративних елементів складної форми. Тому зберігати зовнішні контури декоративних елементів не раціонально, краще зберігати параметри цих елементів, по яким однозначно можна відтворити декоративні елементи [308]. Такі параметри можна розбити на два типи:

- параметри, що відповідають за місце розташування декоративного елемента (вони стандартні для кожного декоративного елемента) та параметри декоративного елементу, які однозначно визначають зовнішній контур декоративного елементу.

До параметрів, що відповідають за місце розташування декоративного елемента можна віднести наступні:

PD - признак деталі, на який наноситься декоративний елемент;

PE - признак декоративного елемента;

Xp,Yp - координати полюса декоративного елементу відносно деталі;

θ – кут повороту декоративного елементу відносно його основного положення.

Приклади побудованих декоративних елементів на деталях взуття представлені на рис. 7.26.

Параметри декоративних елементів представлені в таблиці 1.

Таблиця 1.Параметри декоративних елементів

PE	Назва декоративного елементу	Параметри декоративного елементу
1	Прямокутник	Сторони прямокутника *a* та *b*
2	Ромб	Діагоналі ромба d_1 та d_2
3	Хрест	Довжина *a*, ширина *b*, товщина хреста *h*
4	*N* - кутник	Кількість сторін многокутника *N*, радіус описаного кола *R*
5	Коло	Радіус кола *R*
6	Еліпс	Піввісі еліпса *a* та *b*
7	Крапля	Сторона *a* та радіус півкола *R*
8	Зірка	Кількість сторін зірки *N*, радіус описаного кола *R* та радіус внутрішнього кола *r*

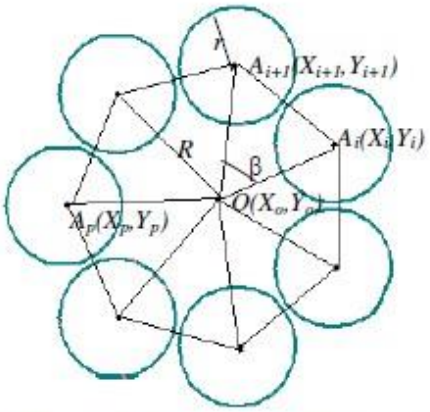


Рис. 1. Модель побудови групових декоративних

Під груповими декоративними елементами ми будемо мати на увазі однакові декоративні елементи розміщені на колі радіуса *R* таким чином, що відстань між сусідніми груповими елементами по дузі цього кола однакові. Тоді $A_iA_{i+1}=A_{i+1}A_{i+2}=...=A_pA_{p+1}=2r+\Delta$, де Δ- відстань між зовнішніми границями двох кіл, що описані навколо двох сусідніх групових декоративних елементів(рис. 1).

Задачу про побудову групових елементів можна сформулювати наступним чином (рис. 1).

Дано: точка $O(X_o,Y_o)$, радіус кола $R=OQ$, до якого дотикаються групові елементи, кількість N групових елементів у групі.

Знайти: координати полюса $A_i(X_i,Y_i), i=0,1,2..N-1$ для кожного із групових елементів, радіус кола $r=QA_p$, описаного навколо групового елемента, кут між двома сусідніми елементами $\angle A_iOA_{i+1} = \angle\beta$.

Розглянемо ΔA_iOA_{i+1} (рис.1). Він рівнобедрений, т.я. $OA_i=OA_{i+1}=R_o$. Тоді $A_iA_{i+1}=R_o/sin(\beta/2)$ та $\angle\beta = 2\pi/N$. Звідси маємо $2r+\Delta=R_o/sin(\beta)$. Але $QO=R=R_o+r$, або $R_o=R-r$. Тоді $2r+\Delta= (R-r)\ sin(\beta/2)$, або $r=(R-\Delta)/(2+sin(\beta/2))$.

Нехай базовий елемент для заданої групи елементів має координати: $\{Xв_j,Yb_j\}$, де $j=1,2,...Nb$. Тоді координати $\{Xq_i^j,Yq_i^j\}$ зовнішнього контуру i-го групового елементу будуть визначатись наступним чином:

$$\begin{cases} Xq_i^j = X_i + Xb_j\cos(\beta/2+i\cdot\beta) - Yb_j\sin(\beta/2+i\cdot\beta) \\ Xq_i^j = X_i + Xb_j\cos(\beta/2+i\cdot\beta) - Yb_j\sin(\beta/2+i\cdot\beta) \end{cases}, \quad (1)$$

де $i=0,1..N, j=1,2,...Nb$.

Всі розглянуті задачі реалізовані в програмному продукті в середовиші програмування Delphi для операційної системи Windows.

Висновки

Запропоноване математичне та програмне забезпечення для автоматизованого проектування декоративних елементів на деталях взуття має практичну значимість, так як воно направлене на підвищення конкурентоспроможності вітчизняного малого виробництва.

АРТЕМЕНКО П.Ю., ЧУПРИНКА Н. В.

РОЗРОБКА МАТЕМАТИЧНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ АВТОМАТИЗОВАНОГО ПРОЄКТУВАННЯ ДЕТАЛЕЙ ВИРОБІВ ШКІРГАЛАНТЕРЕЇ

ARTEMENKO P.Yu., CHUPRYNKA N.V.

DEVELOPMENT OF MATHEMATICAL AND SOFTWARE FOR AUTOMATED DESIGN OF DETAILS OF LEATHER GOODS

The article is devoted to the development of mathematical and software for the automated design of details of leather goods.The software has a friendly interface and does not require additional knowledge of computer science when working with it.

Key word: Parametric models, leather goods, software.

Вступ

Науково-технічний прогрес, пов'язаний з автоматизацією різноманітних етапів проектування, знаходить широке застосування в багатьох галузях промисловості, у тому числі і легкої. Тому при підготовці виробництва

Міністерство освіти і науки України

Київський національний університет
технологій та дизайну

МЕХАТРОННІ СИСТЕМИ: ІННОВАЦІЇ ТА ІНЖИНІРИНГ

ТЕЗИ ДОПОВІДЕЙ VII МІЖНАРОДНОЇ НАУКОВО-ПРАКТИЧНОЇ КОНФЕРЕНЦІЇ

23 листопада 2023

Рекомендовано Вченою радою
факультету мехатроніки та комп'ютерних технологій
Київського національного університету технологій та дизайну

КИЇВ 2023

УДК 688.359

РОЗРОБКА МАТЕМАТИЧНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ПРОЄКТУВАННЯ ДЕКОРАТИВНИХ ЕЛЕМЕНТІВ НА ДЕТАЛЯХ ВИРОБІВ ЛЕГКОЇ ПРОМИСЛОВОСТІ

В. І. Чупринка, доктор технічних наук, професор
Київський національний університет технологій та дизайну
Р.В. Щербатюк, магістрант
Київський національний університет технологій та дизайну
Я.М. Конешький, магістрант
Київський національний університет технологій та дизайну

Ключові слова: програмне забезпечення, декоративні елементи, легка промисловість.

За декоративні елементи приймемо геометричні примітиви (деталі з простою конфігурацією зовнішнього контуру). Тому на цьому питанні ми зупинятись не будемо.

Так як при побудові розкрійних схем нам необхідний тільки зовнішній контур деталі, то інформацію про побудовані декоративні елементи на деталях будемо зберігати в окремому файлі *.res, який пов'язаний із файлом із деталями моделі *.dgt. На деталі виробів легкої промисловості може бути нанесено багато декоративних елементів. Тому зберігати зовнішні контури декоративних елементів не раціонально, краще зберігати параметри цих елементів, по яким однозначно можна відтворити декоративні елементи. Такі параметри можна розбити на два типи:

- параметри, що відповідають за місце розташування декоративного елемента (вони стандартні для кожного декоративного елемента) та параметри декоративного елементу, які однозначно визначають зовнішній контур декоративного елементу (таблиця 1).

До параметрів, що відповідають за місце розташування декоративного елемента можна віднести наступні:

PD - признак деталі, на який наноситься декоративний елемент;

PE - признак декоративного елемента;

Xp, Yp - координати полюса декоративного елементу відносно деталі;

θ – кут повороту декоративного елементу відносно його основного положення.

Задачу про побудову групових елементів можна сформулювати наступним чином (рис. 1.).

Дано: точка $O(X_0, Y_0)$, радіус кола $R=OQ$, до якого дотикаються групові елементи, кількість N групових елементів у групі.

Знайти: координати полюса $A_i(X_i, Y_i), i=0,1,2..N-1$ для кожного із групових елементів, радіус кола $r=QA_p$, описаного навколо групового елемента, кут між двома сусідніми елементами $\angle A_i O A_{i+1} = \angle\beta$.

Розглянемо $\Delta A_i O A_{i+1}$ (рис.1). Він рівнобедрений, т.я. $OA_i=OA_{i-1}=R_o$ Тоді $A_iA_{i-1}=R_o/sin(\beta/2)$ та $\angle\beta=2\pi/N$. Звідси маємо $2r+\Delta=R_o/sin(\beta)$. Але $QO=R=R_o+r$, або $R_o=R-r$. Тоді $2r+\Delta=(R-r)\ sin(\beta/2)$, або $r=(R-\Delta)/(2+sin(\beta/2))$.

Параметри декоративних елементів представлені в таблиці 1.

Таблиця 1.

Параметри декоративних елементів

PE	Назва декоративного елементу	Параметри декоративного елементу
1	Прямокутник	Сторони прямокутника *a* та *b*
2	Ромб	Діагоналі ромба d_1 та d_2
3	Хрест	Довжина *a*, ширина *b*, товщина хреста *h*
4	*N* - кутник	Кількість сторін многокутника *N*, радіус описаного кола *R*
5	Коло	Радіус кола *R*
6	Еліпс	Півосі еліпса *a* та *b*
7	Крапля	Сторона *a* та радіус півкола *R*
8	Зірка	Кількість сторін зірки *N*, радіус описаного кола *R* та радіус внутрішнього кола *r*

Координати полюсів $A_i(X_i,Y_i), i=0,1,2..N-1$ групових декоративних елементів будуть визначатись наступним чином:

$$X_i=X_o+R_o cos(\beta/2+i\beta)$$
$$Y_i=Y_o+R_o sin(\beta/2+i\beta).$$

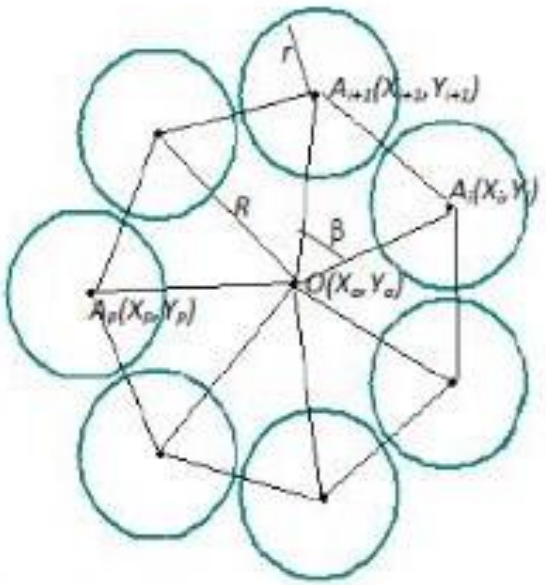


Рисунок 1 - Модель побудови групових декоративних елементів

Так як декоративні елементи в більшості випадків представляють собою технологічні отвори на деталях виробів легкої промисловості, за допомогою роботизованих комплексів.

Додаток Б

Листінг програмного продукту

```
unit UnTsherbatuk;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, Menus, StdCtrls;

type
  TForm1 = class(TForm)
    Button1: TButton;
    MainMenu1: TMainMenu;
    File1: TMenuItem;
    Open1: TMenuItem;
    N1: TMenuItem;
    Image1: TImage;
    ScrollBox1: TScrollBox;
    OpenDialog1: TOpenDialog;
    Image2: TImage;
    GroupBox1: TGroupBox;
    GroupBox2: TGroupBox;
    RadioButton3: TRadioButton;
    RadioButton4: TRadioButton;
    Image3: TImage;
    GroupBox3: TGroupBox;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    Edit4: TEdit;
    Label4: TLabel;
    N2: TMenuItem;
    Exit1: TMenuItem;
    Button2: TButton;
```

```
    GroupBox4: TGroupBox;
    RadioButton1: TRadioButton;
    RadioButton2: TRadioButton;
    Image4: TImage;
    procedure Button1Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Image2MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure RadioButton4MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure RadioButton3MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure Image3MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure Edit1KeyPress(Sender: TObject; var Key: Char);
    procedure Edit2KeyPress(Sender: TObject; var Key: Char);
    procedure Edit3KeyPress(Sender: TObject; var Key: Char);
    procedure Edit1Change(Sender: TObject);
    procedure Edit2Change(Sender: TObject);
    procedure Edit3Change(Sender: TObject);
    procedure Image1MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure Edit4Change(Sender: TObject);
    procedure Exit1Click(Sender: TObject);
    procedure Open1Click(Sender: TObject);
    procedure N1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Image1MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure RadioButton1MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure RadioButton2MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);


  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
```

```
  Form1: TForm1;

implementation

{$R *.dfm}

Const    KolD=40;


   Type
        Mas=array[1..40]of Int64;
        Mas1=array[0..400]of Int64;
        Mas2=array[0..400]of Int64;

        Rec=record
         Nd,Ndek,
         A,B,D,Alf,Nr,Xcur,Ycur:Int64;
        end;

Var     Res:array[1..1000]of Rec;
      f,fu:System.text;
      FileName,FName:string;
      xd,yd,X_dek,Y_dek,X_n,Y_n:array[1..KolD]of Mas1;
      KolDet,pv,NR,NS,NRes,x_k,y_k:integer;
      Delta_X,Delta_Y,S_det:array[1..Kold]of Int64;
      NameDet:array[1..KolD] of string[25];
      KolPointDet:array[1..Kold]of word;
      Model:string[40];
      Dxy,mxy,Fi:real;
      Pr_Ed1,Pr_Ed2,Pr_Ed3,Pr_Ed4,Pr_btn2:boolean;
      A,B,D,Alf,R0,Ra:Int64;
      X_r,Y_r:mas1;
      n_r,Kd:integer;
     XcurR,YcurR:array[1..50]of Int64;

   Procedure MaxMin(var ne:integer; n:integer;
      var xy:array of Int64; var xye:Int64; np:integer);
         Var j:integer;
         begin
          ne:=1;
          xye:=xy[1];
          for j:=0 to n-1 do
```

```
            if xye*np<xy[j]*np then
              begin
               xye:=xy[j];
               ne:=j
              end
          end;

Procedure PolDet(n:integer; var x,y:array of Int64;
                         var xe,ye,xi,yi:Int64);
 var n2,k:integer;
Begin
  k:=1;
  MaxMin(n2,n,x,xe,k);
  MaxMin(n2,n,y,ye,k);
  k:=-1;
  MaxMin(n2,n,y,yi,k);
  MaxMin(n2,n,x,xi,k);
End;

Procedure Skv(n:integer; Var x,y:array of Int64; Var s:Int64);
   Var i:integer;
    Begin
     s:=0;
     For i:=0 to n-2 do
     s:=s+x[i]*y[i+1]-x[i+1]*y[i];
     s:=Round(abs(s)/2)
    End;


Procedure PrKoor(n:integer; var x,y,xr,yr:array of Int64;
              alf:real;xr1,yr1:Int64);
           Var i:integer;
            begin
             for i:=0 to n-1 do
             begin
              xr[i]:=round(x[i]*cos(alf)-y[i]*sin(alf)+xr1);
              yr[i]:=round(x[i]*sin(alf)+y[i]*cos(alf)+yr1);
             end;
            end;

procedure polig(var n:integer;var x,y:array of Int64;r,alf,Xc,Yc:int64);
 var
```

```
  i:integer;
begin
 for i:=0 to n-1 do
  begin
   x[i]:=round(r*cos(pi*(alf/180+2*i/n))+xc);
  y[i]:=round(r*sin(pi*(alf/180+2*i/n))+yc);
  end;
  n:=n+1;
  x[n-1]:=x[0];
  y[n-1]:=y[0];
end;

procedure krest(var n:integer;var x,y:array of Int64;a,b,d,alf:integer;Xc,Yc:Int64);
 var
 i:integer;
xr,yr:mas1;
begin
n:=13;
  xr[0]:=-round(a/2);
  yr[0]:=-round(d/2);
  xr[1]:=-round(a/2);
  yr[1]:=round(d/2);;
  xr[2]:=-round(d/2);
  yr[2]:=round(d/2);
  xr[3]:=-round(d/2);
  yr[3]:=round(b/2);
  xr[4]:=round(d/2);
  yr[4]:=round(b/2);
  xr[5]:=round(d/2);
  yr[5]:=round(d/2);
  xr[6]:=round(a/2);
  yr[6]:=round(d/2);
  xr[7]:=round(a/2);
  yr[7]:=-round(d/2);
  xr[8]:=round(d/2);
  yr[8]:=-round(d/2);
  xr[9]:=round(d/2);
  yr[9]:=-round(b/2);
  xr[10]:=-round(d/2);
  yr[10]:=-round(b/2);
  xr[11]:=-round(d/2);
  yr[11]:=-round(d/2);
```

```
  xr[12]:=-round(a/2);
  yr[12]:=-round(d/2);
PrKoor(n,xr,yr,x,y,alf/180*pi,xc,yc);
end;

procedure star(var n:integer;var x,y:array of Int64;br,mr,alf:integer;Xc,Yc:Int64);
 var
  i:integer;
xr,yr:mas1;
begin
 for i:=0 to n-1 do
  begin
   xr[2*i]:=round(br*cos(2*pi*i/n));
  yr[2*i]:=round(br*sin(2*pi*i/n));
  xr[2*i+1]:=round(mr*cos(2*pi*i/n+pi/n));
  yr[2*i+1]:=round(mr*sin(2*pi*i/n+pi/n));
  end;
  xr[2*n]:=xr[0];
  yr[2*n]:=yr[0];
  n:=2*n+1;
  PrKoor(n,xr,yr,x,y,alf/180*pi,xc,yc);
end;

procedure pryam(var n:integer;var x,y:array of Int64;ar,br,alf:integer;
            Xc,Yc:Int64);
 var
xr,yr:mas1;
begin
  xr[0]:=round(ar/2);
  yr[0]:=round(br/2);
  xr[1]:=-round(ar/2);
  yr[1]:=round(br/2);
  xr[2]:=-round(ar/2);
  yr[2]:=-round(br/2);
  xr[3]:=round(ar/2);
  yr[3]:=-round(br/2);
  n:=5;
  xr[n-1]:=xr[0];
  yr[n-1]:=yr[0];
PrKoor(n,xr,yr,x,y,alf/180*pi,xc,yc);
end;
```

```
procedure romb(var n:integer; var x,y:array of Int64;ar,br,alf:integer;Xc,Yc:Int64);
 var
  xr,yr:mas1;
begin
  xr[0]:=round(br/2);
  yr[0]:=0;
  xr[1]:=0;
  yr[1]:=round(ar/2);
  xr[2]:=-round(br/2);
  yr[2]:=0;
  xr[3]:=0;
  yr[3]:=-round(ar/2);
  n:=5;
  xr[n-1]:=xr[0];
  yr[n-1]:=yr[0];
PrKoor(n,xr,yr,x,y,alf/180*pi,xc,yc);
end;

procedure oval(var n:integer;var x,y:array of Int64;ar,br,alf:integer;Xc,Yc:Int64);
 var
  i:integer;
alfa:real;
xr,yr:mas1;
begin
  for i:=0 to 19 do
   begin
   alfa:=2*pi*i/20;
    xr[i]:=round(ar*cos(alfa+pi*alf/180));
   yr[i]:=round(br*sin(alfa+pi*alf/180));
   end;
  n:=21;
  xr[n-1]:=xr[0];
  yr[n-1]:=yr[0];
PrKoor(n,xr,yr,x,y,alf/180*pi,xc,yc);
end;

procedure Krug(var n:integer;var x,y:array of Int64;ar:integer;Xc,Yc:Int64);
 var
  i:integer;
alfa:real;
xr,yr:mas1;
begin
```

```
  for i:=0 to 19 do
   begin
   alfa:=2*pi*i/20;
    x[i]:=round(ar*cos(alfa)+Xc);
   y[i]:=round(ar*sin(alfa)+Yc);
   end;
  n:=21;
  x[n-1]:=x[0];
  y[n-1]:=y[0];
end;

procedure kaplya(var n:integer;var x,y:array of Int64;
                 r,r0,alf:Integer;Xc,Yc:Int64);
 var
  i:integer;
  xr,yr:mas2;
  eps,ug,v,t,du,c:real;
 begin
   eps:=0.5;
        c:=R*pi;
          ug:=pi;
          t:=1-eps/r;
          v:=Pi/2-ArcTan(t/(sqrt(1-t*t)));
          n:=round(ug/2/v);
          if n<=1 then n:=4;
          du:=ug/(n-1);
        for i:=0 to n-1 do
         begin
          xr[i]:=round(r*cos(i*du));
          yr[i]:=round(r*sin(i*du));
         end;
  n:=n+1;
  xr[n-1]:=0;
  yr[n-1]:=-round(r0);
  n:=n+1;
  xr[n-1]:=xr[0];
  yr[n-1]:=yr[0];
PrKoor(n,xr,yr,x,y,alf/180*pi,xc,yc);
end;

procedure m_circle(var n:integer;var x,y:array of Int64; r:real);
 var
```

```
  i:integer;
eps,ug,v,t,du,c:real;
begin
  eps:=0.5;
        c:=2*R*pi;
          ug:=2*pi;
          t:=1-eps/r;
          v:=Pi/2-ArcTan(t/(sqrt(1-t*t)));
          n:=round(ug/2/v);
          if n<=1 then n:=4;
          du:=ug/(n-1);
        for i:=0 to n-1 do
         begin
          x[i]:=round(r*cos((i-1)*du));
          y[i]:=round(r*sin((i-1)*du));
         end;
end;


 Procedure Scala(dln,shn:word; Var mxy:real);
    Var xxi,yyi,xi,yi,xxe,yye,xe,ye,dl,ds:Int64;
    mx:real;
      j:integer;
     Begin
      PolDet(KolPointDet[1],xd[1],yd[1],xxe,yye,xxi,yyi);
     For j:=2 to KolDet do
       begin
        PolDet(KolPointDet[j],xd[j],yd[j],xe,ye,xi,yi);
           if xxi>xi then xxi:=xi;
           if yyi>yi then yyi:=yi;
           if xxe<xe then xxe:=xe;
           if yye<ye then yye:=ye;
        end;
      dl:=xxe-xxi;
      ds:=yye-yyi;
      mx:=(dln-4)/dl;
      mxy:=(shn-4)/ds;
      if mxy>mx then mxy:=mx;
    End;

 Procedure SCALE_Im1(DlRe,ShRe:int64;ShIm,DlIm:word; var mxy:real);
 Var mx:real;
```

```
 Begin
  mx:=(DlIm-10)/DlRe;
  mxy:=(ShIm-10)/ShRe;
  If mx<mxy then mxy:=mx;
 End;


procedure Graph_im1(col:integer;mmxy:real; n:word; xm,ym:array of int64;
xc,yc:integer; xcd,ycd:int64);
var Xr,Yr:mas1; i:Integer ;
Begin
 for i:=0 to n-1 do {Перераховуємо реальні координати в екранні} begin
       xr[i]:=round((xm[i]-xcd)*mmxy+xc);
       yr[i]:=round((-ym[i]+ycd)*mmxy+yc);
      end;
      with form1.image1.canvas do  begin
       Pen.width:=2;
       Pen.Mode:=pmCopy;
      case col of
       1:  pen.Color:=ClRed; {Встановлюємо колір пера}
       2:  pen.Color:=ClBlack;
       3:  pen.Color:=ClGray;
       4:  pen.Color:=ClYellow;
       5:  pen.Color:=ClPurple;
       6:  pen.Color:=ClSilver;
       7:  pen.Color:=ClGreen;
       8:  pen.Color:=ClAqua;
       9:  pen.Color:=ClBlue;
       10:  pen.Color:=ClFuchsia;
       11:  pen.Color:=ClTeal;
       12:  pen.Color:=ClNavy;
       13:  pen.Color:=ClMaroon;
       14:  pen.Color:=ClOlive
      else pen.Color:=TColor(RGB(255,128,64));
     end;
for i:=0 to n-1 do
    if i=0 then moveto(xr[i],yr[i]) else lineto(xr[i],yr[i]); {Рисуемо контури деталі}
end;
End;
```

```
Procedure GrdM(n:integer; Var xr,yr:array of Int64;
  dxc,dyc,color:integer; dx,dy:Int64;pr:integer);
   Var i:integer;
     xp,yp:array[0..200] of integer;
     Begin
       for i:=0 to n-1 do
        begin
         xp[i]:=round((xr[i]*pr+dx)*mxy)+dxc;
        yp[i]:=round((dy-yr[i]*pr)*mxy)+dyc;
        end;
      With Form1.Image1,Canvas Do
       begin
          If pr=1 then Pen.Color:=ClFuchsia{ClGreen}
         else Pen.Color:=ClLime;
         Pen.Mode:=pmXOR;
          MoveTo(xp[0],yp[0]);
          for i:=1 to n-1 do
          LineTo(xp[i],yp[i]);
//          ellipse(round(dx*mxy)-5,round(dy*mxy)-
5,round(dx*mxy)+5,round(dy*mxy)+5);
        end;
      End;



Procedure  K1_1;
 Begin
  With Form1.Image3.Canvas do
    begin
     Brush.Color:=ClWhite;
     Brush.Style:=bsSolid;
     FloodFill(30,10,clblack,fsBorder);
     Brush.Color:=ClWhite;
     Brush.Style:=bsSolid;
     FloodFill(30,10,clblack,fsBorder);
     pen.Color:=ClRed;
     pen.Width:=2;
     Moveto(10,15);
     LineTo(36,15);
     LineTo(36,31);
     LineTo(10,31);
```

```
    LineTo(10,15);
   end
  End;

Procedure  K1_2;
 Begin
  With Form1.Image3.Canvas do
   begin
    Brush.Color:=ClWhite;
    Brush.Style:=bsSolid;
    FloodFill(60,10,clblack,fsBorder);
    Brush.Color:=ClWhite;
    Brush.Style:=bsSolid;
    FloodFill(60,10,clblack,fsBorder);
    pen.Color:=ClRed;
    pen.Width:=2;
    Moveto(53,26);
    LineTo(53,20);
    LineTo(63,20);
    LineTo(63,10);
    LineTo(69,10);
    LineTo(69,20);
    LineTo(79,20);
    LineTo(79,26);
    LineTo(69,26);
    LineTo(69,36);
    LineTo(63,36);
    LineTo(63,26);
    LineTo(53,26);
   end
  End;

Procedure  K1_3;
Var i,n:integer;
xr,yr:mas1;
eps,ug,v,t,du,c:real;
begin
  eps:=0.5;
       c:=15*pi;
         ug:=pi;
         t:=1-eps/10;
         v:=Pi/2-ArcTan(t/(sqrt(1-t*t)));
```

```
          n:=round(ug/2/v);
          if n<=1 then n:=4;
          du:=ug/(n-1);
         for i:=0 to n-1 do
          begin
          xr[i]:=round(10*cos(i*du)+109);
         yr[i]:=round(10*sin(i*du)+28);
         end;
  n:=n+1;
  xr[n-1]:=109;
  yr[n-1]:=8;
  n:=n+1;
  xr[n-1]:=xr[0];
  yr[n-1]:=yr[0];
  With Form1.Image3.Canvas do
    begin
     Brush.Color:=ClWhite;
     Brush.Style:=bsSolid;
     FloodFill(109,20,clblack,fsBorder);
     Brush.Color:=ClWhite;
     Brush.Style:=bsSolid;
     FloodFill(109,20,clblack,fsBorder);
     pen.Color:=ClRed;
     pen.Width:=2;
     MoveTo(xr[0],yr[0]);
     For i:=1 to n-1 do LineTo(xr[i],yr[i]);
    end;
end;


Procedure  K1_4;
 Begin
  With Form1.Image3.Canvas do
    begin
     Brush.Color:=ClWhite;
     Brush.Style:=bsSolid;
     FloodFill(152,20,clblack,fsBorder);
     Brush.Color:=ClWhite;
     Brush.Style:=bsSolid;
     FloodFill(152,20,clblack,fsBorder);
     pen.Color:=ClRed;
     pen.Width:=2;
```

```
    ellipse(137,8,167,38);
   end
  End;



Procedure  K2_1;
 Begin
  With Form1.Image3.Canvas do
   begin
    Brush.Color:=ClWhite;
    Brush.Style:=bsSolid;
    FloodFill(30,53,clblack,fsBorder);
    Brush.Color:=ClWhite;
    Brush.Style:=bsSolid;
    FloodFill(30,53,clblack,fsBorder);
    pen.Color:=ClRed;
    pen.Width:=2;
    Moveto(10,66);
    LineTo(23,53);
    LineTo(36,66);
    LineTo(23,79);
    LineTo(10,66);
   end
  End;

Procedure  K2_2;
 Begin
  With Form1.Image3.Canvas do
   begin
    Brush.Color:=ClWhite;
    Brush.Style:=bsSolid;
    FloodFill(60,60,clblack,fsBorder);
    Brush.Color:=ClWhite;
    Brush.Style:=bsSolid;
    FloodFill(60,60,clblack,fsBorder);
    pen.Color:=ClRed;
    pen.Width:=2;
    Moveto(53,66);
    LineTo(63,63);
    LineTo(66,53);
    LineTo(69,63);
```

```
    LineTo(78,66);
    LineTo(69,69);
    LineTo(66,79);
    LineTo(63,69);
    LineTo(53,66);
   end
  End;

procedure K2_3;
 var
  i,n:integer;
x,y:mas1;
begin
 n:=5;
 for i:=0 to n-1 do
  begin
   x[i]:=round(15*cos(pi*(2*i/n)))+109;
  y[i]:=round(15*sin(pi*(2*i/n)))+66;
  end;
  n:=n+1;
  x[n-1]:=x[0];
  y[n-1]:=y[0];
  With Form1.Image3.Canvas do
    begin
     Brush.Color:=ClWhite;
     Brush.Style:=bsSolid;
     FloodFill(109,60,clblack,fsBorder);
     Brush.Color:=ClWhite;
     Brush.Style:=bsSolid;
     FloodFill(109,60,clblack,fsBorder);
     pen.Color:=ClRed;
     pen.Width:=2;
     MoveTo(x[i],y[i]);
     For i:=1 to n-1 do LineTo(x[i],y[i]);
    end
end;

Procedure  K2_4;
 Begin
  With Form1.Image3.Canvas do
    begin
     Brush.Color:=ClWhite;
```

```
    Brush.Style:=bsSolid;
    FloodFill(152,60,clblack,fsBorder);
    Brush.Color:=ClWhite;
    Brush.Style:=bsSolid;
    FloodFill(152,60,clblack,fsBorder);
    pen.Color:=ClRed;
    pen.Width:=2;
   ellipse(137,61,167,77);
   end
  End;


Procedure  K1_1In;
 Begin
  With Form1.Image3.Canvas do
    begin
    Brush.Color:=ClSkyBlue;;
    Brush.Style:=bsSolid;
     FloodFill(30,10,clblack,fsBorder);
     Brush.Color:=ClSkyBlue;;
     Brush.Style:=bsSolid;
     FloodFill(30,10,clblack,fsBorder);
     pen.Color:=ClRed;
     pen.Width:=2;
     Moveto(10,15);
     LineTo(36,15);
     LineTo(36,31);
     LineTo(10,31);
     LineTo(10,15);
    end
  End;

Procedure  K1_2In;
 Begin
  With Form1.Image3.Canvas do
    begin
    Brush.Color:=ClSkyBlue;
    Brush.Style:=bsSolid;
     FloodFill(60,10,clblack,fsBorder);
     Brush.Color:=ClSkyBlue;
     Brush.Style:=bsSolid;
     FloodFill(60,10,clblack,fsBorder);
```

```
    pen.Color:=ClRed;
    pen.Width:=2;
    Moveto(53,26);
    LineTo(53,20);
    LineTo(63,20);
    LineTo(63,10);
    LineTo(69,10);
    LineTo(69,20);
    LineTo(79,20);
    LineTo(79,26);
    LineTo(69,26);
    LineTo(69,36);
    LineTo(63,36);
    LineTo(63,26);
    LineTo(53,26);
   end
  End;

Procedure  K1_3In;
Var i,n:integer;
xr,yr:mas1;
eps,ug,v,t,du,c:real;
begin
  eps:=0.5;
        c:=15*pi;
          ug:=pi;
          t:=1-eps/10;
          v:=Pi/2-ArcTan(t/(sqrt(1-t*t)));
          n:=round(ug/2/v);
          if n<=1 then n:=4;
          du:=ug/(n-1);
        for i:=0 to n-1 do
         begin
          xr[i]:=round(10*cos(i*du)+109);
         yr[i]:=round(10*sin(i*du)+28);
         end;
  n:=n+1;
  xr[n-1]:=109;
  yr[n-1]:=8;
  n:=n+1;
  xr[n-1]:=xr[0];
  yr[n-1]:=yr[0];
```

```
  With Form1.Image3.Canvas do
    begin
    Brush.Color:=ClSkyBlue;
    Brush.Style:=bsSolid;
     FloodFill(109,20,clblack,fsBorder);
     Brush.Color:=ClSkyBlue;
     Brush.Style:=bsSolid;
     FloodFill(109,20,clblack,fsBorder);
     pen.Color:=ClRed;
     pen.Width:=2;
     MoveTo(xr[0],yr[0]);
     For i:=1 to n-1 do LineTo(xr[i],yr[i]);
    end;
end;



Procedure  K1_4In;
 Begin
  With Form1.Image3.Canvas do
    begin
    Brush.Color:=ClSkyBlue;
    Brush.Style:=bsSolid;
     FloodFill(152,20,clblack,fsBorder);
     Brush.Color:=ClSkyBlue;
     Brush.Style:=bsSolid;
     FloodFill(152,20,clblack,fsBorder);
     pen.Color:=ClRed;
     pen.Width:=2;
    ellipse(137,8,167,38);
    end
  End;



Procedure  K2_1In;
 Begin
  With Form1.Image3.Canvas do
    begin
    Brush.Color:=ClSkyBlue;
    Brush.Style:=bsSolid;
     FloodFill(30,53,clblack,fsBorder);
```

```
    Brush.Color:=ClSkyBlue;
    Brush.Style:=bsSolid;
    FloodFill(30,53,clblack,fsBorder);
    pen.Color:=ClRed;
    pen.Width:=2;
    Moveto(10,66);
    LineTo(23,53);
    LineTo(36,66);
    LineTo(23,79);
    LineTo(10,66);
   end
  End;

Procedure  K2_2In;
 Begin
  With Form1.Image3.Canvas do
   begin
   Brush.Color:=ClSkyBlue;
   Brush.Style:=bsSolid;
    FloodFill(60,60,clblack,fsBorder);
    Brush.Color:=ClSkyBlue;
    Brush.Style:=bsSolid;
    FloodFill(60,60,clblack,fsBorder);
    pen.Color:=ClRed;
    pen.Width:=2;
    Moveto(53,66);
    LineTo(63,63);
    LineTo(66,53);
    LineTo(69,63);
    LineTo(78,66);
    LineTo(69,69);
    LineTo(66,79);
    LineTo(63,69);
    LineTo(53,66);
   end
  End;

 procedure K2_3In;
 var
  i,n:integer;
x,y:mas1;
begin
```

```
 n:=5;
 for i:=0 to n-1 do
  begin
   x[i]:=round(15*cos(pi*(2*i/n)))+109;
  y[i]:=round(15*sin(pi*(2*i/n)))+66;
  end;
  n:=n+1;
  x[n-1]:=x[0];
  y[n-1]:=y[0];
  With Form1.Image3.Canvas do
    begin
    Brush.Color:=ClSkyBlue;
    Brush.Style:=bsSolid;
     FloodFill(109,60,clblack,fsBorder);
     Brush.Color:=ClSkyBlue;
     Brush.Style:=bsSolid;
     FloodFill(109,60,clblack,fsBorder);
     pen.Color:=ClRed;
     pen.Width:=2;
     MoveTo(x[i],y[i]);
     For i:=1 to n-1 do LineTo(x[i],y[i]);
    end
end;


Procedure  K2_4In;
 Begin
  With Form1.Image3.Canvas do
    begin
    Brush.Color:=ClSkyBlue;
    Brush.Style:=bsSolid;
     FloodFill(152,60,clblack,fsBorder);
     Brush.Color:=ClSkyBlue;
     Brush.Style:=bsSolid;
     FloodFill(152,60,clblack,fsBorder);
     pen.Color:=ClRed;
     pen.Width:=2;
    ellipse(137,61,167,77);
    end
  End;
```

```
Procedure Grd(n:integer; Var xr,yr:array of Int64;
          Xc,Yc:Int64; dxc,dyc:integer; mxy:real;p:integer);

  Var i:integer;
   xp,yp:mas1;
   Begin
    If p>15 then
     If P mod 15=0 then P:=15
     else P:=P mod 15;
    for i:=0 to n-1 do
     begin
      xp[i]:=round((xr[i]-Xc)*mxy)+dxc;
      yp[i]:=round((yr[i]-Yc)*mxy)+dyc;
     end;
    With Form1.Image2,Canvas Do
     begin
       Pen.Mode:=pmCopy;
       case p of
     1: pen.Color:=ClRed; {Встановлюємо колір пера}
     2: pen.Color:=ClBlack;
     3: pen.Color:=ClGray;
     4: pen.Color:=ClYellow;
     5: pen.Color:=ClPurple;
     6: pen.Color:=ClSilver;
     7: pen.Color:=ClGreen;
     8: pen.Color:=ClAqua;
     9: pen.Color:=ClBlue;
     10: pen.Color:=ClFuchsia;
     11: pen.Color:=ClTeal;
     12: pen.Color:=ClNavy;
     13: pen.Color:=ClMaroon;
     14: pen.Color:=ClOlive
     else pen.Color:=TColor(RGB(255,128,64));
     end;
       MoveTo(xp[0],yp[0]);
      for i:=1 to n-1 do
      LineTo(xp[i],yp[i]);
      end;
    End;
```

```
procedure XcYc(n:integer;x,y: array of int64; var xc,yc:int64);
var nxi:integer; xi,xe,yi,ye:int64;
begin
     maxmin(nxi,n,x,xi,-1);
     maxmin(nxi,n,y,yi,-1);
     maxmin(nxi,n,x,xe,1);
     maxmin(nxi,n,y,ye,1);
     xc:=round((xi+xe)/2);
     yc:=round((yi+Ye)/2);
end;



procedure TForm1.Button1Click(Sender: TObject);
begin
 Image4.Visible:=False;
 Open1.Visible:=True;
 N1.Visible:=True;
 end;

procedure TForm1.FormCreate(Sender: TObject);
 Var i,j,X1,Y1,X2,Y2:integer;
begin
 Image1.Cursor:=crNone;
 Pr_btn2:=False;
 NRes:=0;
 Left:=0;
 Top:=0;
 Height:=650;
 Width:=850;
 Form1.Caption:='Введення графічної інформації про деталі моделі' ;

  With Image3.Canvas do
   For j:=1 to 4{koldet} do
   begin
      x1:=3+43*(j-1); x2:=43*j;
    For i:=1 to 2 do
     begin
     pen.Color:=Clblack;
    //  Pen.Width:=2;
```

```
      Y1:=3+43*(i-1);
      Y2:=43*i;
  {      Font.Style:=[fsBold];,fsItalic];}
  MoveTo(X1,Y1);
  LineTo(X1,Y2);
  LineTo(X2,Y2);
  LineTo(X2,Y1);
  LineTo(X1,Y1);
 end;
end;
k1_1; K1_2;  K1_3; K1_4;
k2_1; K2_2;  K2_3; K2_4;
{k1_1In;  K1_2In;  K1_3In;  K1_4In;
k2_1In;  K2_2In;  K2_3In;  K2_4In;  }
end;


Procedure Setka;
 Var K1,K2,i,mxyF:integer;
 Begin
     mxyF:=round(500*mxy);
     K1:=round(Form1.image1.Width/mxyF);
     K2:=round(Form1.image1.Height/mxyF);
  with Form1.image1.Canvas do
    begin
      pen.Width:=1;
     pen.Color:=ClBlack;;
     For i:=0 to K2 do
     begin
     MoveTo(0,i*mxyF);
      LineTo(Form1.image1.Width,i*mxyF);
     end;
     For i:=0 to K1 do
     begin
     MoveTo(i*mxyF,0);
      LineTo(i*mxyF,Form1.image1.Height);
     end;
    end;
   End;


procedure TForm1.Image2MouseDown(Sender: TObject; Button: TMouseButton;
```

```
  Shift: TShiftState; X, Y: Integer);
Var NomD,m,i,q,N_r:integer;
     st:string[20];
     Xc,Yc:Int64;
     X_r,Y_r:mas1;
begin
   GroupBox1.Visible:=True;
      NomD:=0;
     for m:=1 to Koldet Do
      begin
       if (Y>=5+55*(m-1))and(Y<=55*m)and
        (X>=5)and(X<=55) then
        begin
         NomD:=m;
         break;
        end;
     end;
     pv:=NomD;

 SCALE_Im1(2*Delta_X[pv],2*Delta_Y[pv],image1.Height,image1.Width, mxy);

 with image1.Canvas do
   begin
    Pen.Mode:=pmCopy;
    pen.Color:=ClWhite;
    brush.Color:=ClWhite;
    brush.Style:=BsSolid;
    rectangle(0,0,image1.Width,image1.Height);
   end;

   XcYc(kolpointdet[pv],xd[pv],yd[pv],xc,yc);

Graph_im1(pv,mxy,kolpointdet[pv],xd[pv],yd[pv],round(image1.Width/2),round(ima
ge1.Height/2), xc,yc);

 For q:=1 to NRes do
  begin
    I:=q;
    If Res[q].Nd=Pv then
     begin
      Case Res[q].Ndek of
       1:Pryam(n_r, X_r,Y_r,Res[q].A,Res[q].B,Res[q].Alf,Res[q].Xcur,Res[q].Ycur);
```

```
      2:krest(n_r,
X_r,Y_r,Res[q].A,Res[q].B,Res[q].D,Res[q].Alf,Res[q].Xcur,Res[q].Ycur);
      3:kaplya(n_r, X_r,Y_r,Res[q].A,Res[q].B,Res[q].Alf,Res[q].Xcur,Res[q].Ycur);
      4:Krug(n_r, X_r,Y_r,Res[q].A,Res[q].Xcur,Res[q].Ycur);
      5:Romb(N_r, X_r,Y_r,Res[q].A,Res[q].B,Res[q].Alf,Res[q].Xcur,Res[q].Ycur);
      6:begin
        N_r:=Res[q].Nr;
        Star(N_r, X_r,Y_r,Res[q].A,Res[q].B,Res[q].Alf,Res[q].Xcur,Res[q].Ycur);
       end;
      7:begin
        N_r:=Res[q].Nr;
        Polig(N_r, X_r,Y_r,Res[q].A,Res[q].Alf,Res[q].Xcur,Res[q].Ycur);
       end;
      8:Oval(n_r, X_r,Y_r,Res[q].A,Res[q].B,Res[q].Alf,Res[q].Xcur,Res[q].Ycur);
     end;
  // XcYc(kolpointdet[pv],xd[pv],yd[pv],xc,yc);
    Graph_im1(pv,mxy,n_r,x_r,y_r,round(image1.Width/2),round(image1.Height/2),
xc,yc);
 end;
 end;

If RadioButton3.Checked then Setka;

end;

procedure TForm1.RadioButton4MouseDown(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
Var  Xc,Yc:Int64;
    q,N_r:integer;
    X_r,Y_r:mas1;
begin
 with image1.Canvas do
   begin
   pen.Color:=ClWhite;
   brush.Color:=ClWhite;
   brush.Style:=BsSolid;
    rectangle(0,0,image1.Width,image1.Height);
   end;
   XcYc(kolpointdet[pv],xd[pv],yd[pv],xc,yc);

Graph_im1(pv,mxy,kolpointdet[pv],xd[pv],yd[pv],round(image1.Width/2),round(ima
ge1.Height/2), xc,yc);
```

```
  For q:=1 to NRes do
 begin
  // I:=q;
   If Res[q].Nd=Pv then
    begin
     Case Res[q].Ndek of
      1:Pryam(n_r, X_r,Y_r,Res[q].A,Res[q].B,Res[q].Alf,Res[q].Xcur,Res[q].Ycur);
      2:krest(n_r,
X_r,Y_r,Res[q].A,Res[q].B,Res[q].D,Res[q].Alf,Res[q].Xcur,Res[q].Ycur);
      3:kaplya(n_r, X_r,Y_r,Res[q].A,Res[q].B,Res[q].Alf,Res[q].Xcur,Res[q].Ycur);
      4:Krug(n_r, X_r,Y_r,Res[q].A,Res[q].Xcur,Res[q].Ycur);
      5:Romb(N_r, X_r,Y_r,Res[q].A,Res[q].B,Res[q].Alf,Res[q].Xcur,Res[q].Ycur);
      6:begin
         N_r:=Res[q].Nr;
         Star(N_r, X_r,Y_r,Res[q].A,Res[q].B,Res[q].Alf,Res[q].Xcur,Res[q].Ycur);
        end;
      7:begin
         N_r:=Res[q].Nr;
         Polig(N_r, X_r,Y_r,Res[q].A,Res[q].Alf,Res[q].Xcur,Res[q].Ycur);
        end;
      8:Oval(n_r, X_r,Y_r,Res[q].A,Res[q].B,Res[q].Alf,Res[q].Xcur,Res[q].Ycur);
     end;
  // XcYc(kolpointdet[pv],xd[pv],yd[pv],xc,yc);
   Graph_im1(pv,mxy,n_r,x_r,y_r,round(image1.Width/2),round(image1.Height/2),
xc,yc);
 end;
 end;
 end;
```