**Івано-Франківський національний технічний університет нафти і газу**

Інститут інженерної механіки
Кафедра комп'ютеризованого машинобудування

Мельник Святослав Русланович

УДК 62-52

# БАКАЛАВРСЬКА РОБОТА

Навчальний робот на основі Arduino та Python

Інженерія мехатронних систем
(назва освітньої програми)

131- Прикладна механіка
(шифр і назва спеціальності)

_____С. Р. Мельник

(підпис, ініціали та прізвище здобувача освітнього ступеня)

Науковий керівник Копей Володимир Богданович, д-р. техн. наук, доцент

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

**Допущено до захисту**

Завідувач кафедри

професор_____Панчук В. Г.

(посада) (підпис) (дата) (ініціали та прізвище)

Рецензент

_____

(посада) (підпис) (дата) (ініціали та прізвище)

Робота містить результати власних розробок. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

м.Івано-Франківськ-2022 рік

**Івано-Франківський національний технічний університет нафти і газу**

(повне найменування закладу вищої освіти)

Інститут інженерної механіки

Кафедра комп'ютеризованого машинобудування

Освітній рівень - бакалавр

Спеціальність 131-Прикладна механіка

(шифр і назва)

ЗАТВЕРДЖУЮ

**Завідувач кафедри**

_____

«____» _____20__ року

# З А В Д А Н Н Я
# НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТОВІ

*Мельнику Святославу Руслановичу*

(прізвище, ім'я, по батькові)

1. Тема роботи *Навчальний робот на основі Arduino та Python*

затверджена наказом закладу вищої освіти від "18" 05 2022 року № 131/7

керівник роботи *Копей Володимир Богданович, д.т.н., доцент*

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

2. Строк подання студентом роботи *17.06.2022 р.*

3. Вихідні дані до роботи *Література з питань проектування роботів, характеристики крокових двигунів і драйверів, технічні вимоги до робота*

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

*1. Огляд конструювання основних частин роботів 2. Огляд та вибір крокових двигунів 3. Вибір драйвера крокового двигуна 4. Схема підключення КД 5. Обмеження струму на обмотках КД 6. Arduino-скетчі для керування кроковими двигунами 7. Python програми для програмування та керування КД 8. Огляд технологій 3D друку та друк деталей робота 9. Проектування робота 10. Оптимізація конструкції колеса*

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

*1. Складальне креслення робота (формат А1)*

*2. Складальне креслення приводу (формат А1)*

*3. Схема підключення двигунів і програма (формат А1)*

*4. Параметрична модель колеса (формат А1)*

*5. Оптимізація конструкції колеса (формат А1)*

6. Консультанти розділів роботи

| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата | |
|---|---|---|---|
| | | завдання видав | завдання прийняв |
| *1-10* | *Копей В.Б., проф. каф. КМВ* | *10.03.22* | *10.03.22* |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

7. Дата видачі завдання *10.03.22*

## КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів бакалаврської роботи | Термін виконання етапів роботи | Примітка |
|---|---|---|---|
| *1.* | *Аналіз літератури* | *30.04.22* | |
| *2.* | *Проектування робота* | *31.05.22* | |
| *3.* | *Оформлення креслень та ПЗ* | *17.06.22* | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

**Студент** _____ *Мельник С. Р.*
( підпис ) (прізвище та ініціали)

**Керівник роботи** _____ *Копей В. Б.*
( підпис ) (прізвище та ініціали)

# Зміст

# АНОТАЦІЯ

кваліфікаційної бакалаврської роботи

" Навчальний робот на основі Arduino та Python "

Розрахунково-пояснювальна записка: 52 с., 26 рис., 10 табл., 27 джерел.

Графічна частина: 5 аркушів формату А1.

За допомогою САПР SOLIDWORKS спроектовано мобільний універсальний робот, який має плату Arduino Uno, два крокові двигуни з драйверами, маніпулятор та пристрій для програмного перемикання механічних передач. Цей пристрій дозволяє підключити додаткові механізми (до 13 ступенів вільності) та маніпулятори і розширити можливості робота. Реалізовано схему підключення крокових двигунів до плати Arduino за допомогою драйверів A4988. Створено програму мовою Python для програмування цього робота з клавіатури. Роботом керує програма, яка виконується на персональному комп'ютері та за допомогою послідовного порту та протоколу Firmata передає дані на мікроконтролер робота. За допомогою SOLIDWORKS Simulation виконано крок оптимізації конструкції колеса робота за критерієм мінімальної жорсткості із використанням обмежень на максимальні еквівалентні напруження. Розроблений робот може бути використаний в майбутніх навчальних проектах з використанням сенсорів, методів машинного зору та машинного навчання.

Студент Мельник С. Р.

## ABSTRACT

of the bachelor's work

" Training robot based on Arduino and Python. "

Paper: 52 pages, 26 figures, 10 tables, 27 references.

The graphical part: 5 sheets of A1 format.

Using SOLIDWORKS CAD a universal mobile robot is designed, which has an Arduino Uno board, two stepper motors with drivers, a manipulator and a device for software switching mechanical gears. This device allows you to connect additional mechanisms (up to 13 degrees of freedom) and manipulators and expand the capabilities of the robot. A scheme for connecting stepper motors to the Arduino board with the help of the A4988 driver was implemented. A Python program was created to program the robot from the keyboard. The robot is controlled by a program that runs on a personal computer and sends data to the robot's microcontroller using the serial port and Firmata protocol. Using SOLIDWORKS Simulation, the robot wheel design was optimized for minimum stiffness using maximum equivalent stress constraints. The developed robot can be used in future training projects using sensors, machine vision and machine learning methods.

Student Melnyk S.R.

# Вступ

**Актуальність роботи.** Сучасний розвиток науки і техніки дозволяє створювати роботи для автоматизації практично будь-якої діяльності людини. Багато таких роботів можна створити самостійно з відносно невеликими витратами коштів. Так платформа Arduino дозволяє порівняно легко створювати прості роботи, які можуть бути запрограмовані за допомогою скетчу Arduino або іншої мови програмування. За допомогою CNC-роутерів та 3D-принтерів є можливість самостійно виготовити деталі такого робота. Проте необхідні базові знання 3D моделювання, програмування, електроніки і механіки.

**Метою роботи** є створення простого навчального мобільного робота, який має мінімум крокових двигунів та максимум функцій, може бути виготовлений на недорогих фрезерних верстатах з ЧПК та 3D-принтерах і запрограмований за допомогою популярної мови програмування Python. Для досягнення мети потрібно **розв'язати наступні задачі**:

1. Виконати огляд конструювання основних частин роботів.
2. Виконати аналіз крокових двигунів та здійснити їх вибір.
3. Вибрати драйвер двигуна та створити схему підключення.
4. Створити програму для керування одним кроковим двигуном та декількома кроковими двигунами за допомогою Arduino скетчу та мови програмування Python.
5. Створити програму для програмування робота на основі двох крокових двигунів та програму для виконання програми робота.
6. Проаналізувати методи 3D друку та виготовити окремі деталі робота на 3D принтері.
7. Спроектувати пристрій для програмного перемикання механічних передач на основі крокових двигунів.

8. За допомогою SOLIDWORKS спроектувати мобільний робот з цим пристроєм та маніпулятором.

9. Виконати оптимізацію конструкції колеса робота за критерієм мінімальної пружності з використанням обмежень на максимальні еквівалентні напруження.

**Основні технічні вимоги** до мобільного робота:

1. Повинен мати мінімальну собівартість, вагу та використовувати мінімальну кількість крокових двигунів.

2. Повинен містити максимум стандартних деталей.

3. Повинен легко виготовлятись на CNC-роутерах та 3D-принтерах.

4. Повинен виконувати програму, яка передається з персонального комп'ютера через послідовний порт.

5. Повинен мати достатню пружність коліс.

6. Повинен бути універсальним та мати можливість підключення додаткових маніпуляторів.

# ОСНОВНА ЧАСТИНА

## 1. Огляд конструювання основних частин роботів

У широкому розумінні робот може бути визначений як технічна система, здатна заміщати людину або допомагати їй у виконанні різних завдань [1]. Маніпулятором називається механічна система, що складається з ланок та робочого органу.

Механізми рук роботів призначені для переміщення, орієнтації об'єктів маніпулювання, зміни захоплювальних пристроїв (ЗП) та інструменту в автоматичному режимі. Конструювання та розрахунок рук доцільно проводити у такій послідовності [2]:

1. Визначити основні параметри ЗП та спроектувати ЗП.

2. Розробити кінематичну схему орієнтуючого механізму кисті та кінематичну схему руки, спроектувати орієнтуючий механізм.

3. Розробити ескізні компонувальні креслення руки.

4. Розрахувати жорсткість трансмісійних валів та муфти.

5. Визначити передатне відношення передавального механізму, визначити навантаження на його вихідну ланку та спроектувати передавальний механізм.

6. Розробити складальне креслення руки.

7. Провести перевірочний розрахунок приводу, трансмісійних валів, передавальних механізмів.

Захоплювальні пристрої (ЗП) роботів служать для захоплення та утримання об'єктів маніпулювання (ОМ) (рис. 1) [3]. ЗП роботів здійснюють такі функції: утримують об'єкт маніпулювання під час його транспортування; орієнтують об'єкт маніпулювання; базують положення об'єкта маніпулювання стосовно маніпулятора [2, 4].

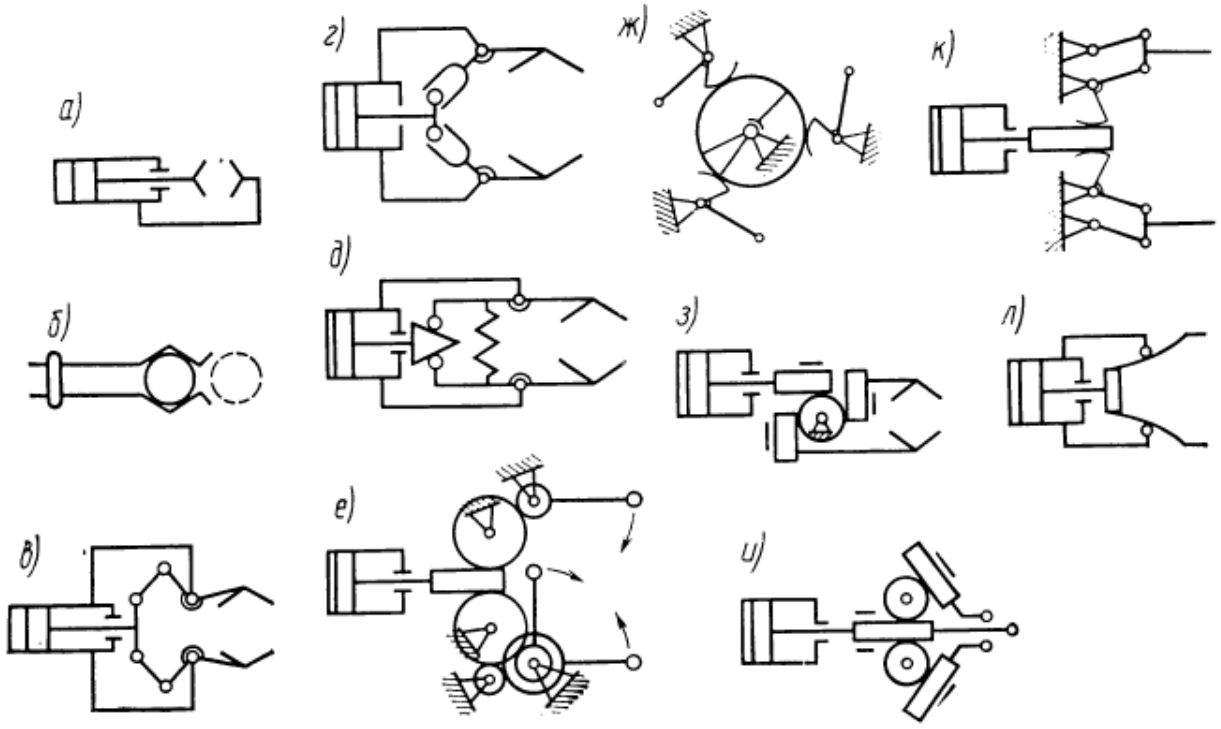



Рисунок 1 - Типи ЗП роботів

Їхні основні функціональні елементи: губки, пальці, двигуни, передачі, корпусні несучі конструкції. Легкозмінні губки з базуючими поверхнями призначені для пристосування робота до специфіки ЗМ. Вони закріплюються на пальцях жорстко або шарнірах з фіксуючими пружинами. У промислових роботах пальці, як правило, односуглобові. Вони можуть бути нерухомими (рис. 1 а), хитатися (рис. 1 б-ж), рухатися поступально прямолінійно (рис. 1 з, і) або по дузі (рис. 1 к), здійснювати складний просторовий рух (рис. 1 л). Завдяки легкості регулювання зусилля ЗП найчастіше оснащуються пневматичними циліндрами, рідше гідро- або електроприводами. Найпростіші ЗП без власного приводу спрацьовують під дією ОМ (рис. 1 б) або зовнішнього обладнання. Двигуни зв'язуються з пальцями безпосередньо (рис. 1 а) або через передачі, що служать для перетворення виду руху, збільшення зусилля затискання або переміщення губок, координації переміщення пальців. Вибір передач значною мірою визначається необхідною залежністю зусилля, що затискає, від розкриття ЗП і відповідно розмірів ОМ. При довільних у певному діапазоні масі та

розмірах ОМ зусилля ЗП бажано мати по можливості постійним. Найкращі щодо цього характеристики у зубчастих передач (рис. 1 е — к). Достатньо стабільне зусилля можна отримати підбором взаємного розташування та розмірів ланок у шарнірно-важільних та копірних передачах (рис. 1 в, г).

Проектування ЗП проводять у такій послідовності [2, 5]:

1. Вибирають тип ЗП.
2. Визначають зусилля захоплення.
3. Визначають потрібне переміщення губок ЗП.
4. Визначають зусилля у кінематичних елементах ЗП.
5. Вибирають тип приводу.
6. Розраховують розміри кінематичних елементів хвата та його приводу.

## 2. Огляд та вибір крокових двигунів

Крокові двигуни (КД) є електромеханічними пристроями, завданням яких є перетворення електричних імпульсів у переміщення валу двигуна на певний кут [6]. Крокові двигуни застосовуються там, де потрібна висока точність переміщень. Перевагами крокових двигунів у порівнянні зі звичайними електродвигунами є:

1. Висока точність позиціонування та повторюваності - якісні КД мають точність не гірше 2,5% від величини кроку, при цьому ця помилка не накопичується при наступних кроках.

2. Кроковий двигун може швидко стартувати, зупинятися та виконувати реверс.

3. Чіткий взаємозв'язок кута повороту ротора з кількістю вхідних імпульсів (в штатних режимах роботи) дозволяє виконувати позиціонування без застосування зворотного зв'язку.

4. Крокові двигуни забезпечують одержання наднизьких швидкостей обертання валу без використання редуктора.

5. Крокові двигуни працюють у широкому діапазоні швидкостей, оскільки. швидкість безпосередньо залежить кількості вхідних імпульсів.

Для керування кроковими двигунами використовують спеціальні пристрої – драйвери крокових двигунів. Популярний драйвер крокового двигуна A4988 [7] працює від напруги 8...35 В і може забезпечити струм до 1 А на фазу без радіатора (до 2 А з радіатором). Модуль A4988 має захист від перевантаження та перегріву. Одним із параметрів крокових двигунів є кількість кроків на один оберт 360 °. Наприклад, для крокових двигунів Nema17 це 200 кроків на оберт, тобто 1 крок дорівнює 1.8 °. Драйвер A4988 дозволяє збільшити це значення за рахунок можливості керування проміжними кроками та має п'ять режимів мікрокроку (1(повний), 1/2, 1/4, 1/8 та 1/16).

Кроковий двигун NEMA 17HS4401 [8] (рис. 2) – дуже вдала модель, яка часто використовується у 3D принтерах та CNC-проектах. Основні характеристики вказані в таблиці 1.


Рисунок 2 – Кроковий двигун NEMA 17HS4401

Таблиця 1 – Параметри крокового двигуна NEMA 17HS4401

| № з/п | Параметр | Значення |
|---|---|---|
| 1. | Кут повороту за один крок | 1,8 º |
| 2. | Діаметр вала | 5 мм |
| 3. | Довжина двигуна | 38 мм |
| 4. | Струм на обмотку | 1,7А |
| 5. | Опір обмотки | 1,5 Ом |
| 6. | Індуктивність обмотки | 2,8 мГн |
| 7. | Крутний момент утримання | 40 Н*см (4,08 кгс*м) |
| 8. | Крутний момент спокою | 2,2 Н*см (0,22 кгс*м) |
| 9. | Інерція ротора | 54 г*см$^2$ |
| 10. | Кількість проводів | 4 |
| 11. | Маса | 280 г |
| 12. | Ціна | ~230 грн (червень 2022 р) |
| 13. | Комплектація | мотор та роз'єм з 4 проводами довжиною 100 см і 4 pin конектором |
| 14. | Розпінування* | червоний 1B, синій 1A, зелений 2A, чорний 2B |

*Якщо крутиться не в ту сторону, перевертаємо контакти дзеркально. Розпінування спеціально зроблено для 3D принтера RAMPS 1.4 і CNC shield верстата з ЧПК.

## 3. Вибір драйвера крокового двигуна

Вибір драйвера для цього двигуна. Можна застосувати мікросхему L293D [9] або L298N [10]. Проте більш простий, надійний і досить дешевий варіант – мікросхема A4988 [11] (табл. 2, 3), яка коштує орієнтовно 30 грн. Мікросхеми-драйвера крокових двигунів такі, як A4988, відрізняються від звичайних Н-

мостів, або драйверів колекторних двигунів таких як L298, можливістю автоматичної стабілізації струму, а також автоматичним формуванням керуючих сигналів на обмотки крокового двигуна для обертання або утримання ротора [12].

Таблиця 2 - Технічні характеристики A4988 [13]

| **Параметр** | **Значення** |
|---|---|
| напруга живлення | 8-35 В |
| режим мікрокроку | 1, 1/2, 1/4, 1/8, 1/16 |
| напруга логіки | 3-5.5 В |
| захист від перегріву | Так |
| максимальний струм на фазу | 1 А без радіатора;<br>2 А з радіатором |
| розмір | 20 х 15 мм |
| вага без радіатора | 2 г |

Таблиця 3 - Призначення контактів драйвера A4988 [13]

| **Контакт** | **Призначення** |
|---|---|
| EN | увімкнення та вимкнення модуля (0 — увімкнено, 5 В — вимкнено). |
| MS1, MS2 та MS3 | вибір режиму мікрокрок (див. таблицю нижче). |
| RST | скидання драйвера. |
| SLP | щоб увімкнути сплячий режим підтягніть до низького стану. |
| STEP | керуючий вивід, при кожному позитивному імпульсі двигун робить крок (залежно від налаштування мікрокроку), чим швидше імпульси, тим швидше |

| Контакт | Призначення |
|---|---|
| | обертатися двигун. |
| DIR | керуючий вивід, якщо подати +5 В двигун буде обертатися за годинниковою стрілкою, а якщо подати 0 - проти годинникової стрілки. |
| VMOT & GND | живлення крокового двигуна двигуна від 8 до 35 В (обов'язкова наявність конденсатора на 100 мкФ). |
| 2B, 2A, 1B та 1A | підключення обмоток двигуна. |
| VDD & GND | живлення внутрішньої логіки від 3 до 5,5 В. |

Значення мікрокроку встановлюється комбінацією сигналів на входах MS1, MS2 та MS3. Є п'ять варіантів подрібнення кроку (див. таблицю 4).

Таблиця 4 - Комбінація значень для вибору мікрокроку [11]

| MS1 | MS2 | MS3 | Режим |
|---|---|---|---|
| 0 | 0 | 0 | Повний крок |
| 1 | 0 | 0 | 1/2 кроку |
| 0 | 1 | 0 | 1/4 кроку |
| 1 | 1 | 0 | 1/8 кроку |
| 1 | 1 | 1 | 1/16 кроку |

Для роботи в режимі мікрокроку потрібен слабкий струм. На модулі A4988 струм можна обмежити потенціометром, що знаходиться на платі. Драйвер дуже чутливий до стрибків напруги живлення двигуна, тому виробник рекомендує встановлювати електролітичний конденсатор великої ємності живлення VMOT для згладжування стрибків.

## 4. Схема підключення КД

Схема підключення драйвера до Arduino [14, 15] і крокового двигуна [16] показана на рис. 3, 4.

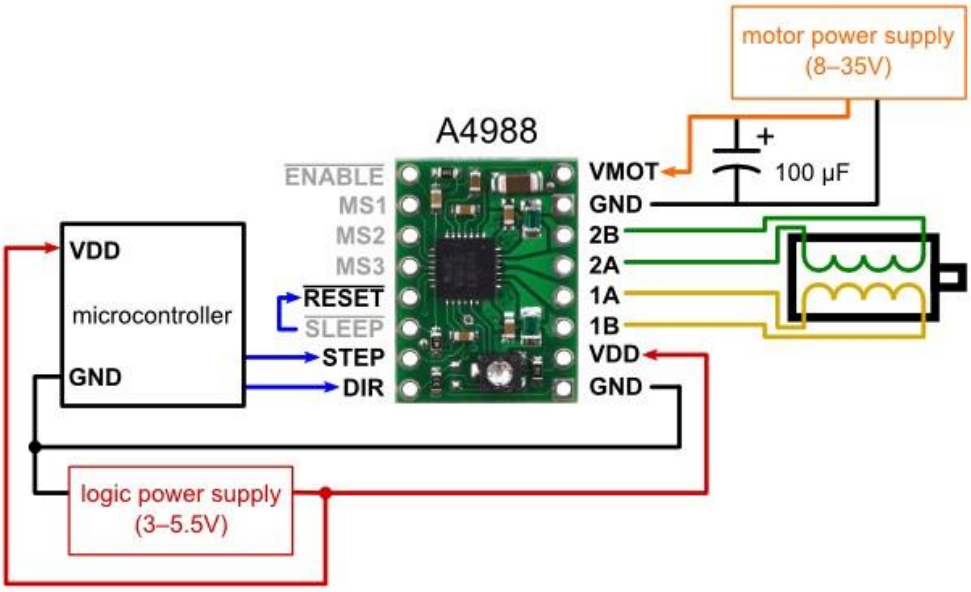



Рисунок 3 – Схема підключення крокового двигуна [15]

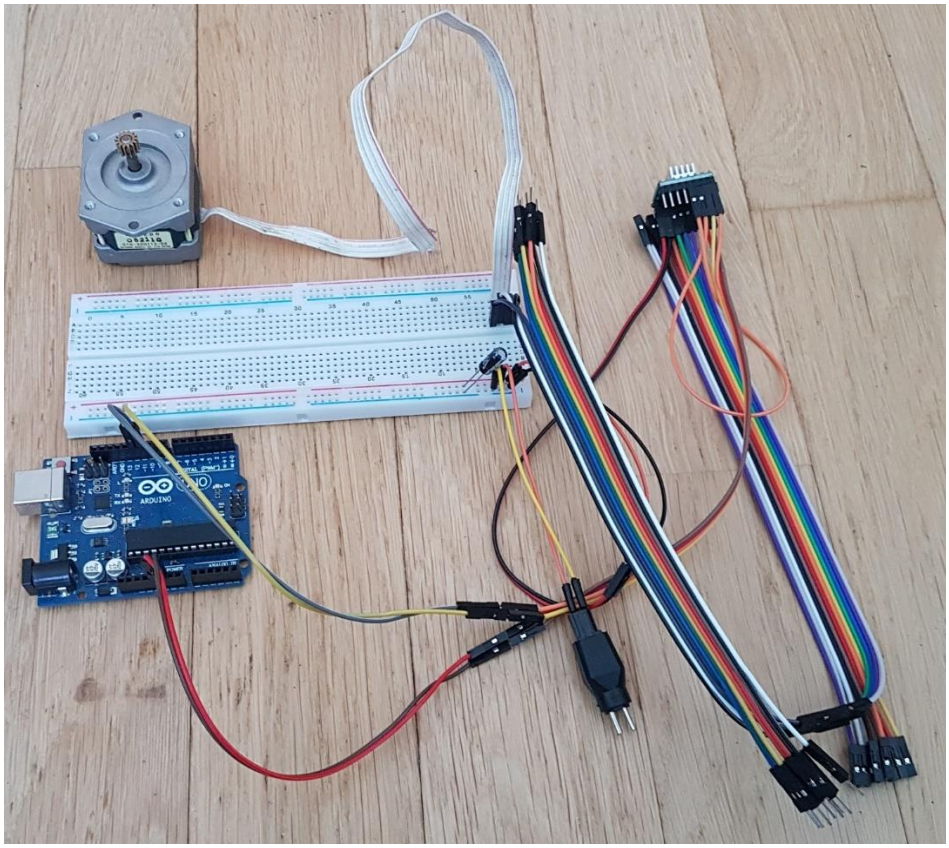


Рисунок 4 – Реалізація підключення одного крокового двигуна до Arduino Uno

Щоб підключити двигун до A4988: піни reset і sleep з'єднайте разом, підключіть VDD до піна 3.3 або 5 на Arduino, підключіть GND до Arduino GND (GND поряд з VDD), підключіть 1A і 1B до 1 котушки крокового двигуна, підключіть 2A і 2B до 2 котушки крокового двигуна, підключіть VMOT до джерела живлення (12 В джерело живлення «+»), // підключіть GRD до джерела живлення (12 В джерело живлення «-»), підключіть 12 пін до step, підключіть 11 пін до dir.

Для живлення двигуна можна взяти адаптер від ноутбука, або блок живлення ПК. Під час вибору джерела живлення зверніть увагу на потужність адаптера. Деякі адаптери може не видавати достатньої сили струму. При неправильному підключенні обмоток двигун не працюватиме. Можна визначити обмотку за допомогою тестера. На контактах має бути близько 1.5 Ом. Ніколи не підключайте чи відключайте двигун, коли на контролер подано живлення. Контролер A4988 може згоріти.

## 5. Обмеження струму на обмотках КД

Перед використанням КД потрібно зробити невелике налаштування, необхідно обмежити максимальну величину струму, що протікає через котушки крокового двигуна та обмежити перевищення номінального струму двигуна. Регулювання здійснюється за допомогою невеликого потенціометра.

Існує два способи налаштування [17]:

1. Заміряти струм, для цього візьмемо амперметр і підключимо його в розрив будь-якої з обмоток (двигун повинен працювати у повнокроковому режимі). При налаштуванні струм повинен становити 70% від номінального струму двигуна.

2. Розрахунок значення напруги *Vref*. Згідно з документацією на A4988, є формула $I\_TripMax = Vref/(8 \times Rs)$, з якої ми можемо отримати формулу:

$$Vref = I\_TripMax \times 8 \times Rs,$$

де *I_TripMax* – номінальний струм двигуна,

*Rs* – опір на резисторі.

У нашому випадку на драйвері A4988 встановлені резистори *Rs* = 0,100 Ом (R100), а номінальний струм двигуна 17HS4401 дорівнює 1,7 А.

$$Vref = 1{,}7 * 8 * 0{,}100 = 1{,}36 \text{ В}$$

Ми розрахували максимальне значення для двигуна 17HS4401, але за такої напруги двигун грітиметься в режимі очікування, необхідно зменшити це значення на 70%, тобто:

$$Vref * 0{,}7 = 0{,}952 \text{ В}.$$

Залишилося тільки налаштувати, беремо викрутку та вольтметр, плюсовий шуп вольтметра встановлюємо на потенціометр, а шуп заземлення на вивід GND та виставляємо потрібне значення.

## 6. Arduino-скетчі для керування кроковими двигунами

Для програмування Arduino використовували літературу [18-20]. Arduino-скетч для тестування крокового двигуна показано нижче. Скетч показує як керувати одним кроком двигуном. У функції setup, яка виконується тільки один раз на початку роботи програми, вводиться режим пінів (виведення) та установлюється напрямок обертання. У функції Loop, яка постійно повторюється, виконується поворот двигуна на один крок. Поворот виконується

шляхом зміни логічної одиниці на піні на логічний нуль з затримкою 10 мілісекунд.

```
int sm = 12; // перепад сигналу 1-0 призводить до повороту на крок
int d = 11; // напрямок - 0 - в одну сторону, 1 - в другу
void setup() // функція виконується один раз на початку
{
pinMode(sm, OUTPUT);  // режим піна - виведення
pinMode(d, OUTPUT);  // режим піна - виведення
digitalWrite(d, HIGH); // напрямок обертання
}
void loop() // функція постійно повторюється
{
// виконати один крок
digitalWrite(sm, HIGH); // подати логічну 1
delay(10); // затримка на 10 мс
digitalWrite(sm, LOW); // подати логічне 0
delay(10);
}
```

Під час обертання ротора двигун може сильно вібрувати, що є проблемою скетчу, а не самого двигуна. Щоб вирішити цю проблему можна використати бібліотеку AccelStepper [21]. Arduino-скетч з цією бібліотекою для тестування крокового двигуна показано нижче:

```
#include <AccelStepper.h> // бібліотека
AccelStepper sm(1, 12, 11); // крековий двигун sm під’єднано до 12, 11
пінів
int d = 1; // напрямок обертання
void setup() { // функція виконується один раз на початку
```

```
sm.setMaxSpeed(3000); // максимальна швидкість, кроків за секунду
sm.setAcceleration(13000); //максимальне прискорення, кроків за секунду в
квадраті
}
void loop() {// функція постійно повторюється
if(sm.distanceToGo()==0){ // чи відробив двигун попередній рух?
sm.move(1600*d); // на 1600 кроків
d = d*(-1); // назад
delay(1000);
}
sm.run(); //запуск двигуна
}
```

## 7. Python програми для програмування та керування КД

Недоліком використання Arduino-скетчу є необхідність прошивання мікроконтролера щоразу коли змінюється програма. Щоб вирішити цю проблему можна завантажити в мікроконтролер скетч StandardFirmata, який підтримує протокол Firmata [22]. За допомогою цього протоколу по послідовному порту на мікроконтролер будуть передаватись дані для керування кроковими двигунами. У майбутньому за допомогою цього протоколу можна буде передавати дані, отримані з сенсорів, які підключені до Arduino робота. Програма, яка керує роботом, виконується на персональному комп'ютері, а дані передаються по послідовному порту [23, 24] на мікроконтролер Arduino. У майбутньому планується реалізувати передачу цих даних за допомогою безпровідного зв'язку – радіозв'язку або оптичного інфрачервоного. Такий спосіб керування дозволяє також виконувати програми, які потребують значних обчислювальних ресурсів і не можуть виконуватись на мікроконтролері Arduino, наприклад, які реалізують методи машинного зору, аналізу даних,

обробки сигналів та машинного навчання [25]. Такий метод програмування є достатньо перспективний, у тому числі і для навчальних цілей.

Розроблено програму stepper.py мовою Python для керування одним крокoвим двигуном з ПК. Не забувайте відключати Arduino, коли підключаєте або відключаєте двигун! Програма мовою Python імпортує необхідні функції з модуля pyfirmata. Далі створюється об'єкт board, який являє собою плату Arduino, що підключена до послідовного порта із заданою швидкістю. Створюються об'єкти пінів кроку і напрямку. В циклі для різних значень напрямку виконується поворот двигуна на 10000 кроків. Код програми:

```
 # -*- coding: utf-8 -*-
from pyfirmata import Arduino, util # імпорт з модуля
time=util.time
board = Arduino('COM19', baudrate=57600) # об’єкт плати Arduino

s1=board.get_pin('d:12:o') # пін кроку
s2=board.get_pin('d:11:o') # пін напрямку

for d in [0,1,0,1]: #
    s2.write(d) # установити напрямок
    for i in range(10000): # обертати на 10000 кроків
        s1.write(1)
        time.sleep(0.001)  # затримка на 1 мс
        s1.write(0)
        time.sleep(0.001) # затримка на 1 мс

board.exit() # завершити роботу з платою
```

Далі розроблено програму stepper2.py мовою Python для програмування робота з 2-ма крокoвими двигунами. Знову не забувайте відключати Arduino,

коли підключаєте або відключаєте двигуни! Ця програма додатково використовує ще два цифрові виводи для другого крокового двигуна. Розроблено клас Motor, який описує поняття крокового двигуна з такими функціями як конструктор класу, функція, що здійснює поворот на один крок, функція, що здійснює поворот в заданому напрямку на задану кількість кроків, функція, що здійснює поворот з поточного положення в задане. Далі створено два об'єкта цього класу та введено початкове значення атрибута value. Для програмування робота використовуються засоби програмування графічного інтерфейсу користувача на основі пакету Tkinter [25]. Це дозволяє перехоплювати події клавіатури, як натиск на клавіші зі стрілками. Якщо виконується подія натиску клавіші, то запускається пов'язана з подією функція-хендлер. Далі визначається код клавіші і виконується поворот двигуна в заданому напрямку на один крок. Якщо робот опинився в заданому положенні, користувач натискайте клавішу P, щоб запам'ятати поточні кути повороту двигуна. Ці кути додаються у список, який і являє собою програму для робота. Код програми:

```
# -*- coding: utf-8 -*-  # кодування символів файлу
from pyfirmata import Arduino, util
time=util.time
board = Arduino('COM19', baudrate=57600) # плата Arduino на порті COM19

s1=board.get_pin('d:12:o') # пін кроку двигуна 1
d1=board.get_pin('d:11:o') # пін напрямку двигуна 1
s2=board.get_pin('d:10:o') # пін кроку двигуна 2
d2=board.get_pin('d:9:o') # пін напрямку двигуна 2

class Motor:
    """Клас крокових двигунів"""
```

```
    def __init__(self, pin_step, pin_dire): # конструктор класу
        "pin_step - пін кроку, pin_dire - пін напрямку"
        self.pin_step=pin_step
        self.pin_dire=pin_dire
        self.value=0 # поточна кількість кроків
        self.dire=1 # нарямок 1 або -1

    def step(self):
        "Поворот на один крок"
        self.pin_step.write(1) # вивести логічну 1
        time.sleep(0.001)
        self.pin_step.write(0) # вивести логічне 0
        time.sleep(0.001)
        self.value+=self.dire  # збільшити/зменшити значення

    def move(self, steps=1):
        "Поворот в напрямку self.dire на steps кроків"
        if self.dire==1: # якщо напрямок 1
            self.pin_dire.write(1) # установити напрямок 1
        else: # інакше
            self.pin_dire.write(0) # установити напрямок 0
        for i in range(steps): # повернути на steps кроків
            self.step()

    def moveTo(self, value):
        "Поворот з поточного положення в положення value"
        steps=value-self.value # визначити кількість кроків
        self.dire = -1 if steps<0 else 1 # визначити напрямок
        self.move(abs(steps)) # повернути на steps кроків


m1=Motor(s1, d1) # двигун 1
```

```
m1.value=0 # введіть початкове значення!
m2=Motor(s2, d2) # двигун 2
m2.value=0 # введіть початкове значення!

def key_handler(event=None):
    "Подія натиску клавіші"
    m=None # двигун
    if event: # якщо клавіша натиснута
        k=event.keycode # код клавіші
        print "keycode=",k
        if   k==39: m,d = m1,-1 # натиск <вправо>
        elif k==37: m,d = m1,1 # натиск <вліво>
        elif k==40: m,d = m2,-1 # натиск <вниз>
        elif k==38: m,d = m2,1 # натиск <вверх>
        elif k==27: print P; board.exit(); r.destroy(); # вихід <Esc>
        elif k==80: addPoint() # додати точку <P>
        else: m=None
        if m: # якщо двигун обрано
            m.dire=d # напрямок
            m.move(10) # повернути на 10 кроків
            print m.value # вивести значення

P=[] # опорні точки
def addPoint():
    "Додає опорну точку в список"
    p=m1.value, m2.value # точка як кортеж
    P.append(p) # додати точку
    print p

if __name__=="__main__": # якщо модуль виконується, а не імпортується
    import Tkinter as tk
    r = tk.Tk() # створити вікно
```

```
    r.bind('<Key>', key_handler) # пов’язати функцію з подією
    r.mainloop() # головний цикл GUI
```

Також розроблено мовою Python програму stepper2_run.py, яка виконує програму для робота з 2-ма кроковими двигунами. Програма містить функцію runProgram, якій передається список-програма для робота. Алгоритм простий – для кожного елемента списку виконати функцію moveTo кожного двигуна. Код програми:

```
# -*- coding: utf-8 -*-
from stepper2 import  m1, m2, board # імпортувати двигуни і плату

def runProgram(prog):
    "Виконує програму prog для робота з 2-ма кроковими двигунами"
    i=1 # номер кадру (точки)
    for p1,p2 in prog: # для кожної точки
        print i, [p1,p2] # вивести
        m1.moveTo(p1) # рухатись двигуну m1 в p1
        m2.moveTo(p2) # рухатись двигуну m2 в p2
        i+=1

# програма для робота як список точок:
prog=[(100, 0),(100, 100), (100, 200), (100, 800), (100, 900), (200, 0),
(100, 800), (800, 700), (800, 600), (800, 700), (800, 600), (800, 700),
(800, 700), (800, 800), (800, 700), (100, 700), (100, 800), (100, 600),
(100, 500), (200, 600), (900, 500)]

runProgram(prog) # виконати програму
board.exit() # завершити
```

## 8. Огляд технологій 3D друку та друк деталей робота

Технологій 3D друку існує дуже багато. Основні типи [26] :

1. Екструзійний друк. В основі цих методів лежить видавлювання (екструзія) витратного матеріалу із послідовним формуванням готового виробу. Як правило, витратні матеріали складаються з термопластиків або композитних матеріалів на їх основі. Це такі методи, як:

1.1. Пошарове наплавлення (FDM).

1.2. Багатоструминний друк (MJM).

2. Плавлення, спікання чи склеювання

Цей підхід ґрунтується на поєднанні порошкового матеріалу в єдине ціле. Формування проводиться у різний спосіб.

2.1. Найбільш простим є склеювання, як у випадку зі струменевим тривимірним друком (3DP). Подібні принтери наносять на робочу платформу тонкі шари порошку, які потім склеюються вибірково сполучним матеріалом. Порошки можуть складатися практично з будь-якого матеріалу, який можна подрібнити до стану пудри - пластику, деревини, металу. Найбільш популярними ж у цій категорії стали технології лазерного спікання (SLS та DMLS) та плавки (SLM), що дозволяють створювати цільнометалеві деталі.

2.2. Стереолітографічні принтери використовують спеціальні рідкі матеріали, які називаються «фотополімерними смолами». Термін "фотополімеризація" вказує на здатність матеріалу твердніти під впливом світла.

2.3. Ламінування. Деякі 3D-принтери вибудовують моделі, використовуючи листові матеріали – папір, фольгу, пластикову плівку. Шари матеріалу наклеюються один на одного і обрізаються за контурами цифрової моделі за допомогою лазера або леза.

FDM – мабуть, найпростіший і доступніший метод тривимірного друку, що обумовлює його високу популярність. FDM-принтери призначені для друку

термопластиками, які зазвичай постачаються у вигляді тонких ниток, намотаних на котушки. Асортимент «чистих» пластиків дуже широкий: полілактид або «PLA-пластик», ABS-пластик, нейлон, полікарбонат, поліетилен, «PVA-пластик» та інші.

Екструдер – це друкована головка FDM-принтера. Екструдер призначений для плавки та нанесення термопластикової нитки. Механізм подачі нитки складається з валиків і шестерень, що рухаються електромотором. Механізм здійснює подачу нитки в спеціальну металеву трубку, що нагрівається, з соплом невеликого діаметру, звану «хот-енд» або просто «сопло».

Хот-енд служить для нагрівання та плавлення нитки, що подається. Діаметр сопла обумовлює товщину розплавленої нитки і, як наслідок, впливає на якість друку. Нагрівання хот-енду регулюється термістором. Регулювання температури дуже важливе. Для того щоб нитка не розплавилася надто рано, верхня частина хот-енду охолоджується за допомогою радіаторів та вентиляторів.

Кількість екструдерів може змінюватись в залежності від призначення 3D-принтера. Найпростіші варіанти використовують одну друкувальну голівку.

Побудова моделей відбувається на спеціальній платформі, що часто оснащується нагрівальними елементами. Підігрів потрібно для роботи з цілим рядом пластиків, включаючи популярний ABS, схильних до високого ступеня усадки при охолодженні. Швидка втрата об'єму холодними шарами в порівнянні зі свіжонанесеним матеріалом може призвести до деформації моделі або розшарування. Підігрів платформи дозволяє значно вирівнювати градієнт температури між верхніми і нижніми шарами. Проте підігрів потрібен не завжди.

Платформа вимагає калібрування перед печаткою, щоб сопло не торкалося нанесених шарів і не відходило занадто далеко, викликаючи друк «по повітрю», що призводить до утворення «вермішелі» із пластику. Процес калібрування може бути як ручним, і автоматичним. У ручному режимі калібрування проводиться позиціонуванням сопла в різних точках платформи та регулювання нахилу платформи за допомогою опорних гвинтів для досягнення оптимальної дистанції між поверхнею і соплом. Для кращого схоплювання першого шару моделі з поверхнею столика найчастіше застосовуються додаткові засоби, наприклад, цукровий сироп.

Один з варіантів, що набирають популярності, є використання дельтаподібної системи координат. Подібні пристрої промисловості називають «дельта-роботами». У дельта-принтерах друкована головка підвішується на трьох маніпуляторах, кожен з яких пересувається вертикальною напрямною. Синхронний симетричний рух маніпуляторів дозволяє змінювати висоту екструдера над платформою, а асиметричний рух спричиняє усунення головки в горизонтальній площині. Дельта-принтери (рис. 5) мають циліндричну область побудови, які конструкція полегшує збільшення висоти робочої зони з мінімальними змінами дизайну з допомогою подовження направляючих.

З використанням такого принтера та слайсера Cura була надрукована шестерня - деталь робота (рис. 6). Матеріал деталі – PETG – сополімер поліетилентерефталата (PET). Це досить довговічний пластик, який найкраще підходить до цього принтера. Під час друку таких маленьких деталей слід уважно підбирати температуру і швидкість друку. В іншому випадку можливий перегрів і значні деформації.

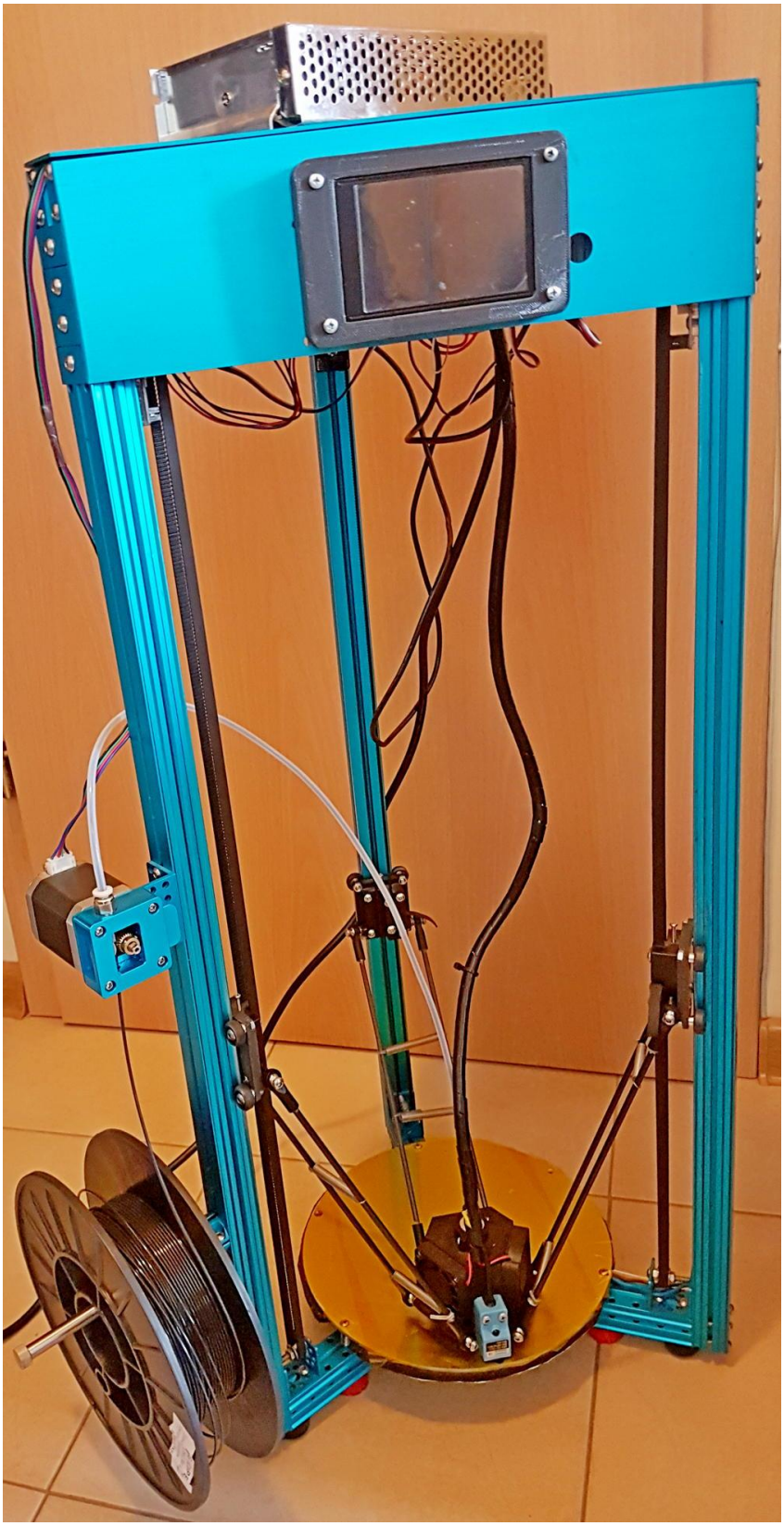


Рисунок 5 – Модифікований дельта-принтер Makeblock mGiraffe

Рисунок 6 - Шестерня, надрукована на 3D принтері

Управління роботою FDM-принтера, включаючи регулювання температури сопла та платформи, темпу подачі нитки та роботи покрокових моторів, що забезпечують позиціонування екструдера, виконується досить простими електронними контролерами. Більшість контролерів ґрунтуються на платформі Arduino, що має відкриту архітектуру.

Програмна мова, що використовується принтерами, називається G-код (G-Code). G-код компілюється програмами, які називаються «слайсерами» – стандартним програмним забезпеченням 3D-принтерів, що поєднує деякі функції графічних редакторів з можливістю встановлення параметрів друку через графічний інтерфейс.

## 9. Проектування робота за допомогою SOLIDWORKS

Метою проектування є створення універсального рухомого робота з маніпулятором, який має мінімальну кількість крокових двигунів і деталі якого можна надрукувати на 3D принтері або вирізати на CNC-роутері. В конструкції також широко використовуються стандартні деталі такі як осі, гвинти, кутники та кріпильні деталі. Під час проектування намагались зменшити кількість унікальних деталей. Наприклад широко в конструкції використовується один і той самий сталевий кутник. Конструкція передньої осі (рис. 7) зібрана майже цілком з таких кутників. Це дозволить зменшити витрати на створення робота та збільшити його міцність.

Окремою проблемою є мінімізація крокових двигунів чи сервоприводів робота. Така мінімізація дозволить зекономити на витратах на їхнє придбання, проте вимагає спеціального механізму для перемикання механічних передач. Розроблнно свого роду «механічний демультиплексор», який складається з двох покрокових двигунів (рис. 8). Один з них здійснює перемикання передач за допомогою  передачі гвинт-гайка, а другий приводить в дію той чи інший механізм робота. Обидва крові двигуни керуються з мікроконтролера Arduino. Конструкція демультиплексора дозволяє його виготовлення на 3D принтерах та cnc-роутерах. Використовується стандартний гвинт, сталеві напрямні осі, муфта, кріпильні деталі та надруковані на 3D принтері шестерні і корпус. Корпус виготовляється з пластикових деталей та може бути надрукований на 3D принтері або вирізаний на CNC-роутері. Запропонована конструкція дозволяє

зменшити зношування шестерень, коли здійснюється перемикання передач. Для цього під час обертання гвинтового гвинта шестерня повинна здійснювати узгоджений з гвинтом поворот. Таким чином усі шестерні будуть знаходитись положенні, яке не дозволяє пошкодження їхніх зубців під час зміни передачі. Запропонований пристрій має шість вихідних валів з однієї сторони і шість другої та дозволяє підключення 13 механізмів. Не всі вони задіяні в конструкції робота, але можуть бути використані в майбутньому. Наприклад для створення додаткової руки або інших механізмів.

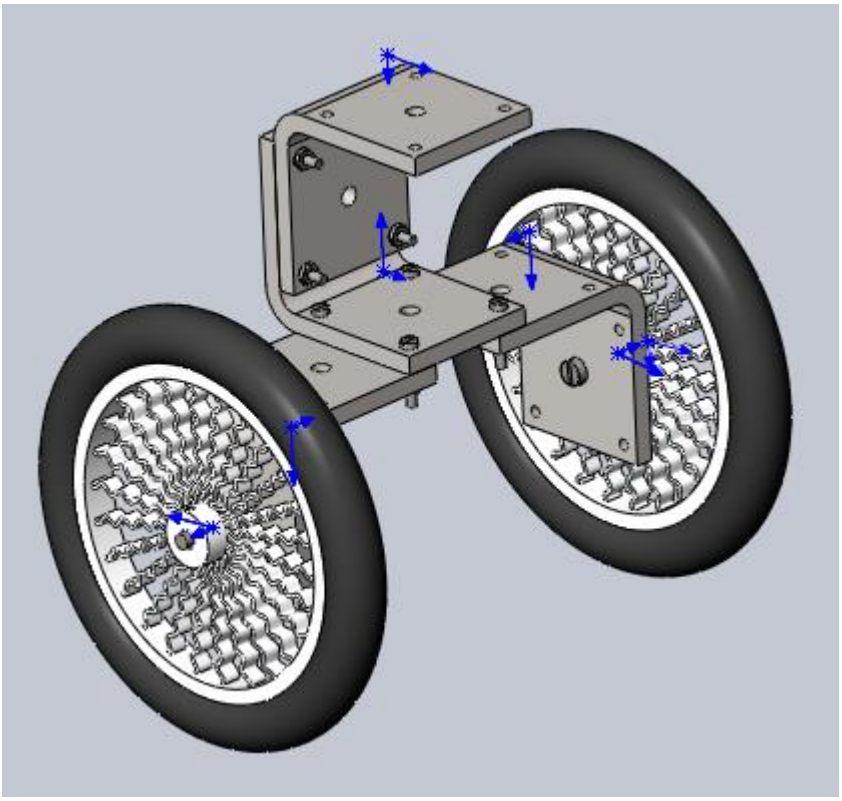


Рисунок 7 - Модель передньої осі

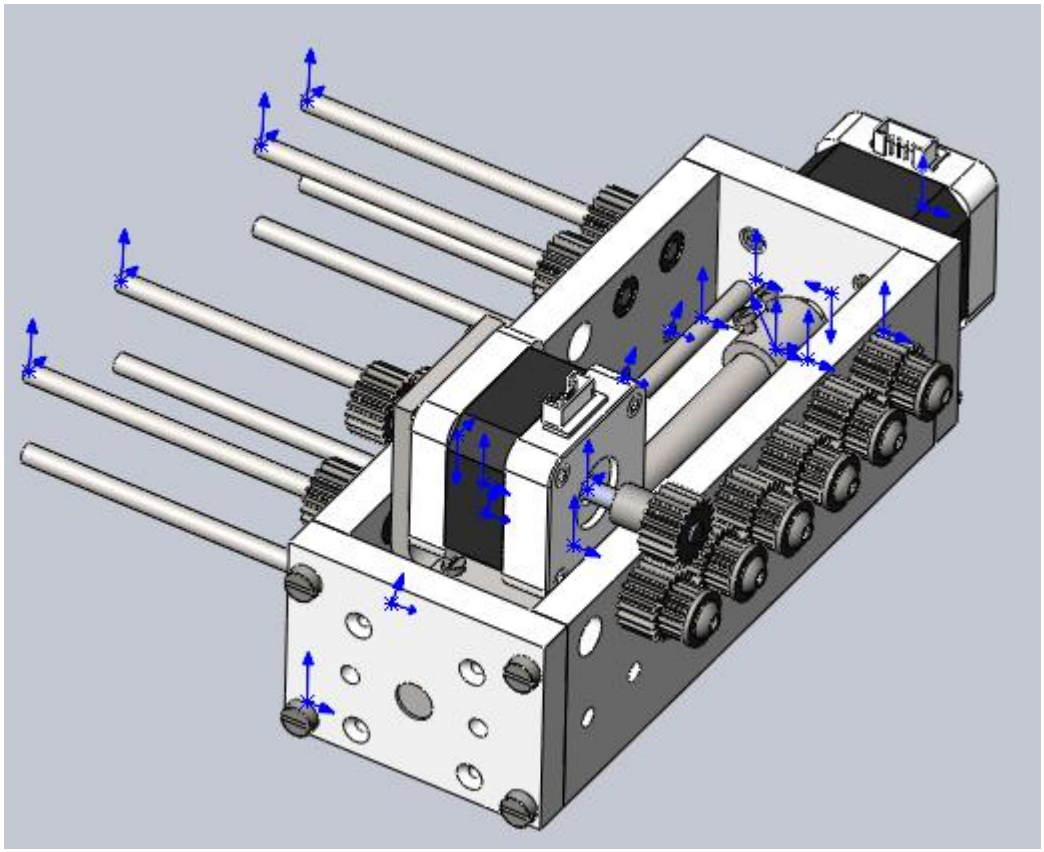


Рисунок 8 - Модель пристрою для перемикання передач

Конструкція робота дозволяє працювати йому за програмою або бути керованим людиною.

Конструкція робота також дозволяє установку важкого акумулятора в нижній полиці. Виявлено, що конструкція витримує до 20 кг ваги.

Робот має модульну конструкцію. Наприклад рука робота може бути замінена на інший її варіант. В даній конструкції рука робота складається з пластикових деталей з’єднаних сталевими гвинтами. Рух ланок робота здійснюється шляхом переміщення тяг з кульовими шарнірами. На захват (ЗП) робота (рис. 9) передається рух за допомогою гнучкого вала. В моделі цей вал не показано. А також не показано пас, який приводить в рух задню вісь. В майбутньому планується удосконалити конструкцію руки і збільшити її надійність.

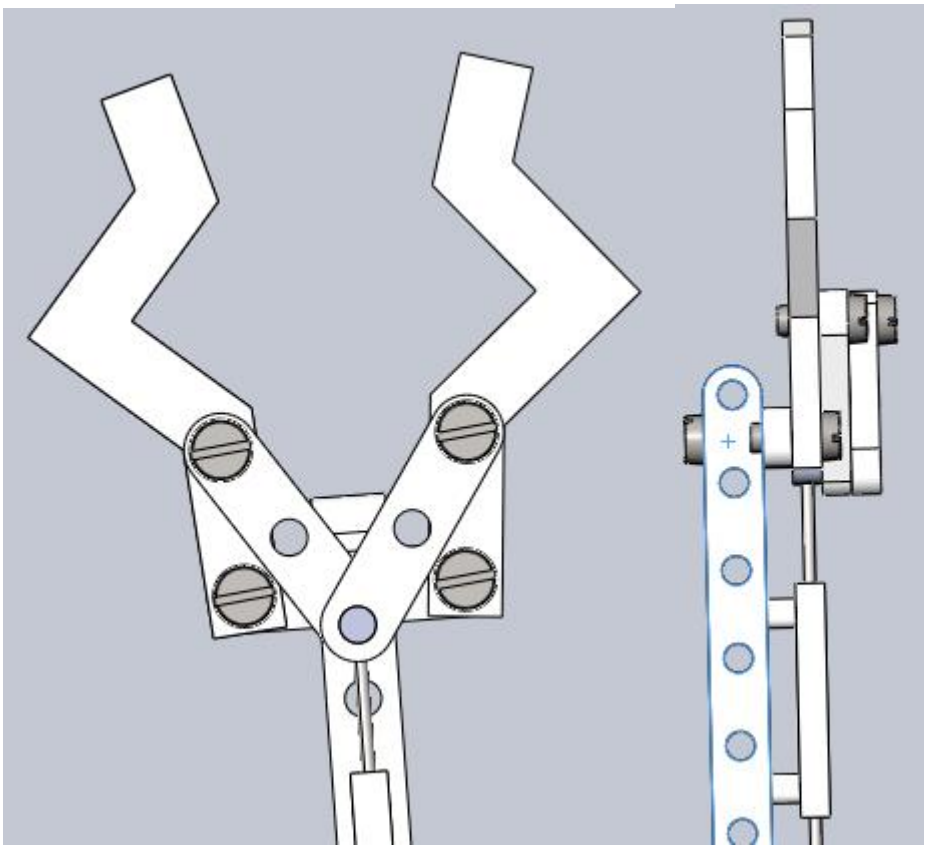


Рисунок 9 - Модель ЗП маніпулятора

Спроектований робот (рис. 10) може бути використаний для навчальних цілей. Студенти можуть практикуватись у програмуванні різних механізмів, які підключені до цього робота. Орієнтовна собівартість цього робота не перевищує 2000 грн при умові що деталі будуть виготовляти на 3D принтері та cnc-роутері. Також цей робот може бути використаний з віддаленим керуванням у недоступних для людини місцях. Завдяки установленню додаткових механізмів є можливість забезпечити надлишковий запас надійності для цього робота - у випадку поломки або неможливості функціонування одного механізму буде задіяно інший. Але головне - забезпечити достатньо високий рівень надійності базового механізму для перемикання передач.

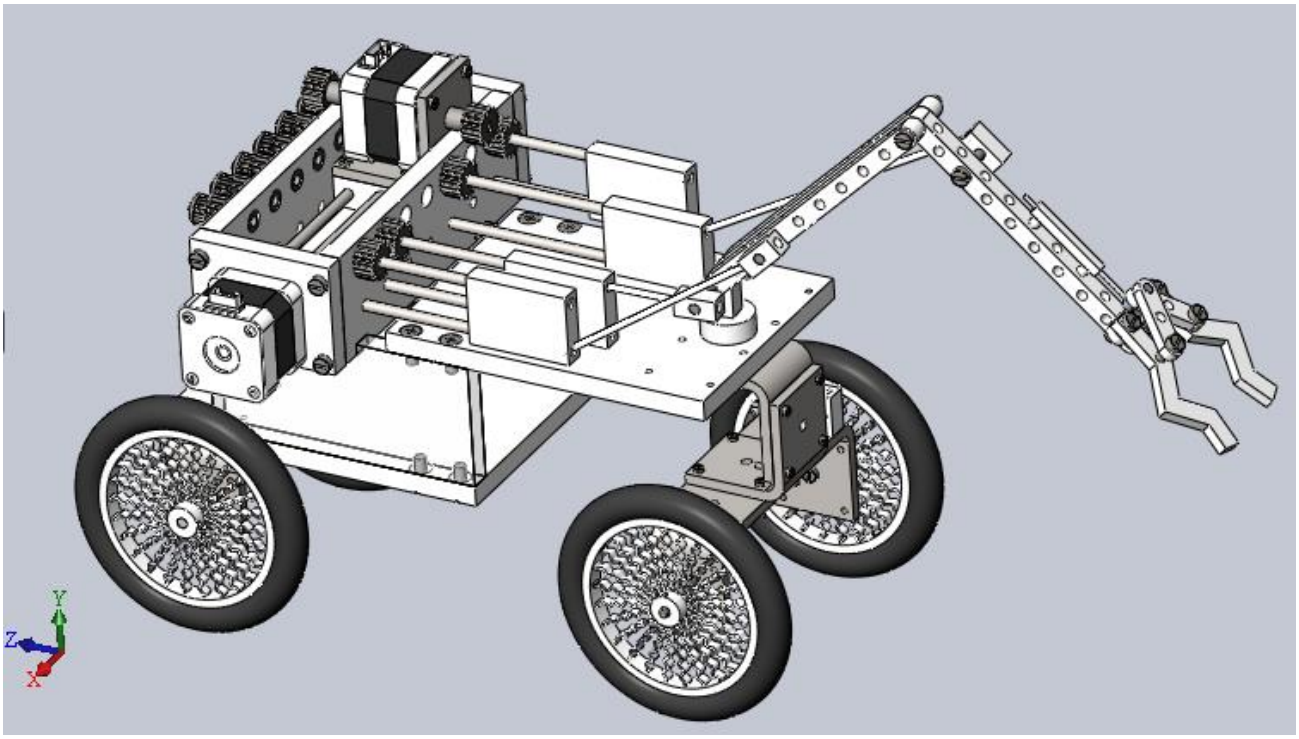


Рисунок 10 - Модель мобільного робота

Нижче детально показано послідовність побудови параметричної моделі колеса. Ця модель буде використана для оптимізації конструкції колеса в наступному розділі. Тому потрібно зробити її повністю параметричною - забезпечити можливість перебудови після зміни значення будь-якого параметра.

Спочатку створено модель шини шляхом повороту круга навколо осі (рис. 11). Потім створюємо колесо без шини - будуємо обід шляхом повороту (рис. 12а) і ступицю шляхом видавлювання (рис. 12б). Створюємо кріпильний отвір в ступиці (рис. 13а) і дзеркальну копію її половини (рис. 13б). Далі будуємо спицю на основі кривої синусоїди (рис. 14а). Є можливість змінити параметри цієї синусоїди. Спиці надаємо товщину (рис. 14б) і створюємо круговий масив спиць (рис. 15а). Після цього створюємо колесо в зборі (рис. 15б).

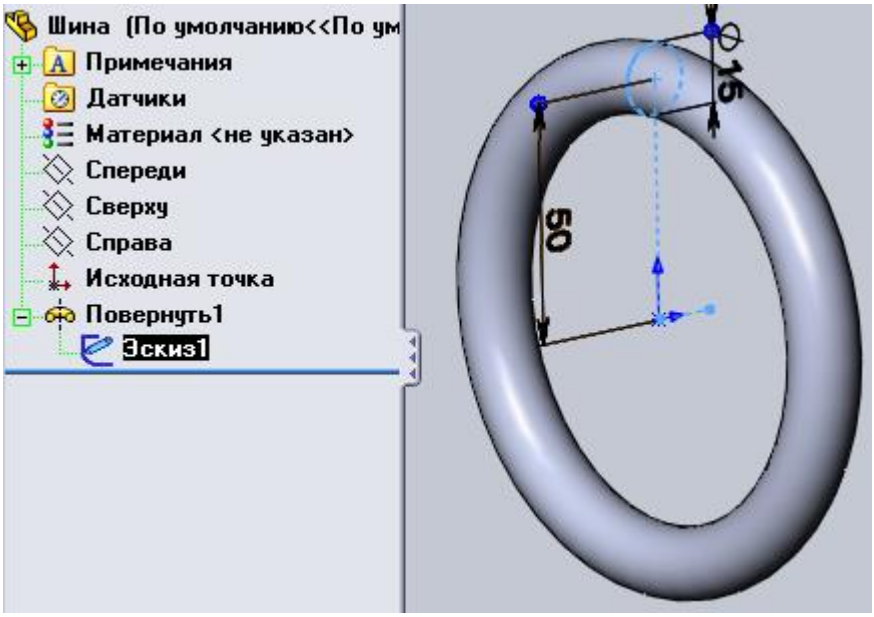



Рисунок 11 - Параметрична модель гумової шини


Рисунок 12 - Обід колеса (а) і ступиця (б)

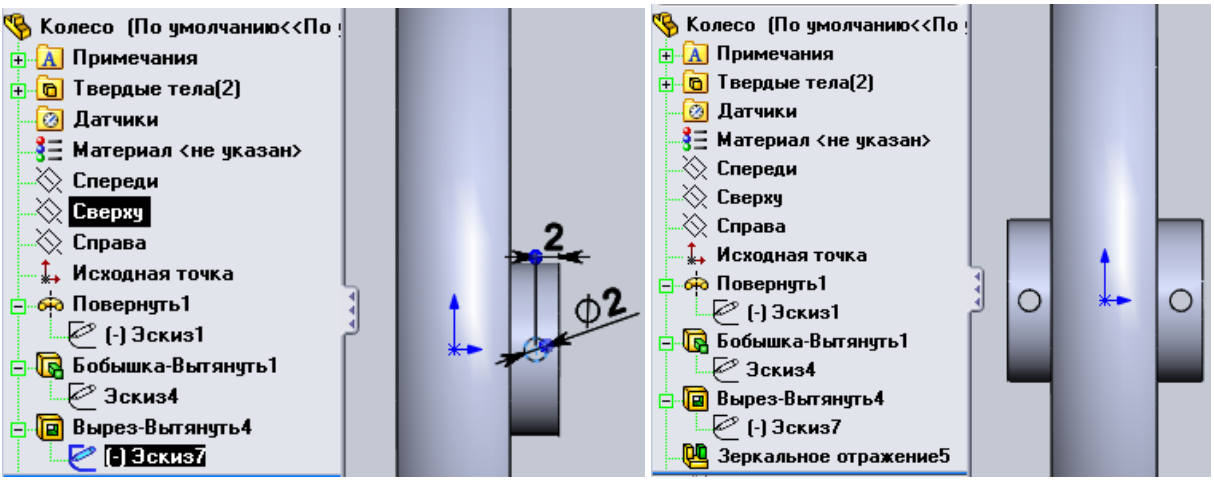
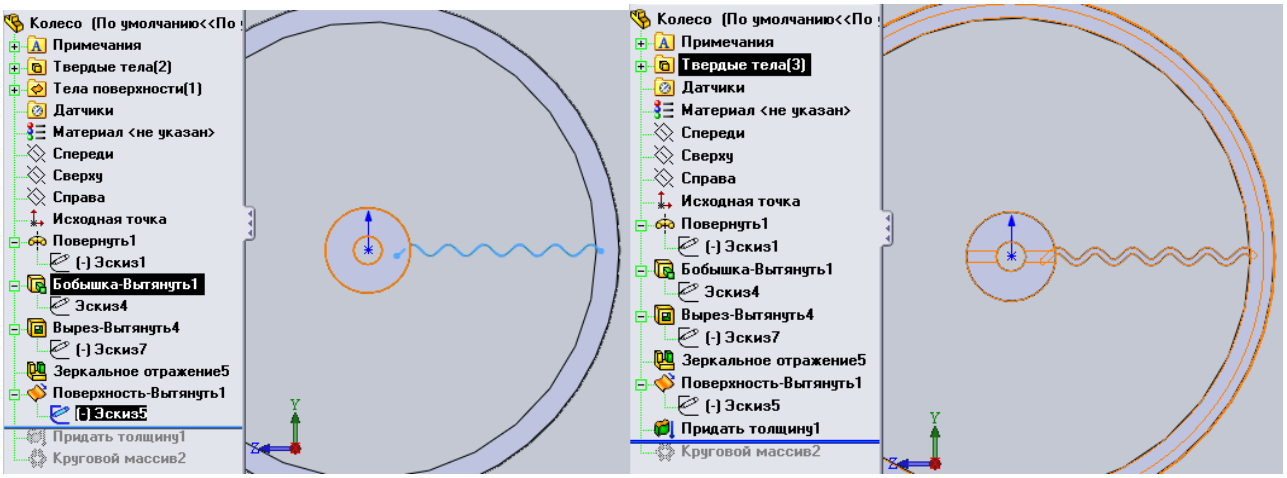



Рисунок 13 - Отвір в ступиці (а) і дзеркальна копія тіла (б)




а б

Рисунок 14 - Крива для побудови спиці (а) та спиця (б)


а б

Рисунок 15 - Масив спиць (а) і колесо в зборі (б)

## 10. Оптимізація конструкції колеса у SolidWorks Simulation

Метою скінченного елементарного аналізу у SolidWorks Simulation 2012 є отримання такої конструкції колеса, яка володіє достатньою пружністю, а напруження у неї менші ніж допустимі. Використано методику дослідження, викладену у праці [27].

Для цього у SolidWorks 2012 побудовано параметричну тривимірну модель колеса в зборі, яка містить гумову шину та пластикове колесо (ступицю, обід і спиці). Колесо має чимало параметрів, але основними з них, які

впливають на його пружність, є: форма спиці, товщина спиці, кількість спиць, ширина спиці. Основні параметри для оптимізації конструкції наведені в таблиці 5.

Таблиця 5 - Параметри моделі, обрані для дослідження

| **Параметр** | **Позна-чення** | **Назва в моделі** | **Границі** |
|---|---|---|---|
| Параметр f функції sin(f*x), яка утворює форму спиці | *f* | параметр рівняння кривої в Эскиз5 | 1..1,4 |
| Товщина спиці | *h* | D1@Придать толщину1 | 0,2..0,6 мм |
| Кількість спиць | *k* | D1@Круговой массив2 | 8..26 |
| Ширина спиці | *s* | D1@Поверхность-Вытянуть1* | 6..15 мм |

*фактично змінюється в налаштуваннях FEA «плоске напруження»

Були вибрані допустимі границі зміни значень параметрів. Алгоритм оптимізації полягає в тому, що для початкового варіанту конструкції отримують залежності напружень та деформацій в конструкції від значень кожного параметра. Після цього значення кожного параметра змінюється на один крок або невелику величину в напрямку зменшення величини жорсткості (збільшення деформацій) колеса. Далі знову отримують такі залежності для нового варіанта конструкції. Цей процес повторюють у разі необхідності. Фактично це є градієнтний метод оптимізації, який реалізується вручну. Під час оптимізації слідкують, щоб напруження не перевищили допустимих.

Наперед ми не знаємо як впливають параметри колеса на напруження і деформації. Проте аналіз дозволить це виявити. Матеріалами колеса є пластик ABS та гума (табл. 6). Слід зауважити що границя міцності колеса, яке виготовляється з пластика ABS - 30 МПа, тому під час оптимізації не можна

збільшувати значення напружень до цієї границі. А краще забезпечити подвійний запас міцності $\sigma < [\sigma]$, де $[\sigma] = \sigma_в/2 = 15$ МПа.

Таблиця 6 - Механічні характеристики матеріалів конструкції

| **Механічна характеристика** | **Пластик ABS** | **Гума** |
|---|---|---|
| Модуль пружності | 2000 МПа | 0,01 МПа |
| Коефіцієнт Пуассона | 0,394 | 0,45 |
| Границя міцності на розрив $\sigma_в$ | 30 МПа | 20 МПа |
| Густина | 1020 кг/м$^3$ | 960 кг/м$^3$ |

Для спрощення обчислень розглянемо двовимірну скінченно-елементну задачу з плоским напруженням. В цьому випадку товщина моделі вводиться в параметрах аналізу (рис. 16).

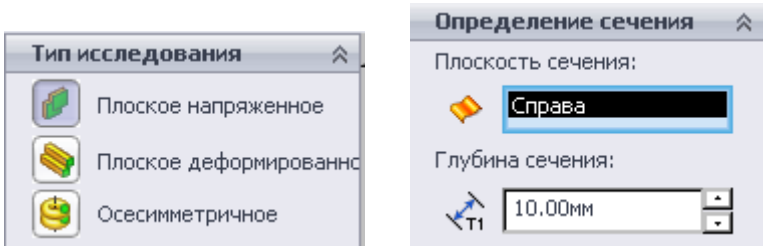



Рисунок 16 - Параметри аналізу

Колесо позбавлене усіх ступенів вільності на внутрішній циліндричній поверхні (рис. 17). На колесо діє сила 50 Н (рис. 17).

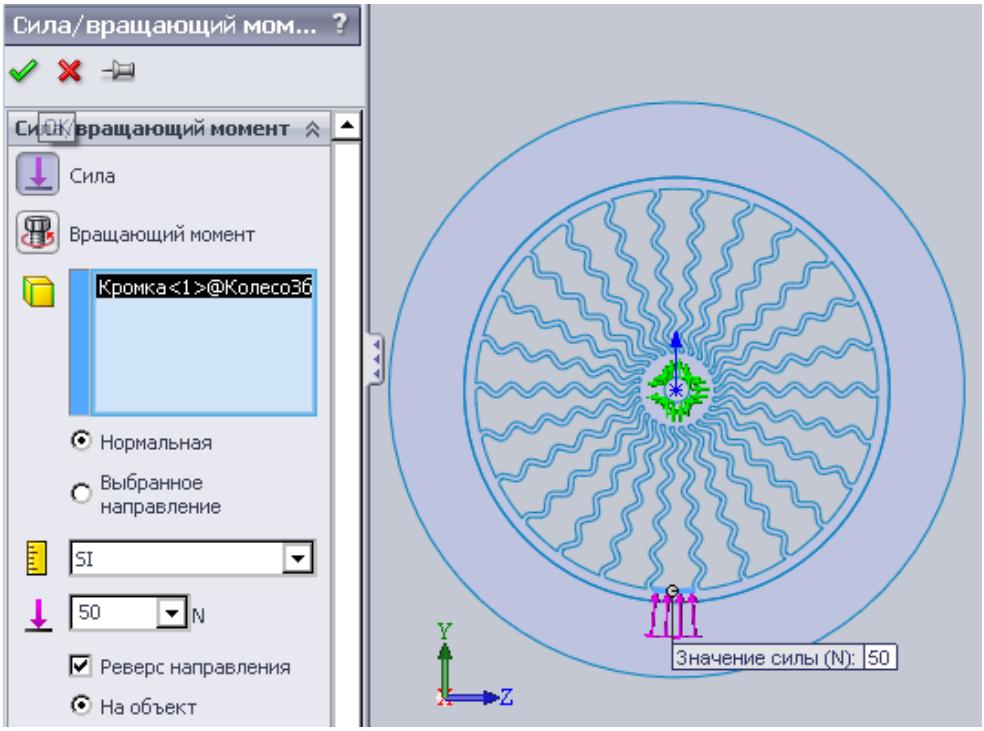



Рисунок 17 - Граничні умови і навантаження

Скінченна елементна сітка показано на рисунку 18. Допустимо зменшити розмір елементів сітки, так як це не призведе до значного уповільнення обчислень, але підвищить точність. Контакт елементів моделюється шляхом опції глобального зв'язаного контакту.

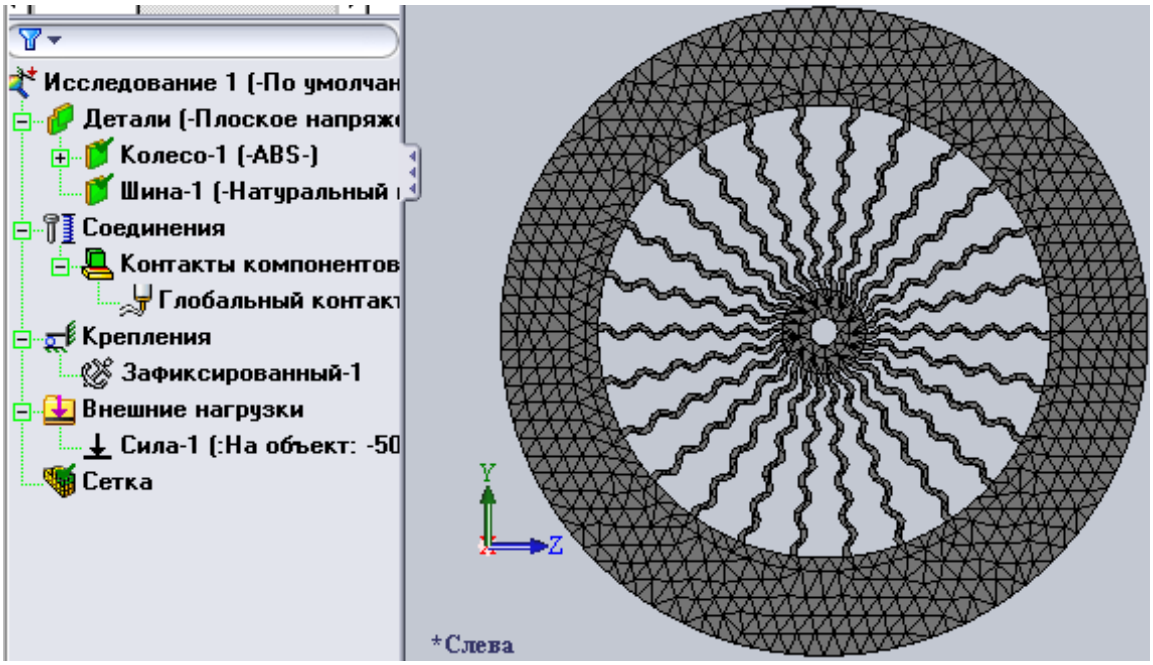



Рисунок 18 - Скінченно-елементна модель і сітка елементів

Початковий варіант конструкції: f=1, h=0,5 мм, k=26, s=10 мм. Результати симуляції початкового варіанту конструкції показані на рисунках 19, 20. Помітно, що найбільші еквівалентні напруження виникають у спиці колеса. На рисунку переміщень деформації умовно збільшено в 10 раз. Чим більші деформації, тим кращі пружні характеристики колеса. Проте слід слідкувати, щоб напруження не вийшли за межі допустимих.

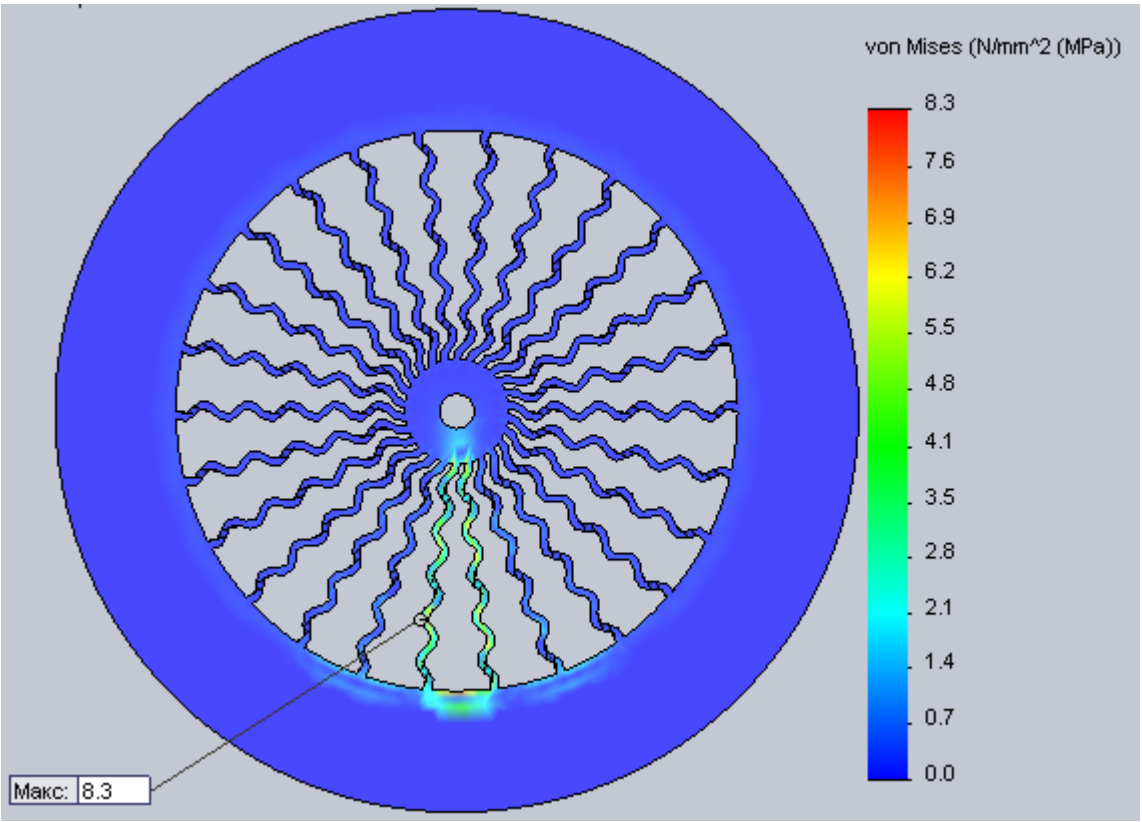



Рисунок 19 - Еквівалентні напруження (МПа) в початковому варіанті конструкції

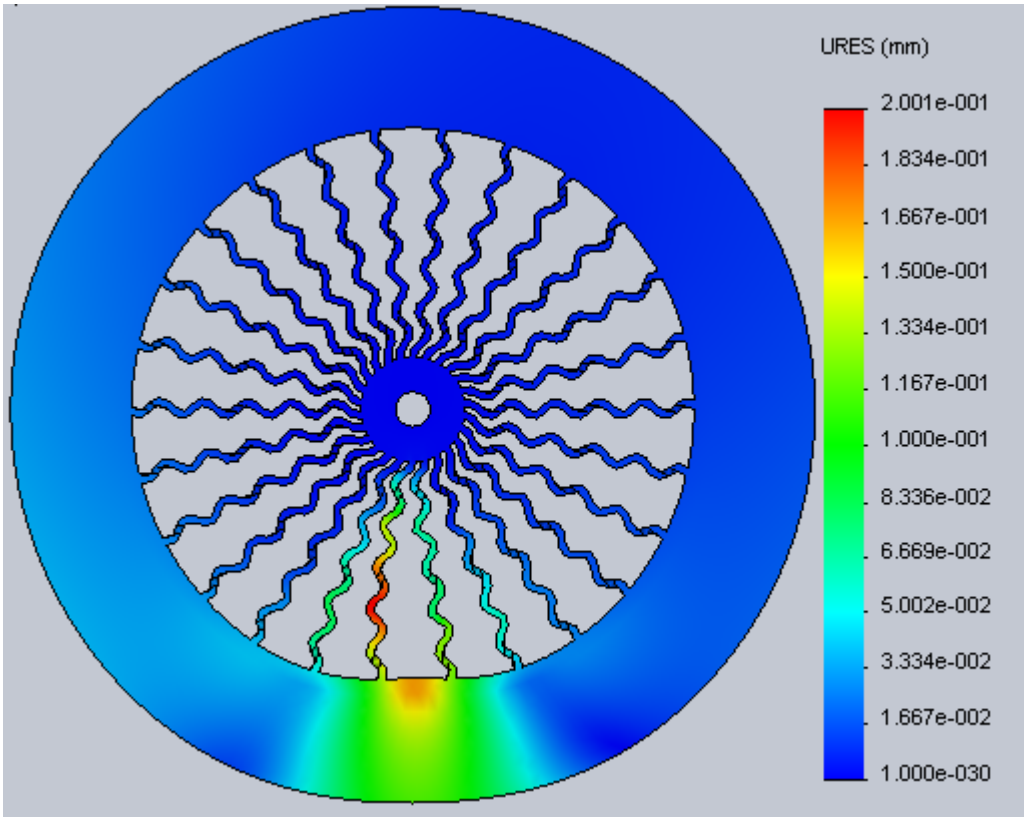


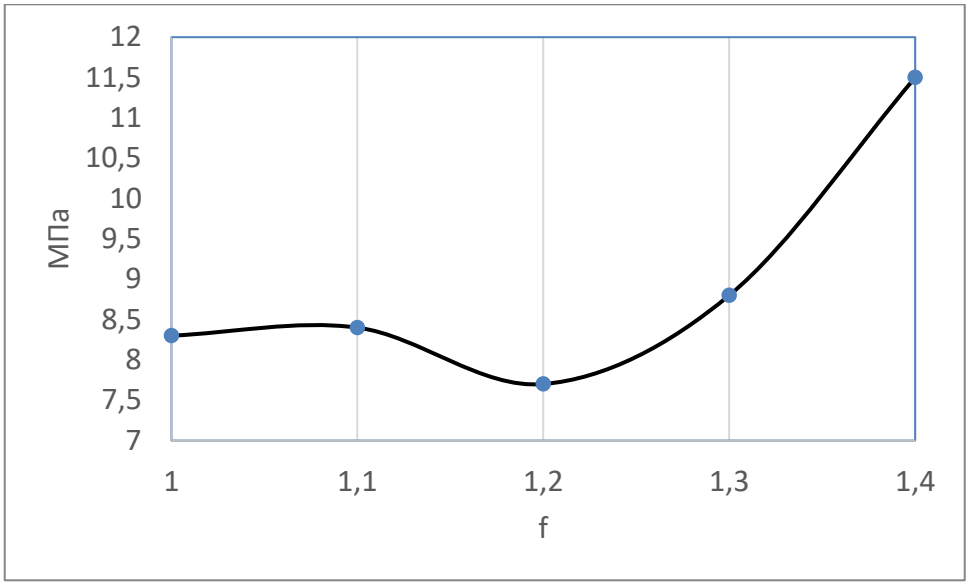

Рисунок 20 - Деформації (мм) в початковому варіанті конструкції (шкала деформації 10)

Параметрична модель дозволяє легко провести параметричний аналіз конструкції. Просто змінюємо значення параметра, виконуємо стимуляцію і переглядаємо результати. Проте деколи модель не перебудовується після зміни значення параметра. В такому разі потрібно натиснути «Редактировать определение» і оновити скінченно-елементну модель. Також після зміни значень параметрів інколи можуть появитися кілька тіл. Їх можна об'єднати в одне за допомогою операції «Объединить тела».

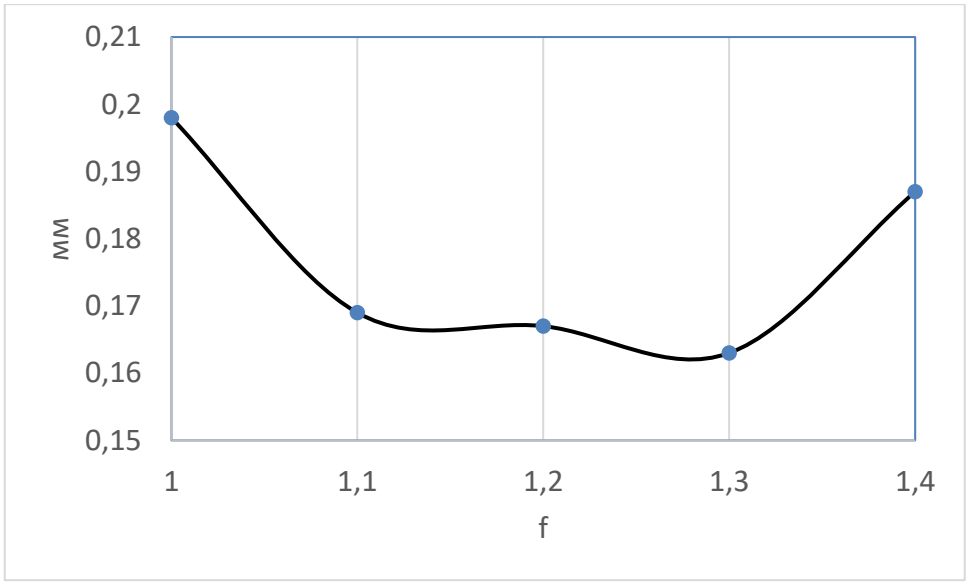
Результати параметричних досліджень напружень і деформації показані на таблицях 7-10 і рисунках 21-24. Деякі дані були апроксимовані степеневою залежністю за допомогою Microsoft Excel.

Таблиця 7 - Залежність максимальних напружень (МПа) та деформацій (мм) від параметра f функції sin(f*x), яка утворює форму спиці

| **f, мм** | **1** | 1,1 | 1,2 | 1,3 | 1,4 |
|---|---|---|---|---|---|
| Напруження, МПа | 8,3 | 8,4 | 7,7 | 8,8 | 11,5 |
| Деформації, мм | 0,198 | 0,169 | 0,167 | 0,163 | 0,187 |




а

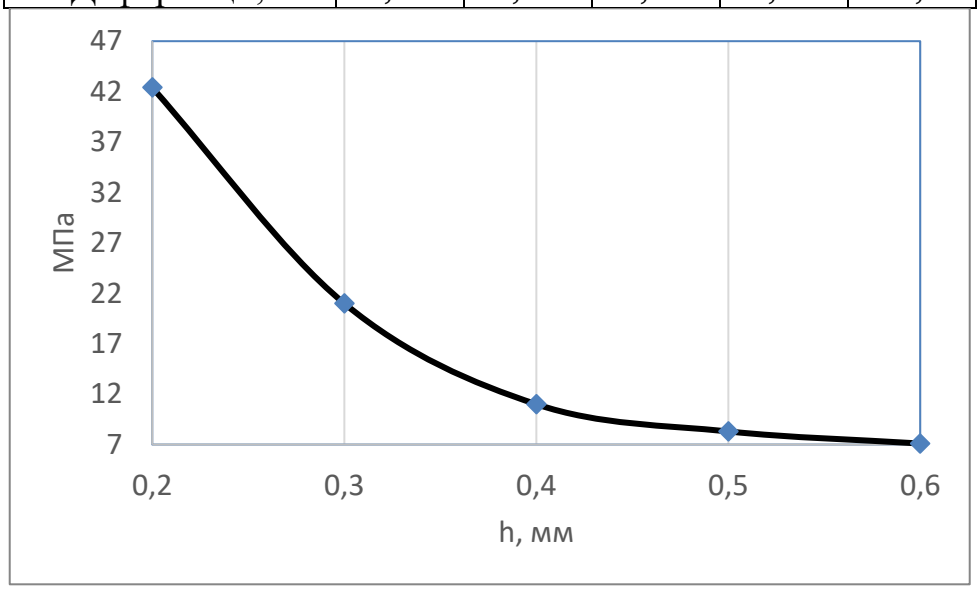



б

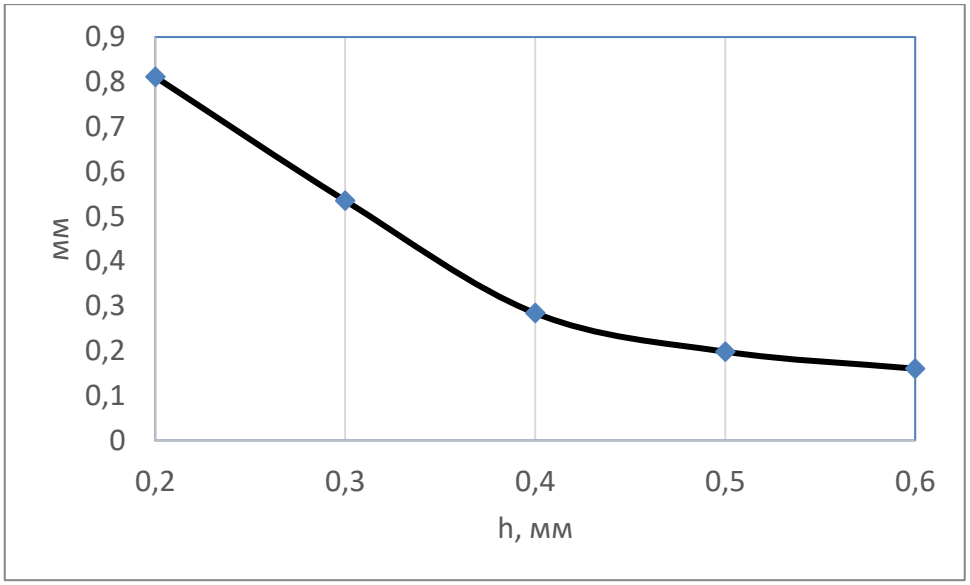
Рисунок 21 - Залежність максимальних напружень (а) та деформацій (б) від параметра f функції sin(f*x), яка утворює форму спиці

Таблиця 8 - Залежність максимальних напружень (МПа) та деформацій (мм) від товщини спиці h

| h, мм | 0,2 | 0,3 | 0,4 | **0,5** | 0,6 |
|---|---|---|---|---|---|
| Напруження, МПа | 42,4 | 21 | 11 | 8,3 | 7,1 |
| Деформації, мм | 0,811 | 0,535 | 0,284 | 0,198 | 0,16 |




а

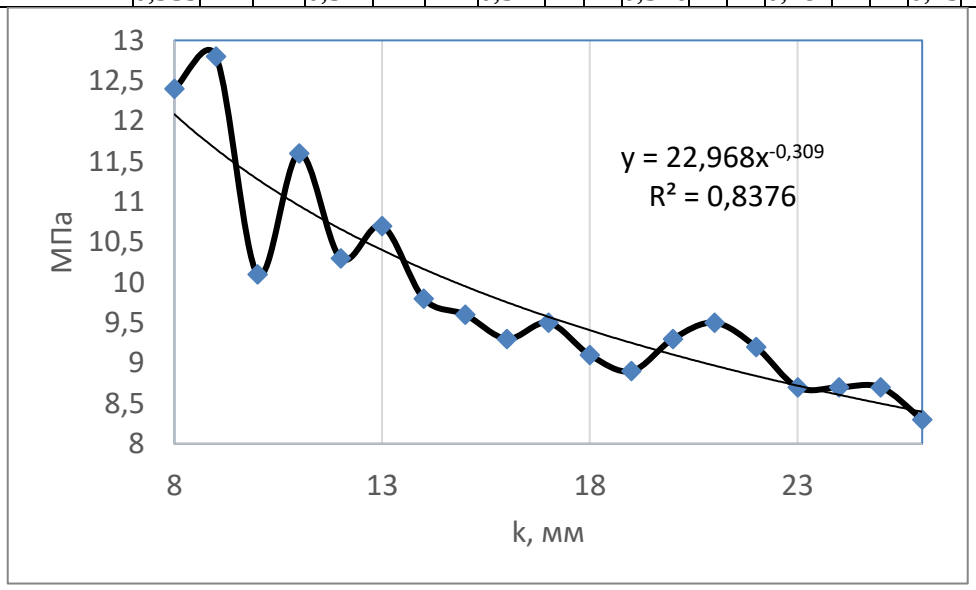



б

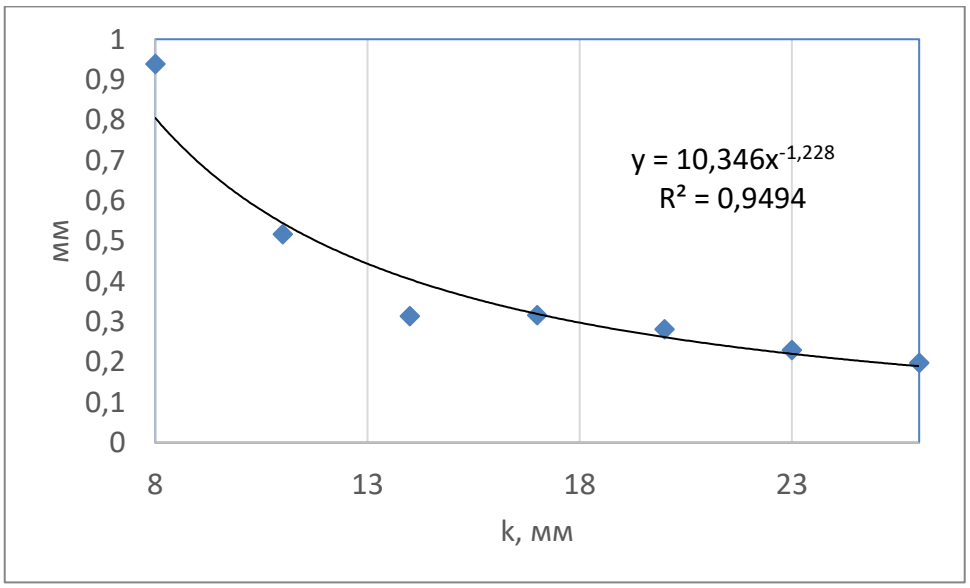
Рисунок 22 - Залежність максимальних напружень (а) та деформацій (а) від товщини спиці h

Таблиця 9 - Залежність максимальних напружень (МПа) та деформацій (мм) від кількості спиць k

| k | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | **26** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Напруження, МПа | 12,4 | 12,8 | 10,1 | 11,6 | 10,3 | 10,7 | 9,8 | 9,6 | 9,3 | 9,5 | 9,1 | 8,9 | 9,3 | 9,5 | 9,2 | 8,7 | 8,7 | 8,7 | 8,3 |
| Деформації, мм | 0,939 | | | 0,517 | | | 0,314 | | | 0,316 | | | 0,281 | | | 0,23 | | | 0,198 |




а

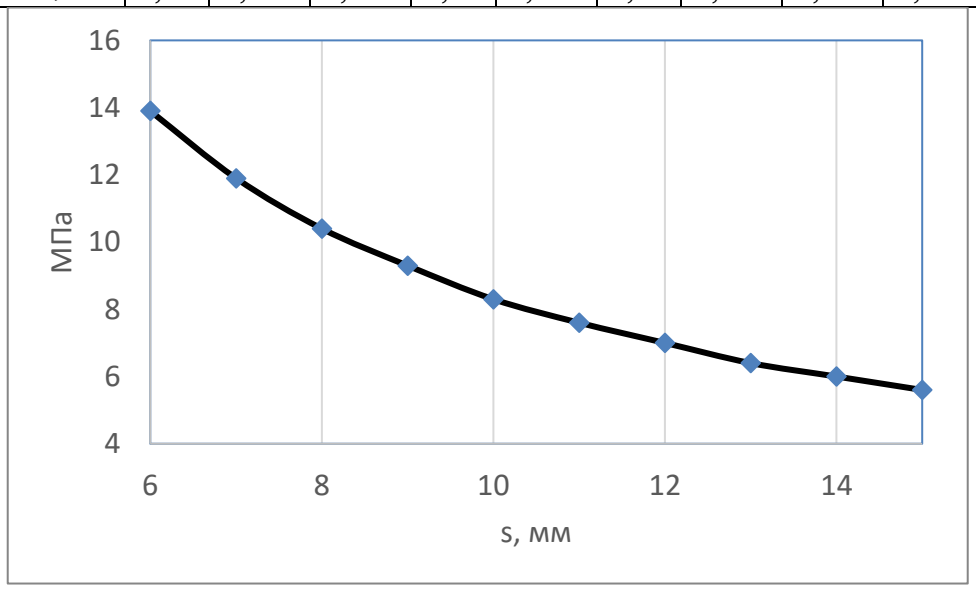



б

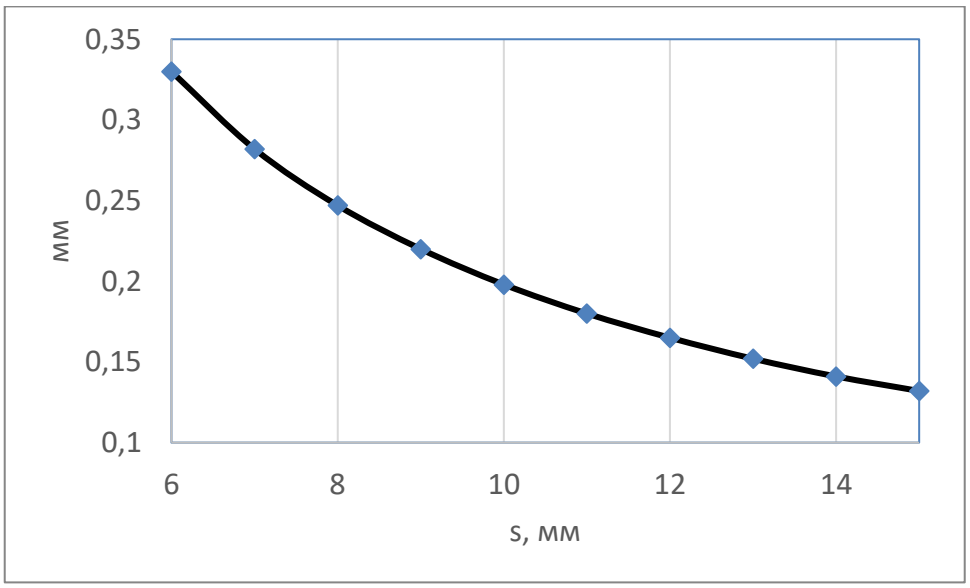
Рисунок 23 - Залежність максимальних напружень (а) та деформацій (б) від кількості спиць k

Таблиця 10 - Залежність максимальних напружень (МПа) та деформацій (мм) від ширини спиці s

| s, мм | 6 | 7 | 8 | 9 | **10** | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|
| Напруження, МПа | 13,9 | 11,9 | 10,4 | 9,3 | 8,3 | 7,6 | 7 | 6,4 | 6 | 5,6 |
| Деформації, мм | 0,33 | 0,282 | 0,247 | 0,22 | 0,198 | 0,18 | 0,165 | 0,152 | 0,141 | 0,132 |




а




б

Рисунок 24 - Залежність максимальних напружень (МПа) та деформацій (мм) від ширини спиці s

Помітно, що збільшення параметра f, який впливає на крок синусоїди, до значення 1,4 призводить до різкого збільшення напружень (рис. 21а), в той час як деформації змінюються не монотонно (рис. 21б).

Збільшення товщини спиці h призводить до зменшення напружень і деформацій (рис. 22). Збільшення кількості спиць k теж призводить до зменшення напружень і деформацій (рис. 23). Ця залежність була апроксимована степеневою функцією. Збільшення ширини спиці s призводить до зменшення напружень і деформацій (рис. 24). Ці залежності також мають степеневий вигляд.

Вибираємо наступний варіант конструкції шляхом зміни значень параметрів в напрямку збільшення деформації. Значення параметрів повинні змінюватись на один крок або на невелику величину. В іншому випадку це не буде градієнтним методом оптимізації. Отож, наступний варіант конструкції: f=1, h=0,4 мм, k=18, s=9 мм. Результати показані на рисунках 25, 26. Помітно, що деформації зросли. Але напруження менші допустимих. У разі необхідності можна повторити цей крок оптимізації і знайти конструкцію з заданою пружністю.

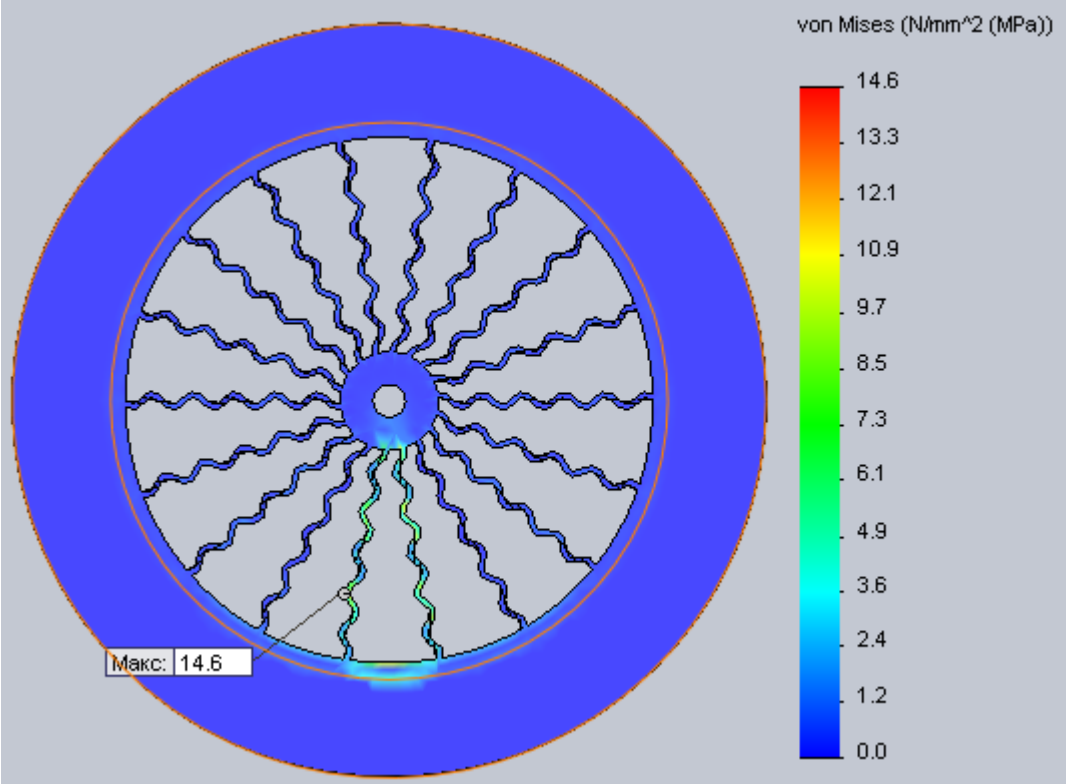



Рисунок 25 - Еквівалентні напруження (МПа) на кроці 2 оптимізації конструкції

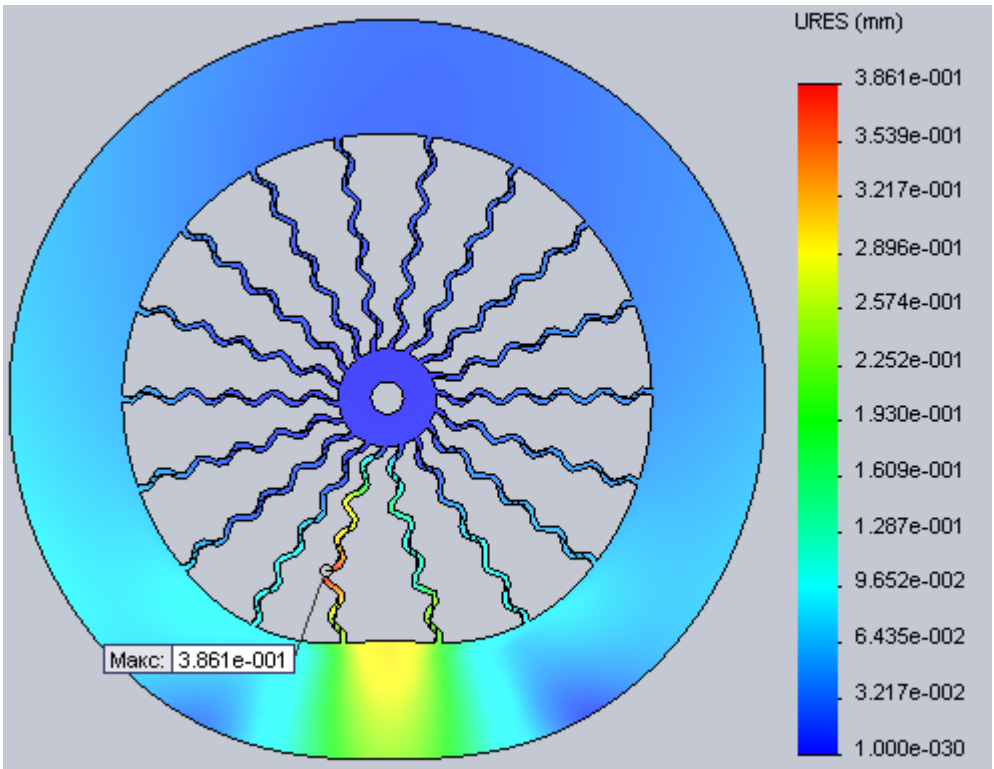



Рисунок 26 - Деформації (мм) на кроці 2 оптимізації конструкції (шкала деформації 10)

## Висновки

1. Реалізовано схему підключення крокових двигунів до плати Arduino за допомогою драйверів A4988. Створено програму мовою Python для програмування цього робота з клавіатури. Роботом керує програма, яка виконується на персональному комп'ютері та за допомогою послідовного порту та протоколу Firmata передає дані на мікроконтролер робота.
2. За допомогою САПР SOLIDWORKS спроектовано пристрій для програмного перемикання механічних передач, що дозволяє підключити додаткові механізми (до 13 ступенів вільності) та маніпулятори і розширити можливості робота.
3. За допомогою САПР SOLIDWORKS спроектовано мобільний універсальний робот, який має плату Arduino Uno, два крокові двигуни з драйверами, маніпулятор та пристрій для програмного перемикання механічних передач. Робот має мінімальну собівартість, вагу, максимум стандартних деталей, мінімум крокових двигунів та максимум функцій, може бути виготовлений на недорогих фрезерних верстатах з ЧПК та 3D-принтерах і запрограмований за допомогою популярної мови програмування Python.
4. За допомогою SOLIDWORKS Simulation виконано крок оптимізації конструкції колеса робота за критерієм мінімальної жорсткості із використанням обмежень на максимальні еквівалентні напруження.
5. Розроблений робот може бути використаний в майбутніх навчальних проектах з використанням сенсорів, методів машинного зору та машинного навчання.

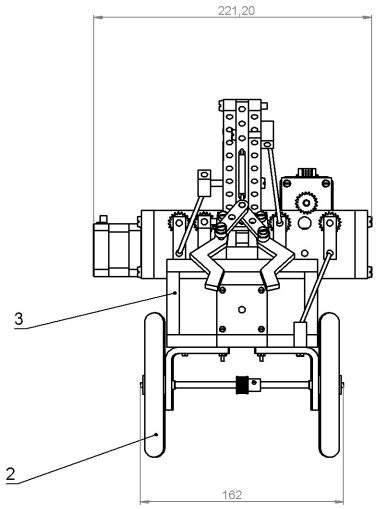
## Література

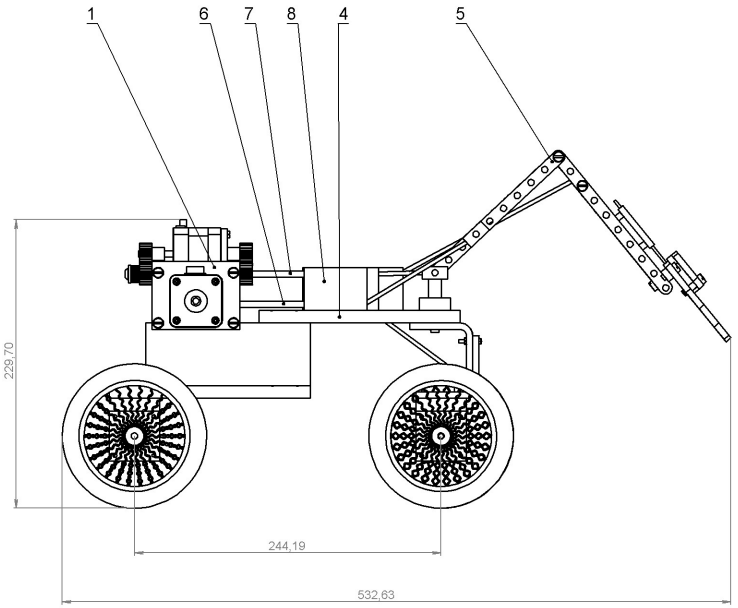

1. Андре П. Конструирование роботов. Перевод с французского Д.М.Далечиной, М.С.Фанченко и канд. техн. наук В.И.Чебуркова, Под редакцией д-ра техн. наук А.М.Долгова. Москва : Мир, 1986. 360 с.
2. Воробьев Е.И., Бабич А.В., Жуков К.П., Попов С.А., Семин Ю.И. Механика промышленных роботов. Книга 3. Основы конструирования. Москва : Высшая школа, 1989. 383 с.
3. Бурдаков С. Ф. и др. Проектирование манипуляторов промышленных роботов и роботизированных комплексов. Москва : Высш. шк., 1986. 264 с.
4. Веселков Р. С. и др. Детали и механизмы роботов. Основы расчета, конструирования и технологии производства. Киев. Выща школа, 1990. 343 с.
5. Отений Я.Н., Ольштынский П.В. Выбор и расчет захватных устройств промышленных роботов. Учебное пособие. Волгоград : ВолгГТУ, 2000. 64с.
6. Бишоп О. Настольная книга разработчика роботов. Киев : МК-Пресс, 2010. 400 с.
7. Драйвер шагового двигателя A4988. URL: https://3d-diy.ru/wiki/arduino-moduli/drajver-shagovogo-dvigatelya-a4988/ (Дата звернення: 1.06.22 р.)
8. Шаговый двигатель NEMA 17HS4401 42bygh 1.7A. URL: https://220v.biz/shagovyy-dvigatel-nema-17hs4401-42bygh-17a/ (Дата звернення: 1.06.22 р.)
9. Модуль драйвера двигунів на L293D. URL: https://arduino.ua/prod2608-modyl-draivera-motorov-na-l293d (Дата звернення: 1.06.22 р.)
10. Обзор драйвера мотора на L298N. URL: https://robotchip.ru/obzor-drayvera-motora-na-l298n/ (Дата звернення: 1.06.22 р.)

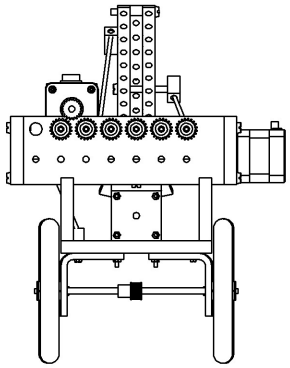
11. A4988 DMOS microstepping driver with translator. URL: https://www.pololu.com/file/download/a4988_DMOS_microstepping_driver_with_translator.pdf?file_id=0J450 (Дата звернення: 1.06.22 р.)
12. Настройка тока драйвера A4988. URL: https://geekmatic.in.ua/a4988_nastroyka_toka (Дата звернення: 1.06.22 р.)
13. Драйвер шагового двигателя A4988. URL: https://3d-diy.ru/wiki/arduino-moduli/drajver-shagovogo-dvigatelya-a4988/ (Дата звернення: 1.06.22 р.)
14. Arduino UNO R3. URL: https://docs.arduino.cc/hardware/uno-rev3 (last access 30.05.2022).
15. Монк С., Шерц П. Электроника. Теория и практика. 4-е изд. Пер. с англ. Санкт-Петербург : БХВ-Петербург, 2018. 1168 с.
16. Arduino for A4988 Pololu Stepper Motor Driver code! URL: https://forum.arduino.cc/t/arduino-for-a4988-pololu-stepper-motor-driver-code/130985 (last access: 1.06.22 р.)
17. Обзор драйвера шагового двигателя A4988. URL: https://robotchip.ru/obzor-drayvera-shagovogo-dvigatelya-a4988/ (Дата звернення: 1.06.22 р.)
18. Bräunl T. Embedded Robotics : From Mobile Robots to Autonomous Vehicles with Raspberry Pi and Arduino. 4-ed. Springer, 2022. 510 p.
19. Smythe R. J. Advanced Arduino Techniques in Science : Refine Your Skills and Projects with PCs or Python-Tkinter. New York : Apress, 2021. 306 p.
20. Cicolani J. Beginning Robotics with Raspberry Pi and Arduino : Using Python and OpenCV. New York : Apress, 2021. 369 p.
21. A small fork of AccelStepper v1.3 with AF_motor (Adafruit motor shield) support! URL: https://github.com/adafruit/AccelStepper (last access: 1.06.22 р.)
22. Documentation of the Firmata protocol. URL : https://github.com/firmata/protocol (last access 30.05.2022).
23. Python serial port access library. URL : https://github.com/pyserial/pyserial (last access 30.05.2022).

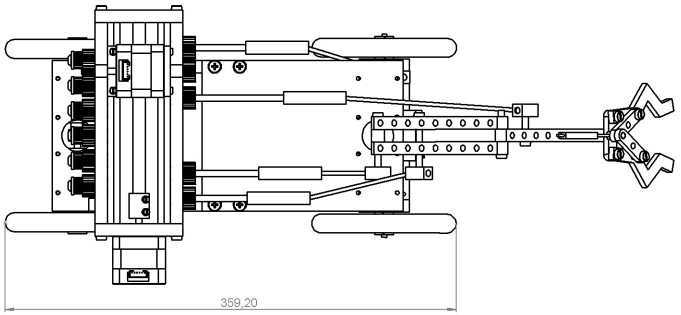
24. Python interface for the Firmata protocol. URL: https://github.com/tino/pyFirmata (last access: 30.05.2022).
25. Копей В. Мова програмування Python для інженерів і науковців: Навчальний посібник. Івано-Франківськ : ІФНТУНГ, 2019. 274 с.
26. 3D-печать для "чайников" или "что такое 3D-принтер?". URL: https://3dtoday.ru/wiki/3dprint_basics (Дата звернення: 1.06.22 р.)
27. Копей Б.В., Копей В.Б. Використання методу скінченних елементів та тривимірного комп’ютерного моделювання для конструювання та оптимізації параметрів нафтогазового обладнання: Навчальний посібник. Івано-Франківськ : Факел, 2008. 117 с.

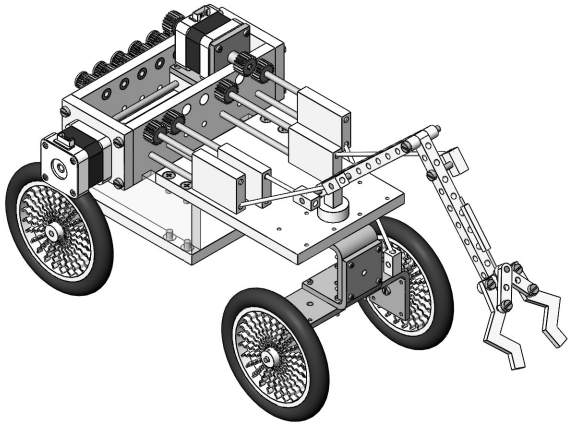
СБ





| Поз. | Назва | Кількість |
|---|---|---|
| 1 | Привід в зборі | 1 |
| 2 | Колесо в зборі | 4 |
| 3 | Задня вісь в зборі | 1 |
| 4 | Передня вісь в зборі | 1 |
| 5 | Маніпулятор в зборі | 1 |
| 6 | Напрямна | 4 |
| 7 | Гвинт | 4 |
| 8 | Гайка | 4 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | БР-096.00.01 СК | |
| Змін. | Лист. | № докум. | Підп. | Дата | Мобільний робот | Лит. / Масса / Масштаб |
| Розраб. | | Мільник | | | | 2.75 / 1:2 |
| Пров. | | Копей | | | | Лист / Листов |
| Т. контр. | | Копей | | | | |
| Н. контр. | | Копей | | | ІФНТУНГ | |
| Утв. | | Панчук | | | | |

Копировал

Формат А1

Перв. примен.

Справ. №

Подп. и дата

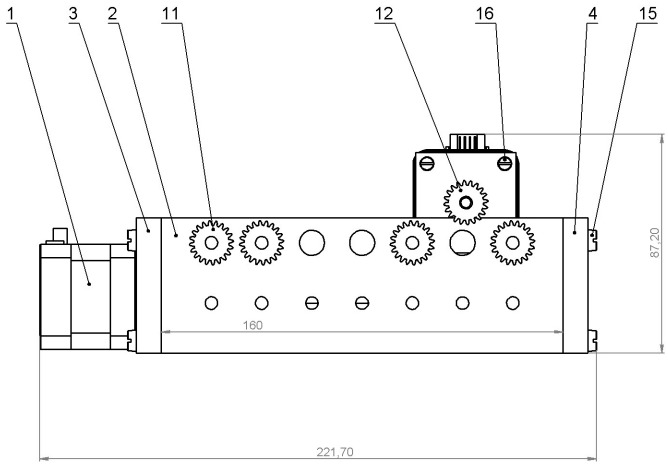
Инв. № дубл.

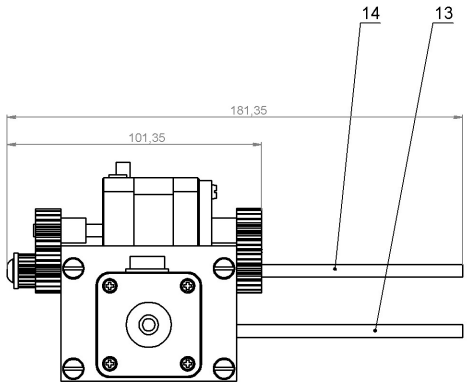
Взам. инв. №

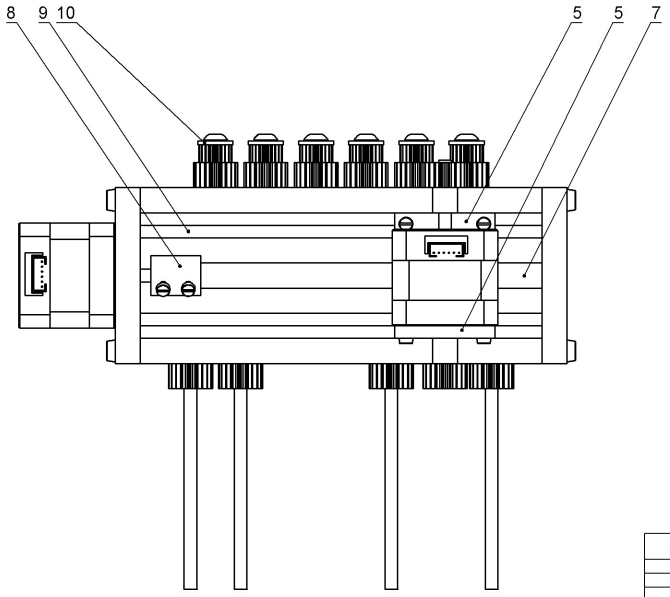
Подп. и дата

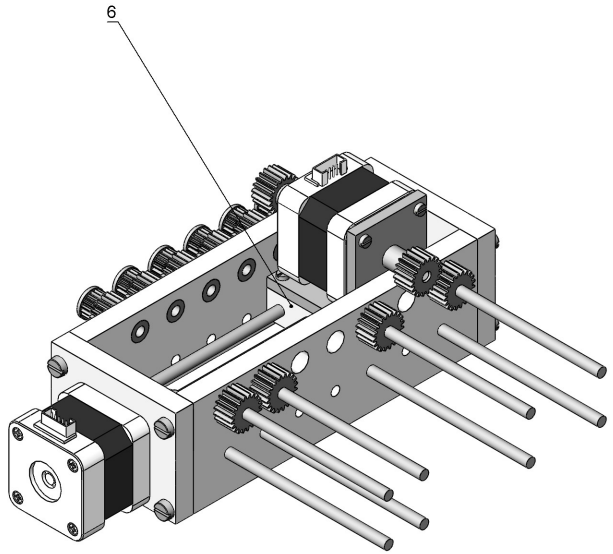
Инв. № подл.

СБ




| Поз. | Назва | Кількість |
|---|---|---|
| 1 | Крокový двигун | 2 |
| 2 | Корпус | 1 |
| 3 | Кришка | 1 |
| 4 | Кришка | 1 |
| 5 | Кутник | 2 |
| 6 | Гайка | 1 |
| 7 | Гвинт | 1 |
| 8 | Муфта | 1 |
| 9 | Напрямна | 2 |
| 10 | Шестерня | 6 |
| 11 | Колесо | 10 |
| 12 | Шестерня | 2 |
| 13 | Напрямна | 4 |
| 14 | Гвинт | 4 |
| 15 | Гвинт | 8 |
| 16 | Гвинт | 8 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | БР-096.00.02 СК | |
| Изм. | Лист. | № докум. | Подп. | Дата | Пристрій для перемикання передач | Лит. / Масса / Масштаб |
| Разраб. | | Мельник | | | | 0.95 |
| Пров. | | Копей | | | | Лист / Листов |
| Т. контр. | | Копей | | | | |
| Н. контр. | | Копей | | | ІФНТУНГ | |
| Утв. | | Панчук | | | | |

Копировал

Формат А1

Перв. примен.

Справ. №

Подп. и дата

Инв. № дубл.

Взам. инв. №

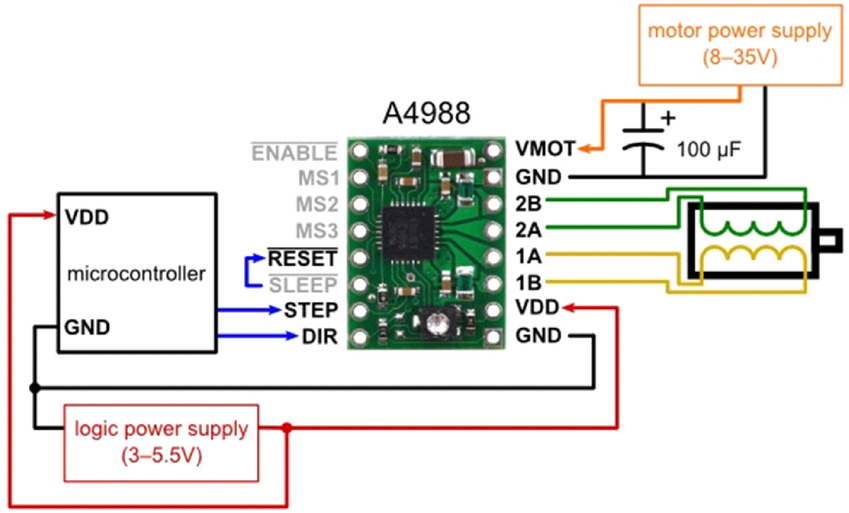
Подп. и дата

Инв. № подл.

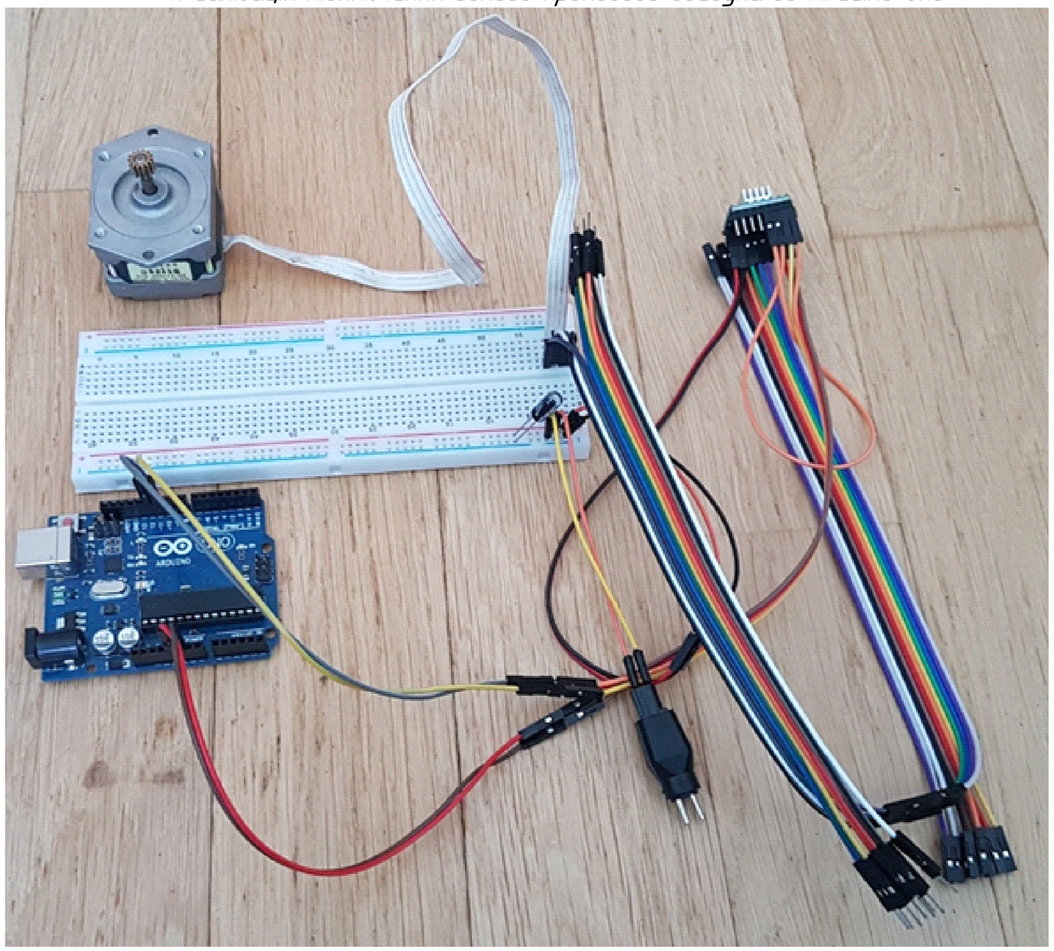
БР-096.00.003

## Крокoвий двигун NEMA 17HS4401


## Схема підключення крокового двигуна




## Реалізація підключення одного крокового двигуна до Arduino Uno



## Python-модуль для програмування робота з двома крокoвими двигунами

```
# -*- coding: utf-8 -*-  # кодування символів файлу
from pyfirmata import Arduino, util
time=util.time
board = Arduino('COM19', baudrate=57600) # плата Arduino на порті COM19

s1=board.get_pin('d:12:o') # пін кроку двигуна 1
d1=board.get_pin('d:11:o') # пін напрямку двигуна 1
s2=board.get_pin('d:10:o') # пін кроку двигуна 2
d2=board.get_pin('d:9:o') # пін напрямку двигуна 2

class Motor:
    """Клас крокових двигунів"""
    def __init__(self, pin_step, pin_dire): # конструктор класу
        "pin_step – пін кроку, pin_dire – пін напрямку"
        self.pin_step=pin_step
        self.pin_dire=pin_dire
        self.value=0 # поточна кількість кроків
        self.dire=1 # напрямок 1 або -1

    def step(self):
        "Поворот на один крок"
        self.pin_step.write(1) # вивести логічну 1
        time.sleep(0.001)
        self.pin_step.write(0) # вивести логічне 0
        time.sleep(0.001)
        self.value+=self.dire  # збільшити/зменшити значення

    def move(self, steps=1):
        "Поворот в напрямку self.dire на steps кроків"
        if self.dire==1: # якщо напрямок 1
            self.pin_dire.write(1) # установити напрямок 1
        else: # інакше
            self.pin_dire.write(0) # установити напрямок 0
        for i in range(steps): # повернути на steps кроків
            self.step()

    def moveTo(self, value):
        "Поворот з поточного положення в положення value"
        steps=value-self.value # визначити кількість кроків
        self.dire = -1 if steps<0 else 1 # визначити напрямок
        self.move(abs(steps)) # повернути на steps кроків


m1=Motor(s1, d1) # двигун 1
m1.value=0 # введіть початкове значення!
m2=Motor(s2, d2) # двигун 2
m2.value=0 # введіть початкове значення!

def key_handler(event=None):
    "Подія натиску клавіші"
    m=None # двигун
    if event: # якщо клавіша натиснута
        k=event.keycode # код клавіші
        print "keycode=",k
        if   k==39: m,d = m1,-1 # натиск <вправо>
        elif k==37: m,d = m1,1 # натиск <вліво>
        elif k==40: m,d = m2,-1 # натиск <вниз>
        elif k==38: m,d = m2,1 # натиск <вверх>
        elif k==27: print P; board.exit(); r.destroy(); # вихід <Esc>
        elif k==80: addPoint() # додати точку <P>
        else: m=None
        if m: # якщо двигун обрано
            m.dire=d # напрямок
            m.move(10) # повернути на 10 кроків
            print m.value # вивести значення

P=[] # опорні точки
def addPoint():
    "Додає опорну точку в список"
    p=m1.value, m2.value # точка як кортеж
    P.append(p) # додати точку
    print p

if __name__=="__main__": # якщо модуль виконується, а не імпортується
    import Tkinter as tk
    r = tk.Tk() # створити вікно
    r.bind("<Key>", key_handler) # повідомити функцію з подією
    r.mainloop() # головний цикл GUI
```

## Arduino-скетч для керування КД

```
int sm = 12; // перепад сигналу 1-0 призводить до повороту на крок
int d = 11; // напрямок – 0 – в одну сторону, 1 – в другу
void setup() // функція виконується один раз на початку
{
pinMode(sm, OUTPUT);  // режим піна – виведення
pinMode(d, OUTPUT);  // режим піна – виведення
digitalWrite(d, HIGH); // напрямок обертання
}
void loop() // функція постійно повторюється
{
// виконати один крок
digitalWrite(sm, HIGH); // подати логічну 1
delay(10); // затримка на 10 мс
digitalWrite(sm, LOW); // подати логічне 0
delay(10);
}
```

## Python-модуль для виконання програми робота

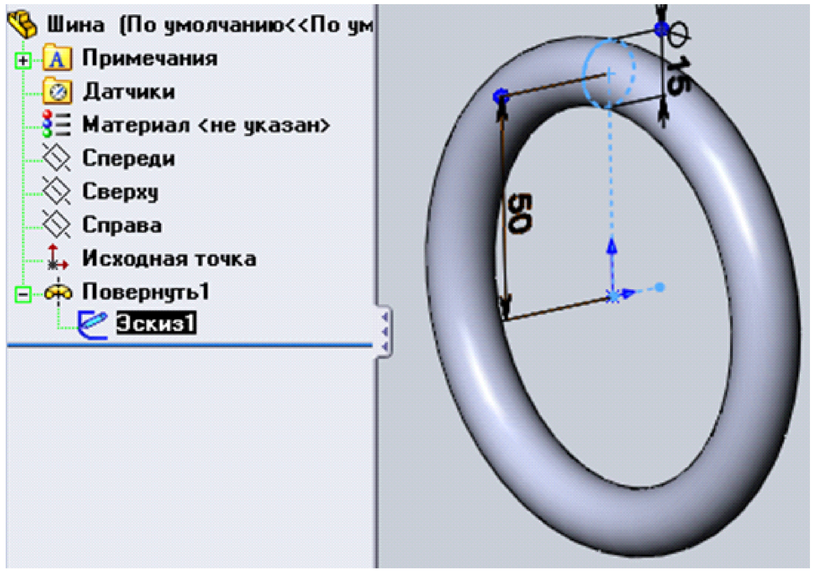
```
# -*- coding: utf-8 -*-
from stepper2 import  m1, m2, board # імпортувати двигуни і плату

def runProgram(prog):
    "Виконує програму prog для робота з 2-ма крокoвими двигунами"
    i=1 # номер кадру (точки)
    for p1,p2 in prog: # для кожної точки
        print i, [p1,p2] # вивести
        m1.moveTo(p1) # рухатись двигуну m1 в p1
        m2.moveTo(p2) # рухатись двигуну m2 в p2
        i+=1

# програма для робота як список точок:
prog=[(100, 0),(100, 100), (100, 200), (100, 800), (100, 900), (200, 0), (100, 800),
(800, 700), (800, 600), (800, 700), (800, 600), (800, 700), (800, 700), (800, 800),
(800, 700), (100, 700), (100, 800), (100, 600), (100, 500), (200, 600), (900, 500)]

runProgram(prog) # виконати програму
board.exit() # завершити
```

| Изм. | Лист. | № докум. | Подп. | Дата | БР-096.00.003 | | | |
|---|---|---|---|---|---|---|---|---|
| Разраб. | | Мельник | | | Схема підключення і спосіб програмування | Лит. | Масса | Масштаб |
| Пров. | | Копей | | | | | | 1:1 |
| Т.контр. | | Копей | | | | Лист | Листов 1 | |
| Н.контр. | | Копей | | | | ІФНТУНГ | | |
| Утв. | | Панчук | | | | | | |

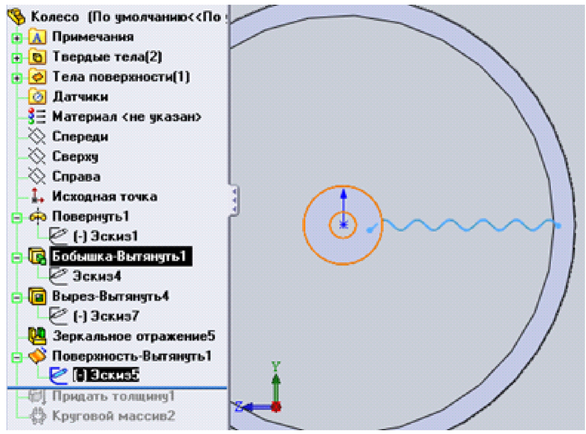
Копировал  Формат А1

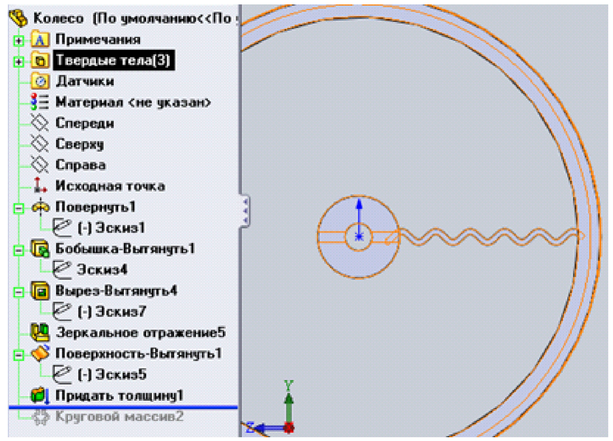
Перв. примен. | Справ. № | Подп. и дата | Инв. № дубл. | Взам. инв. № | Подп. и дата | Инв. № подл.

*Параметрична модель гумової шини*

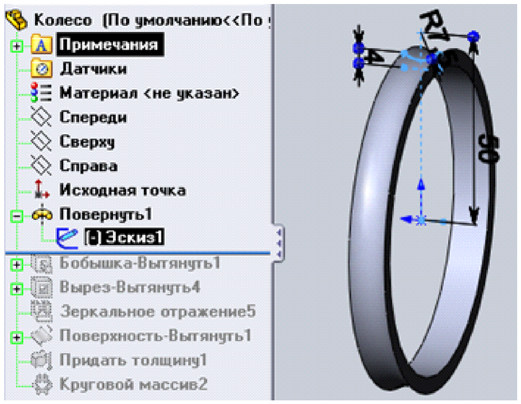


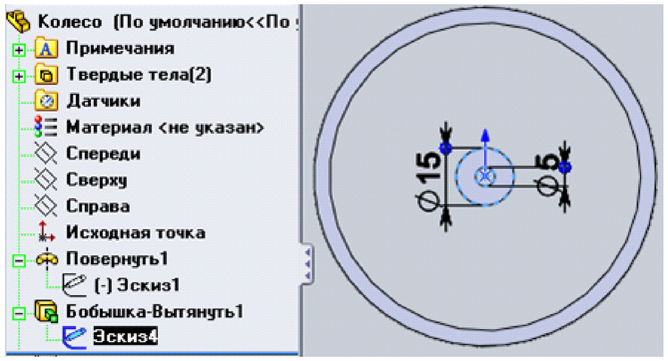

*Крива для побудови спиці (а) та спиця (б)*



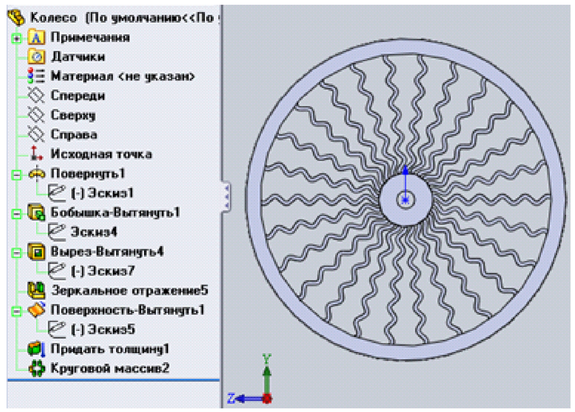



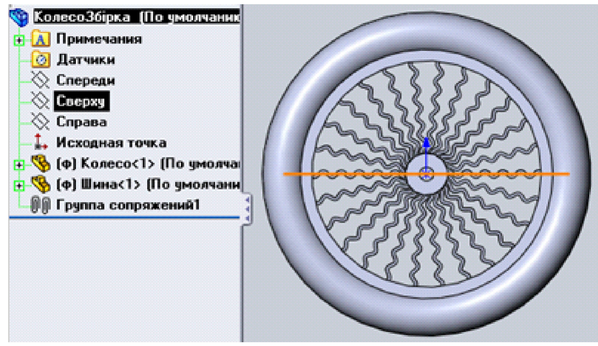

*Обід колеса (а) і ступиця (б)*



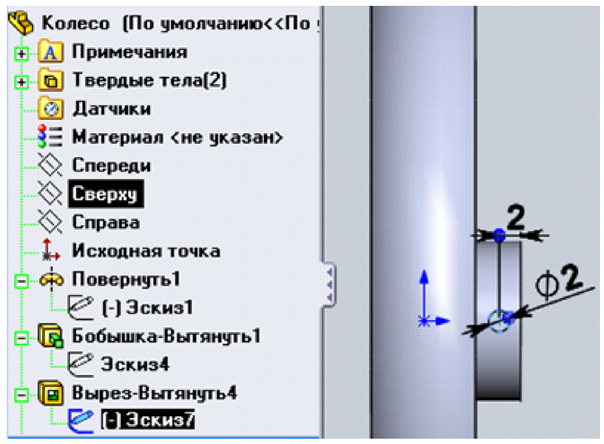



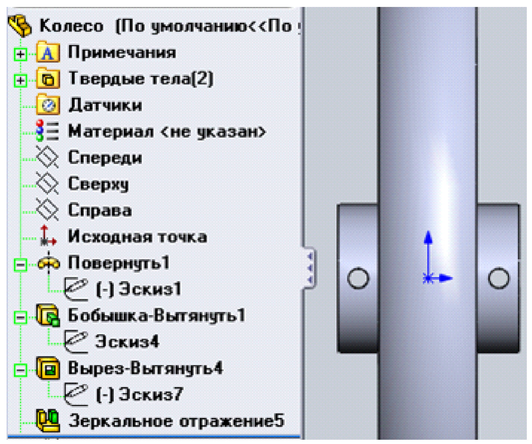

*Масив спиць (а) і колесо в зборі (б)*




*Отвір в ступиці (а) і дзеркальна копія тіла (б)*

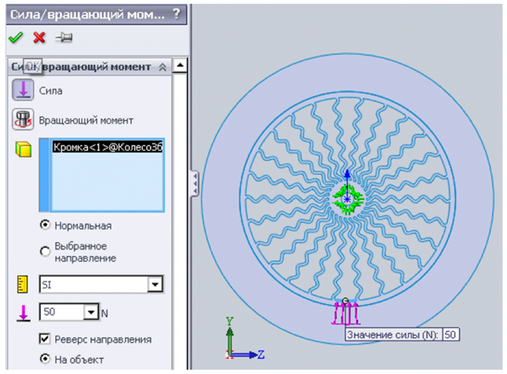


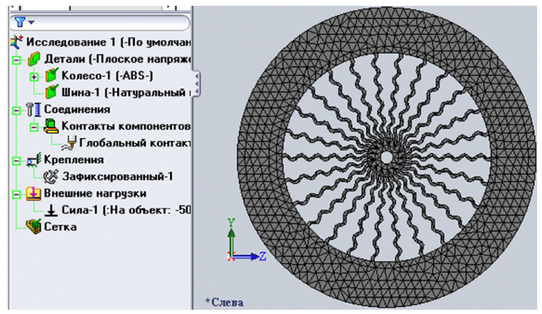

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | БР-096.00.04 | | |
| Изм. | Лист | № докум. | Подп. | Дата | | Параметрична модель колеса | Лит. | Масса | Масштаб |
| Разраб. | | Мельник | | | | | | | 1:1 |
| Пров. | | Копей | | | | | Лист | Листов 1 |
| Т.контр. | | Копей | | | | | | |
| Н.контр. | | Копей | | | | | ІФНТУНГ | |
| Утв. | | Панчук | | | | | | |

Копировал Формат А1

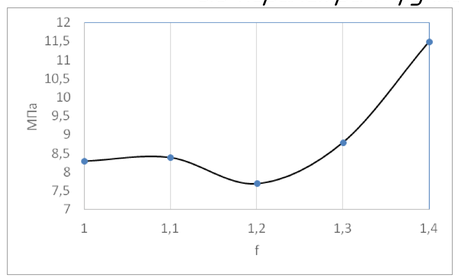
## Граничні умови і навантаження

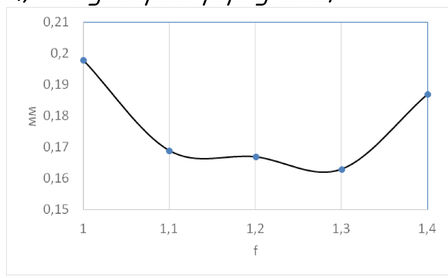



## Скінченно-елементна модель і сітка елементів

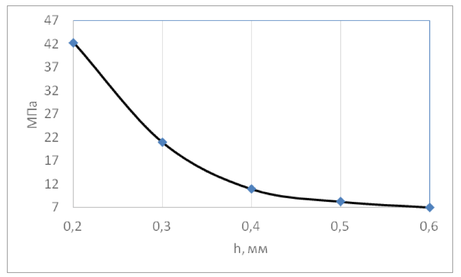


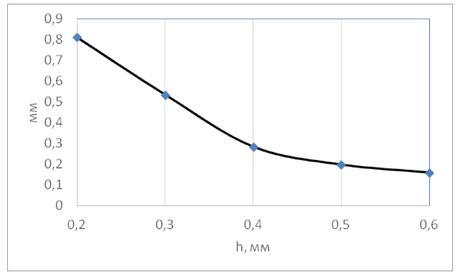

## Залежність максимальних напружень (а) та деформацій (б) від параметра f функції sin(f*x), яка утворює форму спиці



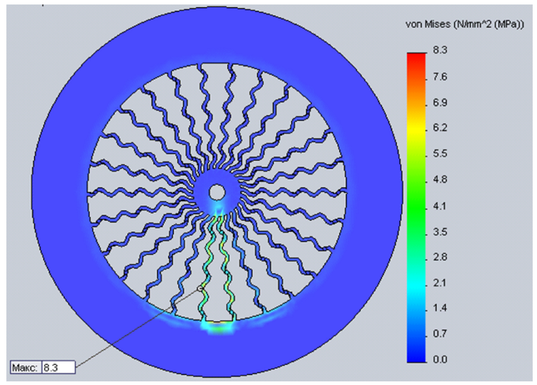




## Залежність максимальних напружень (а) та деформацій (а) від товщини спиці h

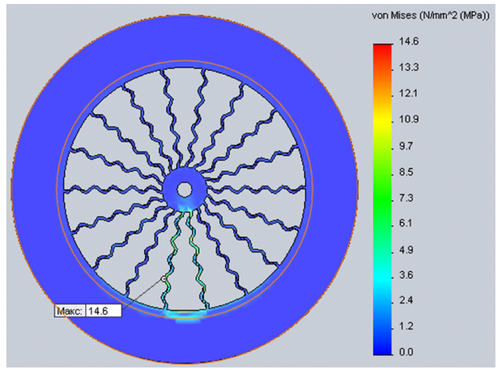


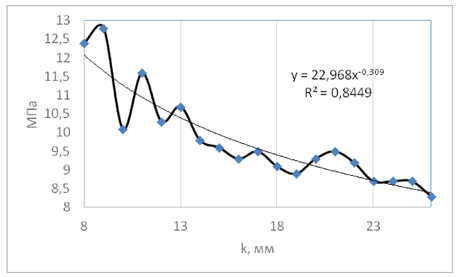

## Еквівалентні напруження (МПа) в початковому варіанті конструкції

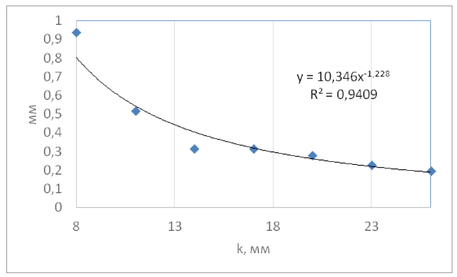



## Еквівалентні напруження (МПа) на кроці 2 оптимізації конструкції

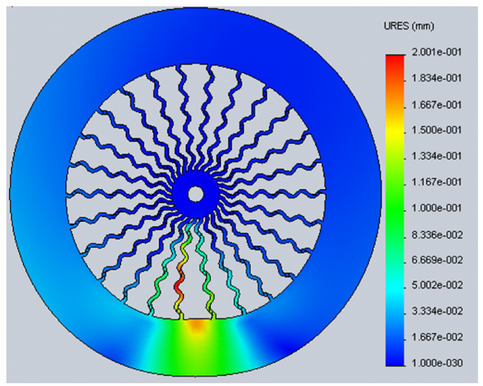



## Залежність максимальних напружень (а) та деформацій (б) від кількості спиць k

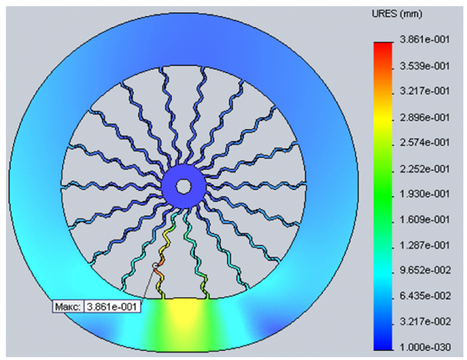


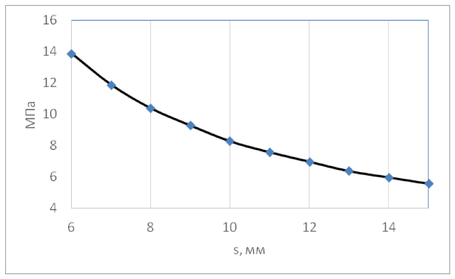

## Деформації (мм) в початковому варіанті конструкції (шкала деформації 10)

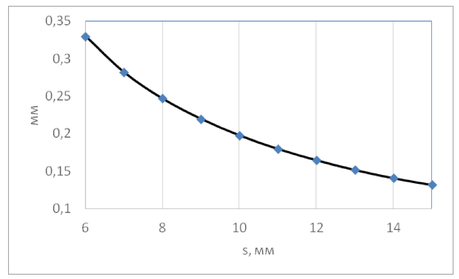



## Деформації (мм) на кроці 2 оптимізації конструкції (шкала деформації 10)




## Залежність максимальних напружень (МПа) та деформацій (мм) від ширини спиці s




| | | | | | | |
|---|---|---|---|---|---|---|
| Изм. | Лист. | № докум. | Подп. | Дата | БР-096.00.05 | |
| Разраб. | | Мельник | | | Оптимізація конструкції колеса | Лит. / Масса / Масштаб: 1:1 |
| Пров. | | Копей | | | | Лист / Листов: 1 |
| Т.контр. | | Копей | | | | |
| Н.контр. | | Копей | | | ІФНТУНГ | |
| Утв. | | Панчук | | | | |

Копировал — Формат А1

Перв. примен. | Справ. № | Подп. и дата | Инв. № дубл. | Взам. инв. № | Подп. и дата | Инв. № подл.