МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ ПОЛІТЕХНІЧНИЙ УНІВЕРСИТЕТ
Кафедра підйомно-транспортного та робототехнічного обладнання

МЕТОДИЧНІ ВКАЗІВКИ ДО РОЗРАХУНКОВО-ГРАФІЧНОЇ РОБОТИ

з дисципліні "Комп'ютерні методи розрахунку роботів"

Перший (бакалаврський) рівень вищої освіти
Спеціальність – 131 ПРИКЛАДНА МЕХАНІКА
Освітня програма – *Інженерія логістичних систем*, *Мехатроніка та промислові роботи*

ОДЕСА: ОНПУ, 2021

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ ПОЛІТЕХНІЧНИЙ УНІВЕРСИТЕТ
Кафедра підйомно-транспортного та робототехнічного обладнання

Михайлов Євген Павлович

МЕТОДИЧНІ ВКАЗІВКИ ДО РОЗРАХУНКОВО-ГРАФІЧНОЇ РОБОТИ

з дисципліні "Комп'ютерні методи розрахунку роботів"

Перший (бакалаврський) рівень вищої освіти
Спеціальність – 131 ПРИКЛАДНА МЕХАНІКА
Освітня програма – *Інженерія логістичних систем*, *Мехатроніка та промислові роботи*

Затверджено
на засіданні кафедри
підйомно-транспортного і
робототехнічного обладнання
Протокол № 8 від 8 лютого 2021 р.

ОДЕСА: ОНПУ, 2021

Методичні вказівки до виконання розрахунково-графічної роботи з дисципліни «Комп'ютерні методи розрахунку роботів» для здобувачів першого (бакалаврського) рівня вищої освіти, спеціальність: 131 - Прикладна механіка, освітні програми: Мехатроніка та промислові роботи, Інженерія логістичних систем / Укл.: Михайлов Є. П. – Одеса: ОНПУ, 2021. - 29 с.

Укладач: Михайлов Є. П.

Зміст

ВСТУП

Розрахунково-графічна робота (РГР) є індивідуальним завданням, яке має на меті не лише поглиблення, узагальнення і закріплення знань студентів з навчальної дисципліни, а й застосування їх при вирішенні конкретного завдання і вироблення вміння самостійно працювати з навчальною літературою, використовуючи сучасні інформаційні засоби.

Обсяг годин індивідуальної роботи студентів для виконання розрахунково-графічної роботи –15 год СРС (0,5 кредиту) за семестр.

Кожне завдання складається з індивідуальних задач середнього рівня складності. Розрахункові роботи виконуються у відповідності до наданих рекомендацій.

Мета: опанування студентами сучасних комп'ютерних методів розрахунку параметрів керування промисловими стаціонарними та мобільними роботами.

Задачі виконання РГР: розв'язання конкретних задач комп'ютерних методів розрахунку є необхідною практичною основою при вивченні курсу комп'ютерні методи розрахунку роботів, також це сприяє розвитку самостійності студента, вихованню навичок самостійного здобуття знань, виробляє уміння застосовувати здобуті знання на практиці.

Етапи виконання:

- видача завдань (1-2 тиждень);
- самостійна робота студентів (3-11 тиждень);
- захист РГР (12 тиждень).

Тема РГР: «Комп'ютерні методи розрахунку щодо вирішування задач переміщення та навігації роботів».

Індивідуальні завдання наведені далі.

План виконання РГР:

- розв'язання задач з розділу **«**Комп'ютерні засоби проектування спеціалізованих роботів**»** –6 год.;
- розв'язання задач з розділу **«**Комп'ютерні засоби проектування універсальних роботів**»** –3 год.;
- розв'язання задач з розділу **«**Комп'ютерні засоби проектування мобільних роботів**»** –6 год.

Графік виконання індивідуальних завдань.

Номери змістових модулів	Найменування змістових модулів	Термін виконання (тиждень)	Кількість балів	Кількість кредитів
Семестровий модуль				
1	Основні принципи та засоби проектування роботів	-	-	-
2	Комп'ютерні засоби проектування спеціалізованих роботів	7	5	0,16
3	Комп'ютерні засоби проектування універсальних роботів	9	2	0,08
4	Комп'ютерні засоби проектування мобільних роботів	11	5	0,16
	Захист розрахунково-графічної роботи	12	8	0,1
Σ			20	0,5

Оцінювання розрахунково-графічної роботи здійснюється за 20-бальною системою.

1 ЗМІСТ І СКЛАД РГР

Тема розрахунково-графічної роботи (РГР) направлена на використання комп'ютерних методів розрахунку щодо вирішування задач переміщення та навігації роботів.

Зміст РГР в основному полягає у використанні комп'ютерних методів розрахунку для обробки даних отриманих з датчиків, а також для формування сигналів керування електроприводів при вирішуванні задач позиціонування та навігації стаціонарних та мобільних промислових роботів на основі апаратно-програмного комплексу Arduino.

Інформація про апаратні та програмні компоненти, які можуть бути використані при вирішуванні задач, що входять у склад РГР, наведені у додатку.

У ході виконання РГР треба вирішити задачі, що пов’язані з використанням комп'ютерних методів розрахунку для математичної обробки даних та формування сигналів керування приводами для переміщення робочого органа та самого робота.

Крім того розглядаються питання програмування роботів з контурною системою керування за допомогою мови G-команд.

На рис. 1 наведена узагальнена структурна схема промислового робота, яка складається з виконавчої системи, до яких входять маніпуляційні системи та системи переміщення, інформаційної системи, яка включає датчики внутрішньої та зовнішньої інформації, системи комп’ютерного керування, що реалізує використання комп'ютерних методів розрахунку для здійснення усіх функцій керування та зв’язку з іншими пристроями, системами відображення інформації та засобами програмування та ручного керування.

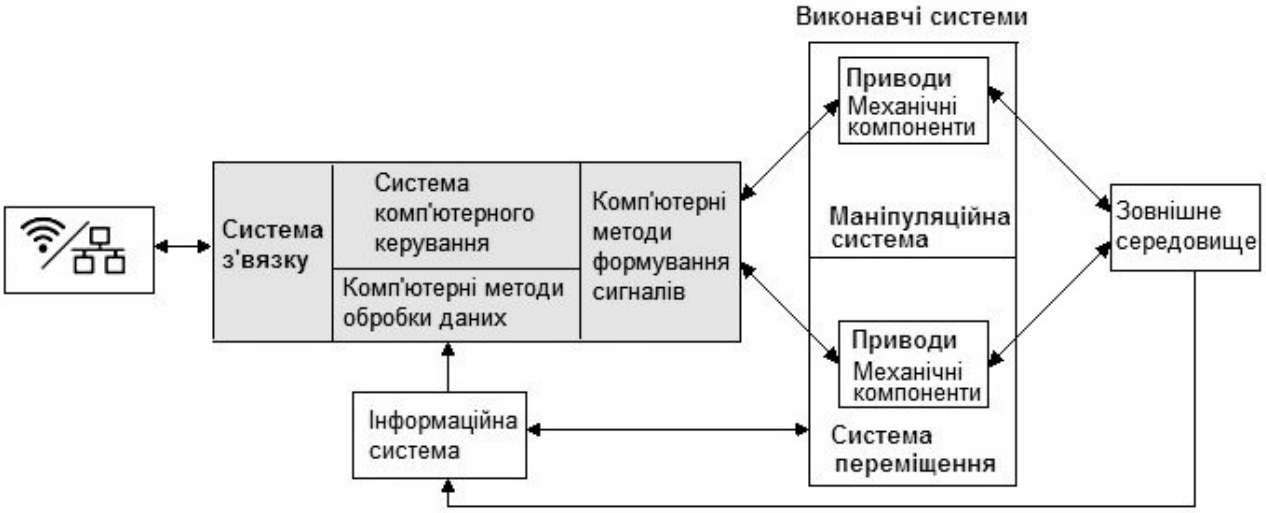


Рис. 1. Узагальнена структурна схема промислового робота

РГР складається з п'яти задач, що мають таку тематику відповідно розділам.

Тематика задач

Розділ «Комп'ютерні засоби проектування спеціалізованих роботів»

Задача 1. Використання комп’ютерних розрахунків для визначення параметрів позиційного переміщення роботів з циліндричною системою координат.

Задача 2. Використання комп’ютерних розрахунків для обробки даних та визначення відстані за допомогою ультразвукового датчика.

Розділ «Комп'ютерні засоби проектування універсальних роботів»

Задача 3. Використання мови мови G-команд для контурного керування.

Розділ «Комп'ютерні засоби проектування мобільних роботів»

Задача 4. Використання комп’ютерних розрахунків для визначення траєкторії переміщення колісного роботу.

Задача 5. Використання комп’ютерних розрахунків для орієнтації мобільного транспортного робота за допомогою лазерного сканера.

2 ІНДИВІДУАЛЬНІ ЗАВДАННЯ

Задача 1. Використання комп'ютерних розрахунків для визначення параметрів позиційного переміщення роботів з циліндричною системою координат

Теоретична частина

Маніпулятор з циліндричною системою координат має одну ланку (рука), та робочий орган, що переміщується вздовж руки (рис. 1).

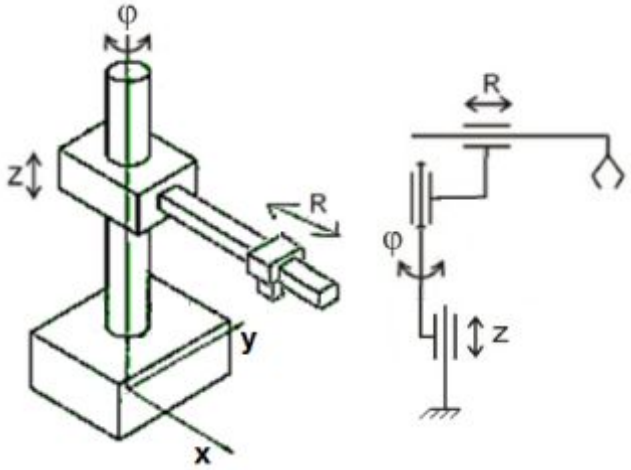


Рис. 1. Маніпулятор з циліндричною системою координат

Координати робочого органу x, y та параметри виконавчих пристроїв, а саме, кут повороту ланки маніпулятора φ та положення робочого органу на ланці *R* пов'язані такими залежностями:

$$\varphi = \text{atan}(y / x);$$
$$R = (x^2 + y^2)^{1/2}.$$

Завдання до задачі

Скласти програму для визначення та виведення на монітор параметрів φ та *R* для таких варіантів координат x та y. Значення φ вивести в градусах. Програма призначена для контролера Arduino (Основи програмування наведені у Додатку).

Варіант	1	2	3	4	5	6
x	10	15	22	16	18	12
y	12	16	24	15	8	19

Приклад вирішування задачі

Приклад програми для такої залежності:

$$R = (x^2 + y^2)^{1/2};$$
$$\varphi = \text{acos}(y / R).$$

Вихідні дані: x 12 y=16.

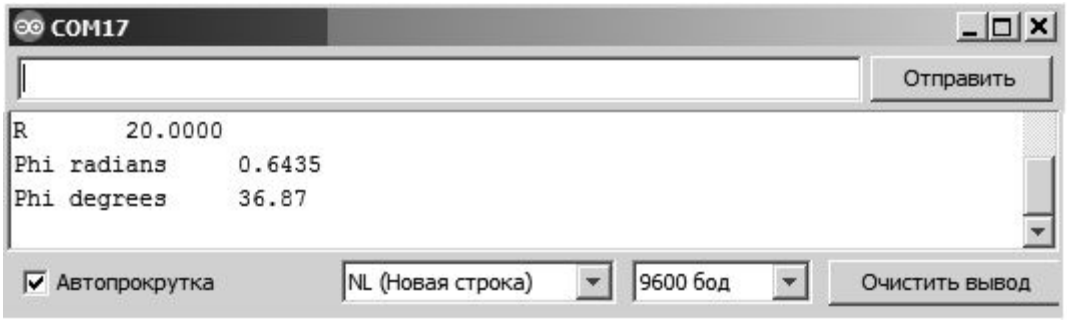
Код програми

```
float VarX = 12.0; //вихідні дані X
float VarY = 16.0; //вихідні дані Y
float VarR; //результат R
float VarPhiRad; //результат Phi у радіанах
float VarPhiDeg; //результат Phi у градусах

void setup() {Serial.begin(9600);}
```

```
void loop() {
//розрахунок
VarR = sqrt(sq(VarX) + sq(VarY));
VarPhiRad= acos (VarY/VarR);  //результат у радіанах
VarPhiDeg = VarPhiRad * RAD_TO_DEG; //результат у градусах
//вивід на монітор
Serial.print("R \t");
Serial.println(VarR, 4); //результат R
Serial.print("Phi radians \t");
Serial.println(VarPhiRad, 4); //результат Phi у радіанах
Serial.print("Phi degrees \t");
Serial.println(VarPhiDeg); //результат Phi у градусах
delay (1000);
}
```

Результат виконання програми на моніторі



Задача 2. Використання комп'ютерних розрахунків для обробки даних та визначення відстані за допомогою ультразвукового датчика

Теоретична частина

Швидкість розповсюдження ультразвуку у повітрі 343, 1 м/с (при температурі повітря 20°С). При підвищенні або зменшені температури в діапазоні від 0°С до 40°С швидкість відповідно змінюється на 0,58 м/с на градус. Таким чином при зміні температури від 0°С до 40°С швидкість розповсюдження ультразвуку у повітрі зміниться на 23,2 м/с, що складає 7% від значення температури при 0°С. Тому для мобільних роботів, що працюють на відкритому просторі, при вимірюванні відстані до об'єктів треба враховувати температуру зовнішнього середовища.

Ультразвуковий датчик визначає відстань до перешкоди шляхом вимірювання часу від випромінювання до повернення ультразвукового імпульсу у мікросекундах.

На рис 1 наведено підключення ультразвукового датчика HC-SR04 до контролера.

Датчик має 4 виводи:

VCC: "+" живлення;

TRIG (T): вхідний сигнал, підключений до виходу 9;

ECHO (R): вихідний сигнал, підключений до входу 8;

GND: "-" живлення (земля).

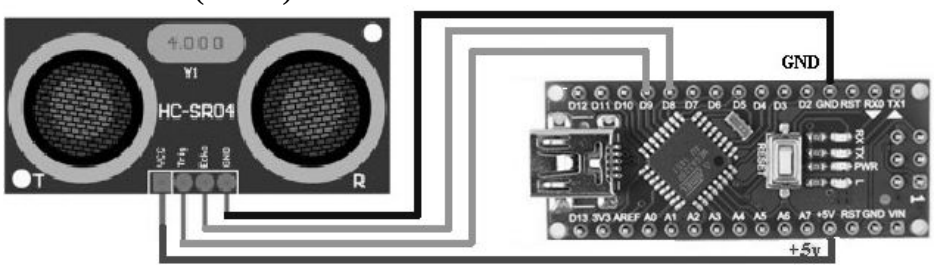


Рис. 1. Підключення далекоміра на основі ультразвукового датчика до контролера

Треба скласти алгоритм вимірювання відстані до перешкоди з урахуванням температури повітря, та програму, що реалізую цей алгоритм за допомогою контролера Arduino (Основи програмування наведені у Додатку).

Для визначення температури можна використовувати термістори, опір яких залежить від температури. Для обчислення температури шляхом вимірювання опору термістора використовують рівняння Стейнхарта-Харта, яке у спрощеній формі виглядає так:

$1/T = 1/T0 + \ln(R/R0) / B$,

де T0 – кімнатна температура, що дорівнює 25°С або 298,15 К;

T – температура, що визначається, за Кельвіном,

B - коефіцієнт, що залежить від типу термістора (у нашому випадку B = 3950);

R – опір термістора для температури, що визначається;

R0 - опір термістора для кімнатної температури (у нашому випадку R0 = 10 КОм).

Маємо температуру за Цельсієм:

$T = 1 / (1/T0 + \ln(R/R0) / B) = 1 / (0{,}003354 + \ln(10/R0) / 3950) - 273.15$.

Так, якщо шляхом вимірювання опору термістора получено значення R = 6,504, отримаємо T = 308,15 К або T = 35°С.

Завдання до задачі

1. Скласти алгоритм вимірювання відстані до перешкоди в міліметрах з урахуванням температури повітря.

2. Для заданої температури повітря визначити формулу перерахунку та перерахувати швидкість розповсюдження ультразвуку.

3. Визначити коефіцієнт перерахунку результату опитування датчика в відстань в міліметрах, на який треба поділити результат опитування датчика

4. Скласти програму, що реалізую цей алгоритм за допомогою контролера Arduino для вказаного варіанта значення температури та включає світлодіод, що підключений до виходу 13, якщо відстань до перешкоди менше встановленого варіанта значення. Для визначення змінних використовувати формат **float**.

Варіант	1	2	3	4	5	6
Температура °С	10	15	25	30	35	40
Відстань до перешкоди, мм	250	200	260	100	350	340
Варіант	7	8	9	10	11	12
Температура °С	10	15	25	30	35	40
Відстань до перешкоди, мм	100	350	340	250	200	260

5. У прикладах наведена програма для визначенні температури за Кельвіном. Змінити програму для визначення температури по Цельсію.

Визначити температуру, якщо отримані такі значення для опору термістора R:

Варіант	1	2	3	4	5	6
Опір термістора R	13,6987	11,9441	10,44	9,14743	7,3771	6,23934
Варіант	7	8	9	10	11	12
Опір термістора R	32,96	28,276	24,3274	20,9889	18,1582	15,7511

Приклад вирішування задачі

Вихідні дані:

Температура 20°С;

Відстань до перешкоди, 300 мм.

Швидкість розповсюдження ультразвуку у повітрі при температурі повітря 20°С складає 343, 1 м/с, або 34310 см/с.

Визначаємо коефіцієнт перерахунку результату опитування датчика в відстань в сантиметрах, на який треба поділити результат опитування датчика:

Kus = 1000000 мкс * 2 / 34310 см/с = 58,29 ≈ 58.

1. Алгоритм роботи можна умовно розділити на 4 етапи:
   1) подаємо імпульс тривалістю 10 мкс, на вивід Trig;
   2) датчик перетворює вхідний імпульс в 8 імпульсів частотою 40 КГц і надсилає вперед через випромінювач "T";
   3) дійшовши до перешкоди, послані імпульси відбиваються і поступають на приймач "R", внаслідок чого отримуємо вихідний сигнал на виводі Echo, тривалість якого пропорційна відстані до об'єкту;
   4) функція pulseIn(Echo, HIGH) повертає тривалість імпульсів в мікросекундах
   5) перетворюємо отриманий сигнал в відстань за формулою: тривалість імпульсу (мкс) / Kus = дистанція (см)

2. Програма для вимірювання відстані до об'єкту в сантиметрах. Якщо відстань менше або дорівнює 30 см, включається світлодіод на виході 13.

Код програми

```
#define Trig 9
#define Echo 8
#define ledPin 13                //світлодіод
```

```
void setup() {
pinMode(Trig, OUTPUT);    //визначаємо як вихід
pinMode(Echo, INPUT);     // визначаємо як вхід
pinMode(ledPin, OUTPUT); //визначаємо як вихід
}
unsigned int Kus=58;
unsigned int impulseTime=0;
unsigned int distance_sm=0;

void loop() {
digitalWrite(Trig, HIGH);    // Подаємо імпульс на вхід trig
delayMicroseconds(10);       // що дорівнює 10 мікросекунд
digitalWrite(Trig, LOW);     // відключаємо
impulseTime=pulseIn(Echo, HIGH);          // Вимірюємо тривалість імпульсу
distance_sm=impulseTime/Kus;              // Перераховуємо в сантиметри
if(distance_sm>30)            //Якщо відстань більше ніж 30 см
{
digitalWrite(ledPin, LOW);   // Світлодіод не горить
}
else                          // Якщо відстань менше або дорівнює 30 см
{
digitalWrite(ledPin, HIGH);  // Світлодіод горить
}
delay(50);
}
}
```

3. Програма для вимірювання температури за Кельвіном

Код програми

```
float R; //вихідні дані R0
float T; //результат T

void setup() {
  Serial.begin(9600);
  Serial.println("T = 1/(0,003354 + ln (R/10) / 3950)");
}

void loop() {
  // Коли в буфері є дані
  while (Serial.available() > 0)
  {
    R = Serial.parseFloat(); // Отримане число
   if (Serial.read() == '\n') // Кінець передачі
{
    T =  1 /(0.003354 + (log(R/10))/3950);
    Serial.print("R = \t");
    Serial.println(R, 4); //вихідні дані R
    Serial.print("T = \t");
    Serial.println(T, 4); //вихідні дані  T
  }
}
}
```

Задача 3. Використання мови мови G-команд для контурного керування

Теоретична частина

Для роботів з контурною системою керування програмування переміщення часто здійснюється за допомогою так званої системи G-команд.

Програма складається з кадрів, в яких як правило включені команди для зазначення параметрів одного переміщення, наприклад, функція руху (прискорений, лінійна чи кругова інтерполяція і т.д.), координати кінцевої точки руху, швидкість переміщення тощо.

В командах може використовувати абсолютне або відносне зазначення розміру. У першому разі усі розміри зазнаються відносно початкової точки координат. У другому – відносно положення робочого органу на початок виконання команди.

Деякі команді переміщення наведені у табл. 1.

Таблиця 1

Команда	Зазначення
G0 X… Y… Z…	прискорений рух до точки X… Y… Z… (найбільш можлива швидкість)
G1 X… Y… Z… F…	лінійна інтерполяція до точки X… Y… Z… (F… задана швидкість переміщення мм/хв)
G2 X… Y… Z… CR=…	колова інтерполяція за годинниковою стрілкою до точки X… Y… Z… з радіусом CR=…
G3 X… Y… Z… CR=…	колова інтерполяція проти годинникової стрілки до точки X… Y… Z… з радіусом CR=…
G90	абсолютне зазначення розміру
G91	відносне зазначення розміру

Завдання до задачі

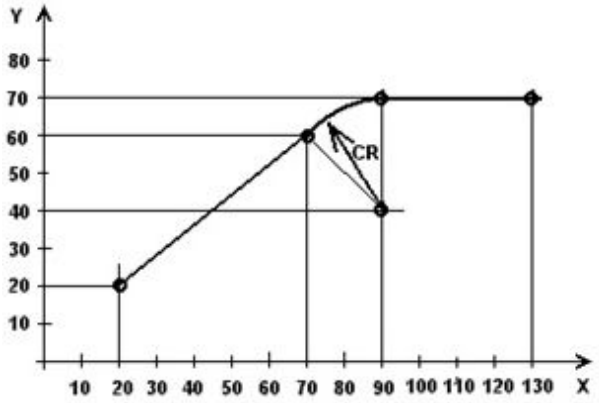
Для вказаного на рисунку варіанта переміщення по траєкторії:

1) навести послідовність переміщень по вказаній траєкторії;

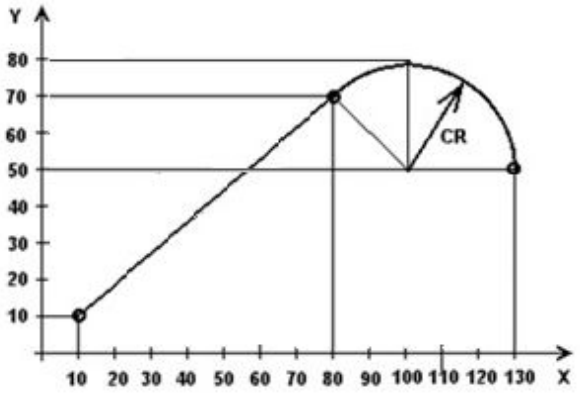
2) скласти програму переміщення за допомогою системи G-команд.

Варіанти траєкторій переміщення

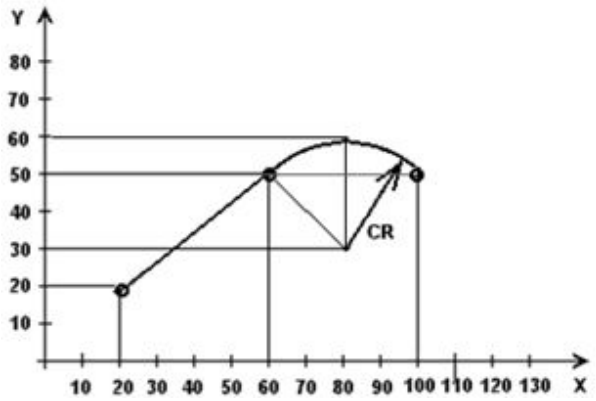
Варіант 1



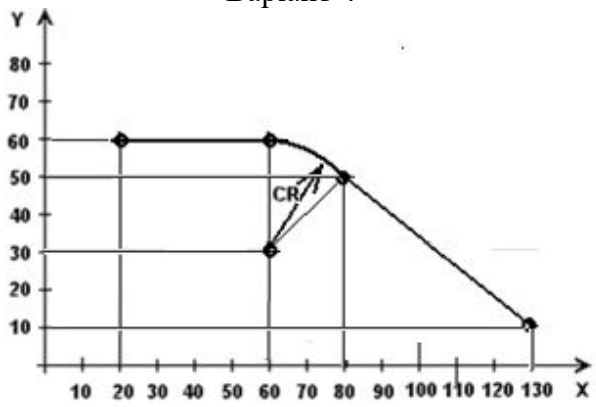
Варіант 2



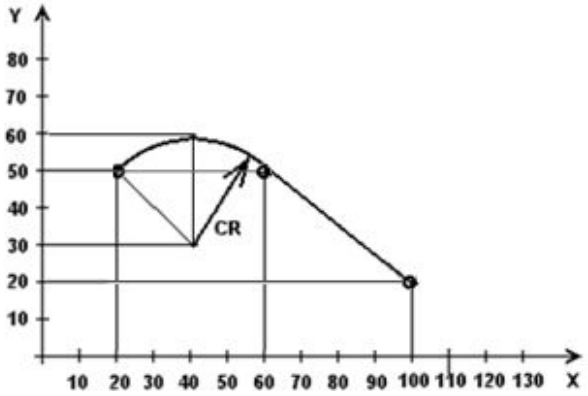
Варіант 3



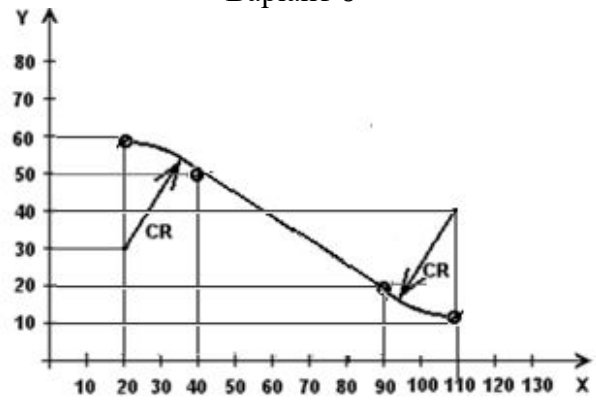
Варіант 4



Варіант 5

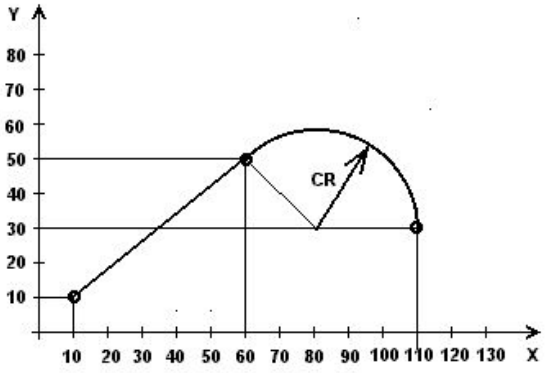


Варіант 6



Приклад вирішування задачі

Для приклада розглянемо програму переміщення по траєкторії, що наведена на рисунку,



Послідовність переміщень по вказаній траєкторії:

1) швидке переміщення в початкову позицію X=10, У=10,
2) лінійне переміщення у позицію X=60, У=50 при заданій швидкості 500 мм/хв,
3) переміщення за допомогою колової інтерполяції у позицію X=110, У-30 із радіусом CR=30.

Відповідна програма має такий вигляд:

N10 G0 G90 X10 Y10
N20 G1 X60 Y50 F500
N30 G2 X110 Y30 CR=30

Задача 4. Використання комп'ютерних розрахунків для визначення траєкторії переміщення колісного роботу

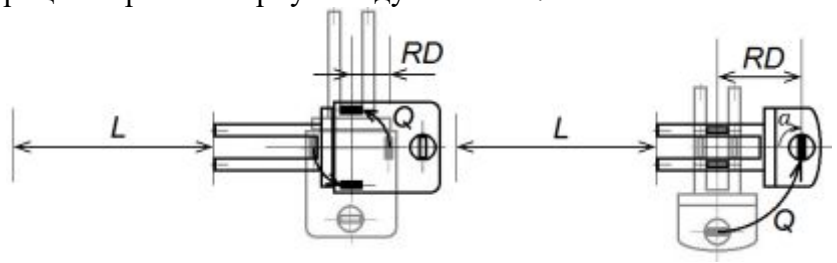
Теоретична частина

Для переміщення вантажу на складі часто використовують мобільні транспортні роботи з диференційним приводом та приводом типу трицикл.

Робот з диференційним приводом (рис. 1,а) має два мотора, по одному на кожне колесо (рис. 1). Зміна напрямку руху здійснюється за рахунок різних швидкостей коліс.

Триколісний робот типу трицикл (рис. 1,б) має два неповоротних опорних колеса та ведуче (приводне) рульове колесо з двома моторами — один для руху, інший для руління.

Переміщення по прямій робота з диференційним приводом здійснюється, коли оба колеса обертаються з однаковою швидкістю, а робота з приводом типу трицикл, коли ведуче рульове колесо знаходиться у тому ж напрямку, як і опорні колеса. Для повороту робота з диференційним приводом колеса обертаються з різною швидкістю, а для повороту робота з приводом типу трицикл треба повернути ведуче колесо.



а) б)

Рис. 1. Рух триколісного робота (а) та розворот з мінімальним радіусом (б)

Найбільш простим є алгоритм переміщення з використанням кусочно-ламаної траєкторії. Для програмування шляху переміщення роботів при цьому використовують переміщення по прямій та поворот на місці.

При переміщенні по прямій відстань, яка буде пройдена роботом та, відповідно, приводним колесом, дорівнює *L*.

При використанні для визначення шляху переміщення одометричного датчика, який має *nd* імпульсів на одне обертання колеса з діаметром *dW* , шлях *Ld*, який пройде колесо за одне обертання складає

$$Ld = PI * dW.$$

Тоді кількість імпульсів *nL*, яку зафіксує одометричний датчик при переміщенні на відстань *L* дорівнює

$$\boldsymbol{nL = (L * nd) / (PI * dW).}$$

Для зменшення площі, що потрібна для розвороту робота доцільно здійснювати розворот з мінімальним радіусом (розворот на місці). У цьому випадку для повороту робота з диференційним приводом колеса обертаються з однаковою швидкістю в різні сторони, а для повороту робота з приводом типу трицикл ведуче рульове колесо повернено на 90° ($a = \pi/2$), а радіус розвороту дорівнює *RD* (рис. 4).

Для розвороту на 360° ведуче колесо повинне пройти шлях

$$L360 = 2 * PI * RD,$$

де *RD* - радіус розвороту.

Таким чином, для повороту на кут *Q* градусів ведуче колесо повинно пройти шлях *LQ*, що дорівнює

$$LQ = Q * 2 * PI * RD / 360 = Q * PI * RD / 180.$$

Тоді кількість імпульсів *nQ*, яку зафіксує одометричний датчик при переміщенні на відстань *LQ* дорівнює

$$\boldsymbol{nQ = (LQ * nd) / (PI * dW) = (Q * RD * nd) / (180 * dW).}$$

Завдання до задачі

Колісний робот типу трицикл має діаметр ведучого рульового колеса *dW* м. Відстань між віссю повороту рульового колеса та віссю опорних коліс складає *RD* м. Кількість імпульсів одометричного датчика на одне обертання колеса *nd* .

Треба визначити кількість імпульсів *nL*, яку повинен зафіксувати одометричний датчик при переміщенні по прямій на відстань *L*, та кількість імпульсів *nQ*, яку повинен зафіксувати одометричний датчик при повороті робота на кут *Q*.

Маршрут переміщення мобільного робота:

- рух прямою на відстань *L;*
- поворот на *Q* градусів.

Варіанти параметрів робота та маршруту наведені в таблиці.

Варіант	1	2	3	4	5	6
dW, м	0,2	0,25	0,3	0,35	0,4	0,3
RD, м	0,4	0,5	0,6	0,7	0,8	0,6
nd	20	24	26	32	28	40
L, м	2	2,5	3	2,6	4	3,2
Q, °	30	60	45	90	15	75

Скласти програму для визначення *nL* та *nQ* та виведення отриманих параметрів на монітор. За результатами визначити точність переміщення.

Приклад вирішування задачі

Вихідні дані

dW = 0,3 м; RD = 0,7 м; nd =20; L = 3,5 м; Q, = 30°

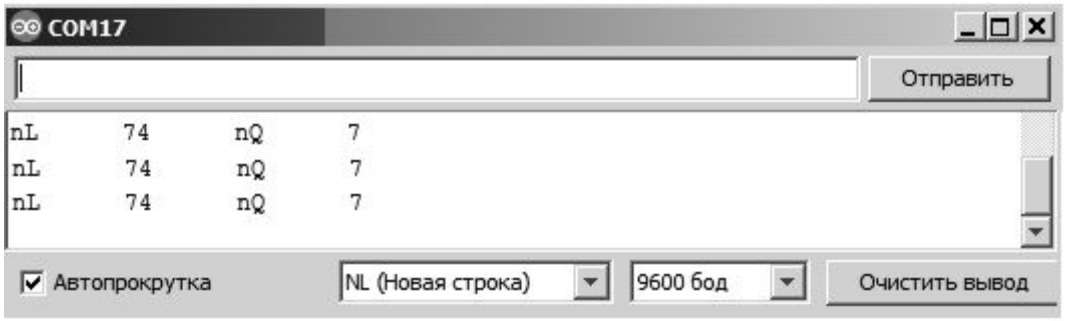
Код програми

```
float dW = 0.3;         //вихідні дані dW
float RD = 0.7;         //вихідні дані RD
float nd = 20;          // вихідні дані nd
float L =3.5;           // вихідні дані L
float Q = 30.0;         // вихідні дані Q
int nL;                 //результат nL
int nQ;                 //результат nQ

void setup() {Serial.begin(9600);}

void loop() {
//розрахунок
nL = (L * nd) / (PI * dW);
nQ = (Q *  RD * nd) / (180.0 * dW)
//вивід на монітор
Serial.print("nL \t");
Serial.print(nL); //результат nL
Serial.print(" \t");
Serial.print("nQ \t");
Serial.println(nQ); //результат nQ
delay (1000);
}
```

Результат виконання програми на моніторі



Точність переміщення по прямій L / nL = 0,047 м, точність переміщення при розвороті Q / nQ = 4°.

Задача 5. Використання комп'ютерних розрахунків для орієнтації мобільного транспортного робота за допомогою лазерного сканера

Теоретична частина

Розглянемо орієнтацію мобільного робота за допомогою лазерного сканера з двома рефлекторами.

Приклад використання лазерних сканерів для визначення положення робота при переміщенні вздовж стелажу наведений на рис. 1. Лазерний сканер видає відстані до рефлекторів L_1 та L_2, а також відповідні кути повороту сканера відносно орієнтації робота та рефлекторів α_1, α_2.

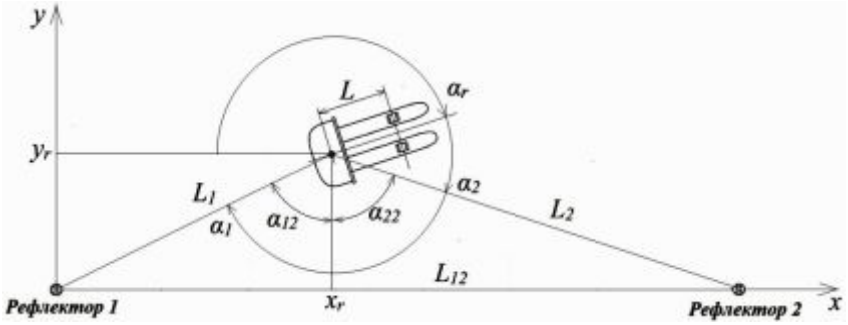


Рис. 1. Використання лазерних датчиків для визначення положення транспортного робота

На рисунку використовуються такі позначення:

L_{12} – відстань між рефлекторами 1, 2;

L_1 - відстань до рефлектора 1;

L_2 - відстань до рефлектора 2;

α_1 – кут напрямку на рефлектор 1 відносно вихідної орієнтації робота;

α_2 - кут напрямку на рефлектор 2 відносно вихідної орієнтації робота;

α_{12} - кут між лінією напрямку на рефлектор 1 та лінією, перпендикулярною до лінії відстані між рефлекторами;

α_{22} - кут між лінією напрямку на рефлектор 2 та лінією, перпендикулярною до лінії відстані між рефлекторами;

α_r – орієнтація робота;

x_r та y_r - поточні координати робота.

Для визначення положення та орієнтації робота треба знайти його координати x_r та y_r, а також його орієнтацію α_r.

Вибираємо систему координат, де ось x збігається з лінією відстані між рефлекторами а ось y в напрямку, де здійснюється переміщення робота. Тоді x_r та y_r позначають поточні координати робота, що пов'язані з відстанями до рефлекторів L_1, L_2 та відстанню між рефлекторами L_{12} такими залежностями:

$$y_r^2 + x_r^2 = L_1^2, \qquad y_r^2 + (L_{12} - x_r)^2 = L_2^2, \; y_r^2 = L_1^2 - x_r^2.$$

Після проведення наступних перетворювань:

$$L_1^2 - x_r^2 + (L_{12} - x_r)^2 = L_2^2; \quad L_1^2 - x_r^2 + L_{12}^2 - 2 L_{12} x_r + x_r^2 = L_2^2;$$

$$L_1^2 + L_{12}^2 - L_2^2 = 2 L_{12} x_r,$$

отримаємо, що x_r та y_r дорівнюють:

$$\boldsymbol{x_r = (L_1^2 + L_{12}^2 - L_2^2) / 2 L_{12}; \qquad y_r = (L_1^2 - x_r^2)^{1/2}.}$$

Оскільки:

$$\cos \alpha_{22} = y_r / L_2, \qquad \alpha_{22} = \text{arc } \cos (y_r / L_2).$$

Для орієнтації робота α_r отримаємо:

$$\boldsymbol{\alpha_r} = 270° - \alpha_{22} - \alpha_2 = \mathbf{270° - \alpha_2 - arc\ cos\ (y_r / L_2)}.$$

Завдання до задачі

Треба визначити положення та орієнтації робота треба знайти його координати x_r та y_r, а також його орієнтацію α_r, виходячи з відстанями між рефлекторами $\boldsymbol{L_{12}}$ та результатів

вимірювання L_1 , L_2 , α_1 , α_2,, які у даному випадку визначаються на макеті (рис. 2) та встановлюються за допомогою монітора.

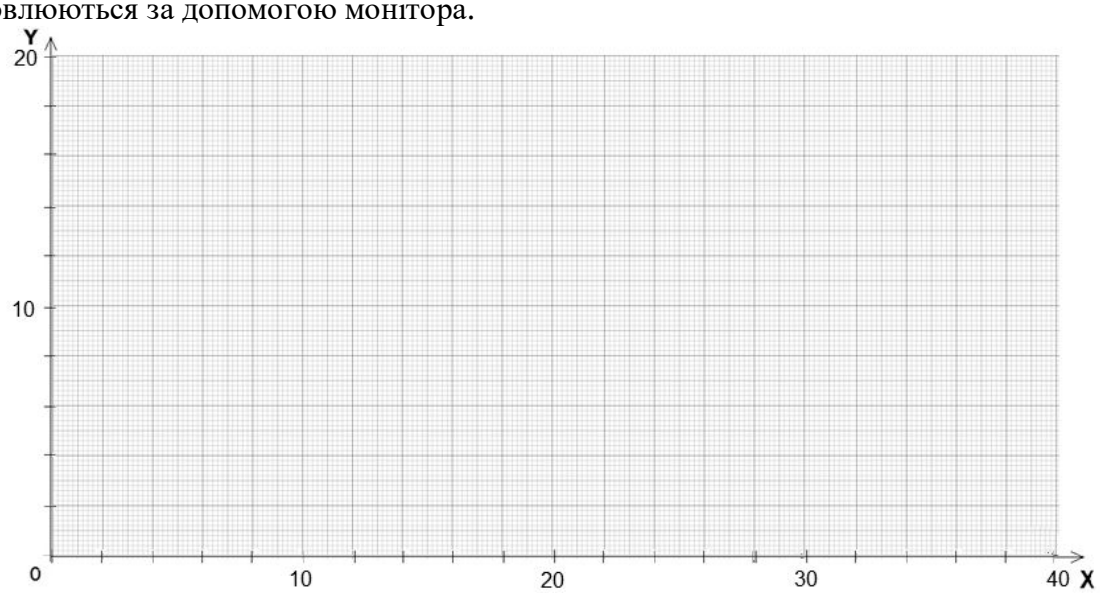


Рис. 2. Макет для визначення результатів опитування датчика

Варіанти параметрів робота та маршруту наведені в таблиці.

Варіант	1	2	3	4	5	6
Відстань між рефлекторами L_{12}, *м*	22	25	30	35	37	40

Скласти програму для визначення x_r та y_r та виведення отриманих параметрів на монітор.

Приклад вирішування задачі

Вихідні дані

L_{12} =30 м.

Дані датчика, що отримані за допомогою стенда (рис. 3).

L_1 = 20 м; L_2 = 16 м; α_1 = 293°; α_2 = 180°.

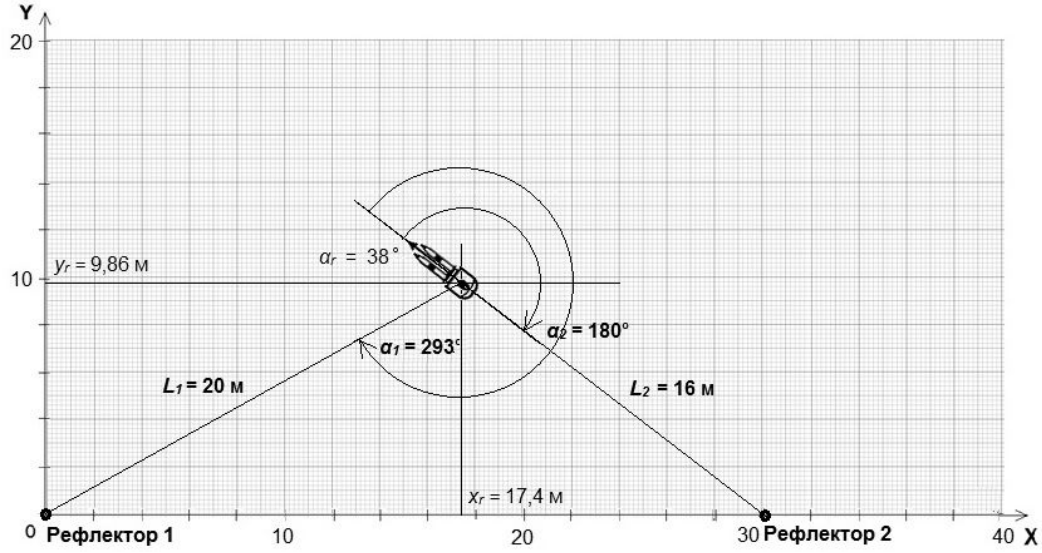


Рис. 3. Результати опитування датчика

Код програми

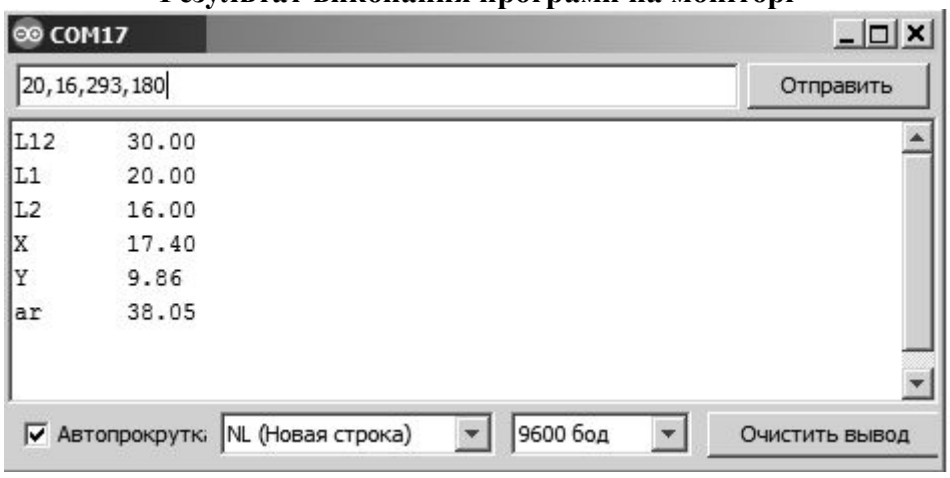
```
float VarL12 = 30.0; //відстань між рефлекторами L12
float VarL1; //результат положення L1
float VarL2; //результат положення L2
float a1Deg;  //результат кута a1 у градусах
```

```
float a2Deg;  //результат кута a2 у градусах
float VarX; //X
float VarY; //Y
float a22Rad; //результат кута a22 у радіанах
float a22Deg; //результат кута a22 у градусах
float arDeg; //результат кута ar у градусах

void setup() {
  // put your setup code here, to run once:
Serial.begin(9600);     }

void loop() {
// Коли в буфері є дані
while (Serial.available() > 0)  {
VarL1 = Serial.parseFloat();  // Отримане число L1
VarL2 = Serial.parseFloat();  // Отримане число L2
a1Deg = Serial.parseFloat();  // Отримане число a1
a2Deg = Serial.parseFloat();  // Отримане число a2
if (Serial.read() == '\n') {         // Кінець передачі
VarX = (sq(VarL1) + sq(VarL12)-sq(VarL2))/(2*VarL12);
VarY = sqrt(sq(VarL1) - sq(VarX));
a22Rad= acos (VarY / VarL2);  //результат a22 у радіанах
a22Deg = a22Rad * 180.0 / PI; //результат a22 у градусах
arDeg = 270 - a22Deg - a2Deg;
Serial.print("L12 \t");
Serial.println(VarL12, 2); //вихідні дані
Serial.print("L1 \t");
Serial.println(VarL1, 2); //вихідні дані
Serial.print("L2 \t");
Serial.println(VarL2, 2); //вихідні дані
Serial.print("X \t");
Serial.println(VarX, 2); //результат X
Serial.print("Y \t");
Serial.println(VarY, 2); //результат Y
Serial.print("ar \t");
Serial.println(arDeg, 2); //результат ar
}
}
}
```

Результат виконання програми на моніторі



3 ОФОРМЛЕННЯ РЕЗУЛЬТАТІВ РОБОТИ

Результати розрахунково-графічної роботи оформляються у вигляді пояснювальної записки, яка оформляється відповідно до вимог і стандартів.

Пояснювальна записка розрахунково-графічної роботи повинна містити такі розділи.

1. Вступ.
2. Результат вирішування задачі 1.
3. Результат вирішування задачі 2.
4. Результат вирішування задачі 3.
5. Результат вирішування задачі 4.
6. Результат вирішування задачі 5.
7. Висновки по роботі.

Список літератури

1. C++. Основи програмування. Теорія та практика : підручник / [О.Г. Трофименко, Ю.В. Прокоп, І.Г. Швайко, Л.М. Буката та ін.] ; за ред.О.Г.Трофименко. – Одеса: Фенікс, 2010. – 544 с. ISBN 978-966-438-240-0 URL: http://www.dut.edu.ua/uploads/l_538_66572861.pdf (дата звернення 16.04.2020)

2. Доля В.М. Програмування, введення та відпрацювання управляючих програм для верстатів з ЧПУ та РТК: Навчальний посібник. − Харків: НТУ „ХПІ”, 2004. − 169 с. URL: http://repository.kpi.kharkov.ua/bitstream/KhPI-Press/5820/1/Dolya_Prohramuvannia_vvedennia_2004.pdf (дата звернення 17.04.2020)

3. Навчальний посібник з дисципліни Маніпулятори та промислові роботи. Для студентів бакалаврів, спеціальності: 131 - Прикладна механіка, 133 – Галузеве машинобудування, / Укладачі: Михайлов Є. П., Лінгур В.М. – Одеса: ОНПУ, 2019. - 233 с. Рег ном. МВ09960 23.11.2018, №6223-РС-2018 URL: http://memos.library.opu.ua:8080/memos/jsp/materials.iface?mId=37024

4. Михайлов Є. П. Навчальний посібник з дисципліни "Мобільні роботи"для студентів за фахом 131 - Прикладна механіка – спеціалізація - Мехатроніка та промислові роботи / Укладач: Михайлов Є. П. Одеса: ОНПУ. 2016, – 239 с. Укладач: Михайлов Є. П. Одеса: ОНПУ, 2016, – 239 с. Рег.ном. НП07368 01.06.2016 №3765-РС-2016 Одеса: ОНПУ, 2016 URL: http://memos.library.opu.ua:8080/memos/jsp/materials.iface?mId=28063

Електронні ресурси

2. Полный список команд языка Ардуино веб-сайт. URL: https://alexgyver.ru/lessons/arduino-reference/ (дата звернення 16.04.2020)

Додаток

Структура програми

Синтаксис мови Arduino повторює мову С, але є деякі спрощення. Програма, або як її ще називають, скетч Arduino складається з самостійного файла, в якому, на відміну від мови С, треба визначити принаймні, дві секції: перша називається setup(), а друга loop(). Змінні, доступні з обох секцій програми, повинні бути оголошені за їх межами, як глобальні змінні.

Функція setup () викликається, коли стартує скетч. Використовується для ініціалізації змінних, визначення режимів роботи висновків, запуску використовуваних бібліотек і т.д. Функція setup запускає тільки один раз, після кожної подачі живлення або скидання плати Arduino.

Після закінчення обробки setup() Arduino починає циклічне виконання інструкцій в функції loop(). Після виконання всіх операндів, цикл знову повторюється (нескінченний цикл).

```
void setup () {
 ...
 }
void loop () {
 ...
 }
```

Обидві функції setup() та loop() задекларовані як блоки void, тобто вони нічого не повертають.

Призначення даних

Змінні

Змінна - це місце зберігання даних. Вона має ім'я, значення і тип.

Програма використовує змінні для зберігання проміжних даних обчислень.

Для обчислень можуть бути використані дані різних форматів, різної розрядності, тому у змінних в мові С є, наприклад, такі типи (табл. 1).

Типи даних вибираються виходячи з необхідної точності обчислень, форматів даних і т.п.

Таблиця 1

Тип даних	Розрядність, біт	Діапазон чисел
boolean	8	true, false
byte	8	0 … 255
int	16	-32768 … 32767
unsigned int	16	0 … 65535
word	16	0 … 65535
long	32	-2147483648 … 2147483647
unsigned long	32	0 … 4294967295
float	32	-3.4028235E+38 … 3.4028235E+38
double	32	-3.4028235E+38 … 3.4028235E+38

Можна використовувати типи даних, у яких перша частина uint або int означає число без знаку або зі знаком, далі йде розмір даних в бітах, а _t вказує що це тип. Наприклад:

uint32_t x; - означає, що x змінна без знаку, яка складається з 4 байт,
int16_t x; - означає, що x змінна зі знаком, яка складається з 2 байт.
Всі змінні повинні бути оголошені до того як будуть використовуватися.
Для оголошення змінних треба вказуєти тип даних, а потім ім'я змінної.
int x; // оголошення змінної з ім'ям x типу int
float widthBox; // оголошення змінної з ім'ям widthBox типу float

Змінна може бути оголошена в будь-якій частині програми, але від цього залежить, які блоки програми можуть її використовувати.

Змінні, що оголошені на початку програми, до функції void setup (), вважаються глобальними і доступні в будь-якому місці програми.

Локальні змінні, що оголошуються всередині функцій або таких блоків, як цикл ***for***, можуть використовуватися тільки в оголошених блоках.

При оголошенні змінної можна задати її початкове значення (проініціалізувати).
int x = 0; // оголошується змінна x з початковим значенням 0

Можна використовувати стандартні константі формату float, наприклад,
Перерахування радіанів у градуси:
RAD_TO_DEG = 57.295779513082320876798154814105f;
Перерахування градусів у радіани:
DEG_TO_RAD = 0.017453292519943295769236907684886f;
Число π:
PI = 3.14159265358979323846264338327 95f;
Число π/2:
PI_BY_TWO = 1.57079632679489661923132169163 98f;
Число 2π:
TWO_PI = 6.28318530717958647692528676655 9f;

Функції

Арифметичні операції:

= присвоєння;
+ складання;
- віднімання;
* множення.

Ці оператори повертають результат виконання арифметичних дій над двома операндами. Результат, що повертається, буде залежати від типу даних операндів, наприклад, 9/4 поверне 2, тому що операнди 9 і 4 мають тип int. Також слід стежити за тим, щоб результат не вийшов за діапазон допустимих значень за видом даних. Так, наприклад, складання 1 зі змінною типу int і значенням 32 767 поверне -32 768. Якщо операнди мають різні типи, то тип з більш "широким" діапазоном буде використаний для обчислень.

Якщо один з операндів має тип float або double, то арифметика "з плаваючою комою" буде використана для обчислень.

Синтаксіс:
result = value1 + value2;
result = value1 - value2;
result = value1 * value2;
result = value1 / value2;
Параметри:
value1: будь-яка змінна або константа;
value2: будь-яка змінна або константа.

Операції відношення:

== дорівнює
!= не дорівнює

<	менше
>	більше
<=	менше або дорівнює
>=	більше або дорівнює

Логічні операції:

&&	логічне І
||	логічне АБО
!	логічне НІ

Квадратний корінь числа.
sqrt(x)
Параметри:
x: число, будь-який тип даних.
Значення, що повертаються:
double, квадратний корінь числа.

Квадрат числа.
sq(x)
Параметри:
x: число, будь-який тип даних.
Значення, що повертаються:
квадрат числа.

Натуральний логарифм
log (x)
Параметри:
x: число, тип даних float.
Значення, що повертаються:
натуральний логарифм числа.

Десятинний логарифм
log10 (x)
x: число, тип даних float.
Значення, що повертаються:
десятинний логарифм числа.

Тригонометричні функції
sin(rad), cos(rad), tan(rad).
Параметри:
rad: кут у радіанах, тип даних float.
Значення, що повертаються:
тригонометрическая функція, тип даних double.

Зворотні тригонометричні функції
asin(x), acos(x), atan(x).
Параметри:
x: число, тип даних float.
Значення, що повертаються:
зворотна тригонометрическая функція у радіанах, тип даних float.

Управління програмою.

Оператор IF перевіряє умову в дужках і виконує наступне вираження або блок у фігурних дужках, якщо умова істинна.

```
if (x == 5)          // якщо x=5, то виконується z=0
z=0;
if (x > 5)           // якщо x > 5, то виконується блок z=0, y=8;
{    z=0;    y=8; }
```

Оператор IF … ELSE дозволяє зробити вибір між двох варіантів.

```
if (x > 5)           // якщо x > 5, то виконується блок z=0, y=8;
 { z=0;
   y=8; }
 else                 // в іншому випадку виконується цей блок
 { z=0;
   y=0;   }
```

Оператор ELSE IF – дозволяє зробити багаторазовий вибір

```
if (x > 5)           // якщо x > 5, то виконується блок z=0, y=8;
  { z=0;
    y=8;    }
else if (x > 20)   // якщо x > 20, виконується этот блок
  {   }
else                 // в іншому випадку виконується цей блок
  { z=0;
    y=0;   }
```

Цикл FOR. Конструкція дозволяє організовувати цикли з заданим кількістю ітерацій. Синтаксис виглядає так:

```
for (дія до початку циклу; умова продовження циклу; дія в кінці кожної ітерації)
{// код тіла циклу
}
```

Приклад циклу з 100 ітерацій.

```
for ( i=0; i < 100; i++ )        // початкове значення 0, кінцеве 99, крок 1
{
    sum = sum + I;
}
```

Функції управління вводом / виводом.

Для роботи з цифровими виводами в системі Ардуіно є такі функції, дозволяють задати режим виводу та встановити виводи в певний стан.

Для визначення стану виводів в цих функціях використовуються константи HIGH і LOW, які відповідають високому і низькому рівню сигналу.

Функція pinMode(pin, mode)

Встановлює режим виводу (вхід або вихід).

Параметри:

- pin - номер виводу;
- mode - режим виведення.

mode = INPUT вивід визначено як вхід, підтягуючий резистор відключений.

mode = INPUT_PULLUP вивід визначено як вхід, підтягуючий резистор підключений (вихідний стан HIGH).

mode = OUTPUT вивід визначено як вихід.

Функція не повертає нічого.

Функція digitalWrite(pin, value)

Встановлює стан виходу (високий або низький).
Параметри:

- pin - номер виводу;
- value - стан виходу.

value = LOW встановлює вихід в низький стан
value = HIGH встановлює вихід в високе стан
Функція не повертає нічого.

Функція digitalRead(pin)

Зчитує стан входу.
Параметри: pin - номер виводу.
Повертає стан входу:

- якщо digitalRead (pin) = LOW, то на вході низький рівень;
- якщо digitalRead (pin) = HIGH, то на вході високий рівень

Функція analogRead(pin)

Зчитує стан аналогового входу.
Параметри: pin - номер виводу.
Повертає стан входу, що мають значення від 0 до 1023, які відповідають вхідній напрузі в діапазоні 0 – 5В.

Функція analogWrite(pin, value)

Видає аналогову напругу у вигляді сигналу з широтно-імпульсною модуляцією.
Параметри:

- pin - номер виводу;
- value – коефіцієнт заповнення у межах від 0 (нема сигналу) до 255 (постійна напруга)..

Функції часу.

Дозволяють визначити та встановити час.

Функція millis()

Повертає кількість мілісекунд, що пройшли з моменту старту програми.

Функція micros()

Повертає кількість мікросекунд, що пройшли з моменту старту програми.

Функція delay(ms)

Призупиняє виконання програми на ms мілісекунд.

Функція delayMicroseconds (µs)

Призупиняє виконання програми на µs мікросекунд.

Функція pulseIn(pin, value, timeout)

Повертає тривалість імпульсу (HIGH або LOW) в мікросекундах або 0, якщо імпульс відсутній за час timeout.
Параметри:

- pin - номер виводу;
- value - стан виходу;
- timeout - (за вибором): час очікування імпульсу в мікросекундах; значення за замовчуванням - одна секунда (unsigned long).

Засоби обміну даними

Монітор порту Ардуіно та Плотер по послідовному з'єднанню - це утиліти, які вбудовані в середу програмування Arduino IDE і служить для зв'язку комп'ютера з контролером.

Монітор порту видає дані у символьному вигляді.

Плотер по послідовному з'єднанню видає дані у графичному вигляді.

Для відкриття утиліти Монітор порту необхідно натиснути на іконку в правому верхньому куті Arduino IDE, використовувати комбінацію клавіш Ctrl + Shift + M або вибрати в панелі меню: Инструменты > Монитор порта.

Для відкриття утиліти Плотер по послідовному з'єднанню необхідно вибрати в панелі меню: Инструменты > Плоттер по последовательному соединению (рис. 2).

Монітор порту Ардуіно

Для роботи з утилітою, використовують такі команди:

Serial.begin (); - команда запускає послідовний порт

Serial.end (); - зупиняє і очищає послідовний порт

Serial.print (val, format); - відправляє дані val в послідовний порт, format дозволяє налаштувати показ даних: BIN, OCT, DEC, HEX виведуть число у відповідній системі числення, а цифра після виведення float дозволяє налаштовувати виведене кількість знаків після точки, за замовчуванням 2.

Serial.print(78) - відправляє "78"

Serial.print(1.23456) - відправляє "1.23"

Serial.print(1.23456, 4) - відправляє "1.2346"

Serial.print('N') - відправляє "N"

Serial.print("Hello world.") - відправляє "Hello world."

Serial.print("\t"); - табуляція

Serial.println (); - відправляє дані з перенесенням рядка

Serial.read (); - приймає дані з послідовного порту

Serial.parseInt (); - читання цілих чисел з монітора

Serial.parseFloat(); - читання чисел з плаваючою точкою з монітора

Serial.available() - повертає кількість байт (символів) доступних для зчитування з буфера послідовного порту

Введення даних з монітору здійснюється за допомогою функції:

```
while (Serial.available() > 0)
{
Var1 = Serial.parseFloat(); // Перше отримане число
Var2 = Serial.parseFloat(); // Друге отримане число
if (Serial.read() == '\n') // Кінець передачі
{
Var3 = sqrt(sq(Var1) + sq(Var2)); //обчислення кореня квадратного з суми квадратів
…
}
```

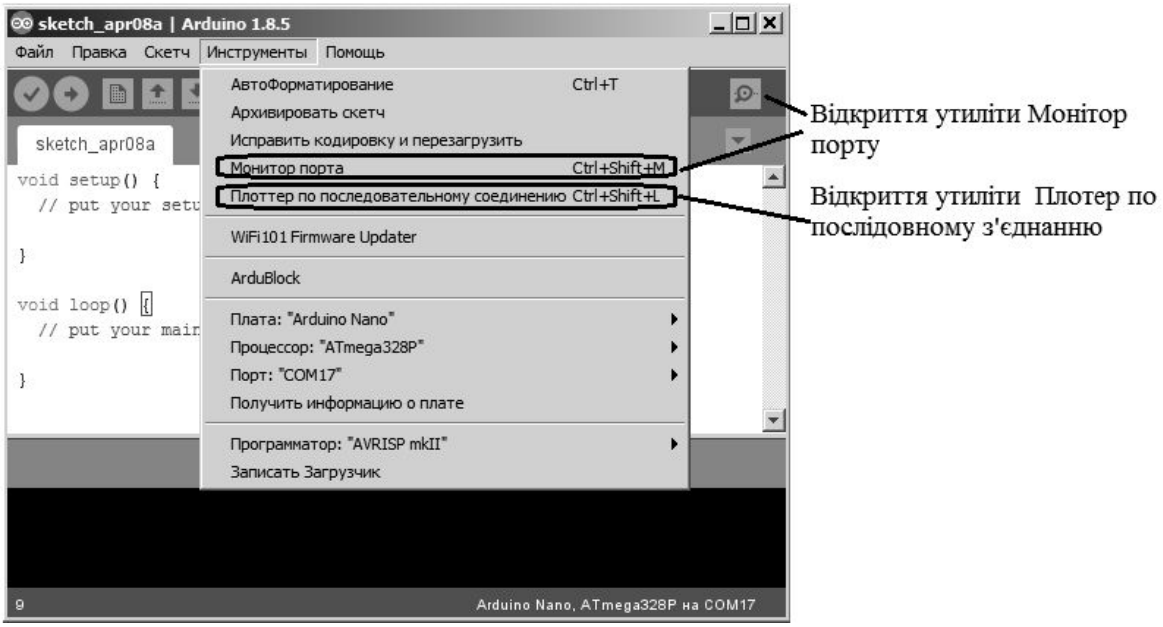


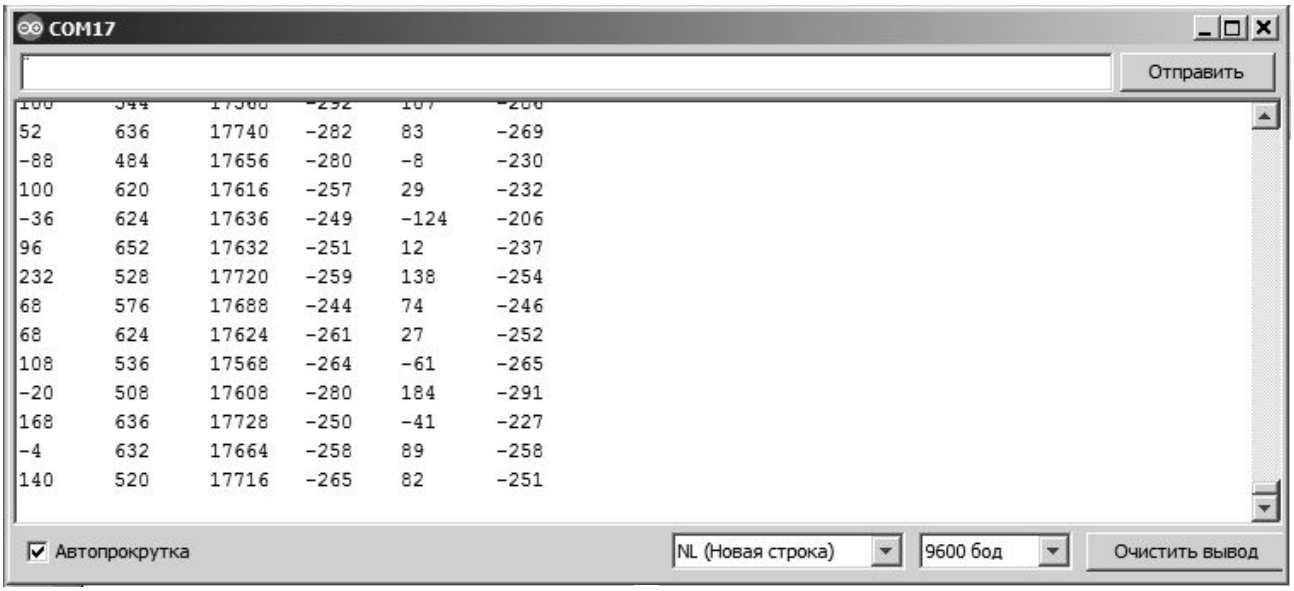
Рис. 2. Відкриття утиліт Монітор порту та Плотер по послідовному з'єднанню

Плотер по послідовному з'єднанню

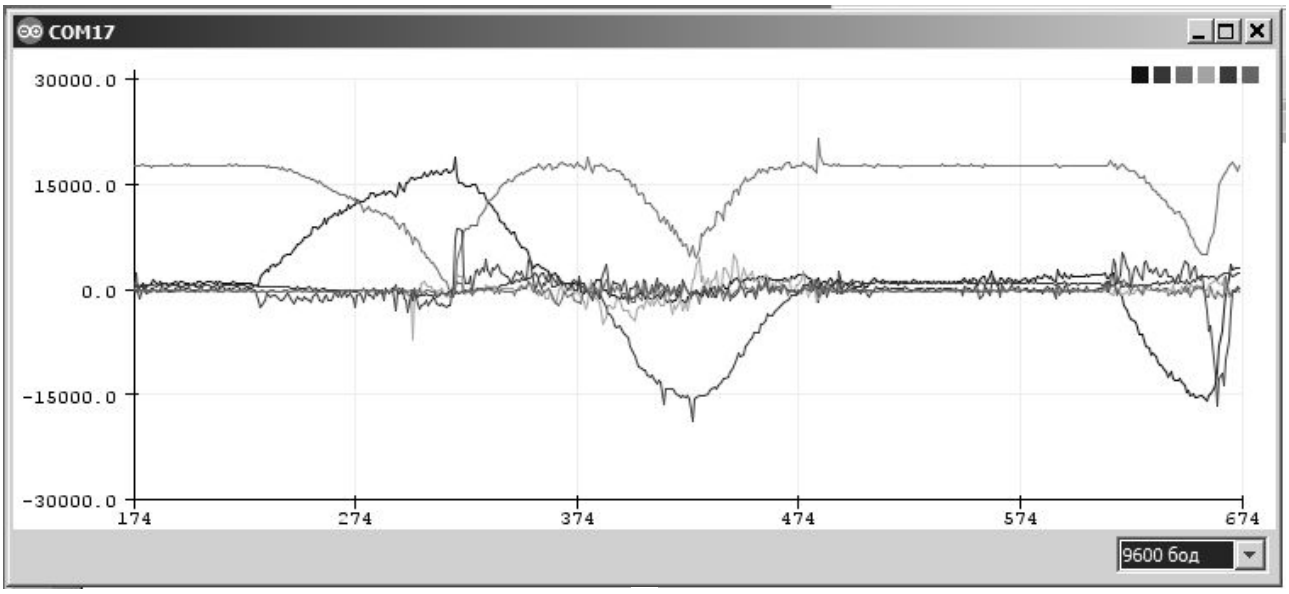
Плотер по послідовному з'єднанню здійснює виведення даних з контролера на монітор комп'ютера у графічному вигляді.

Для виведення даних на Плотер по послідовному з'єднанню використовуються команди Serial.print (); для кожної змінної, що поділяються командою табуляції Serial.print("\t"); . Для виведення останної змінної використовуються команда Serial.println ();.

На рис. 3 наведений приклад виведення даних на Монитор порта та Плотер по послідовному з'єднанню.



а)



б)

Рис. 3. Виведення даних на Монитор порта (а) та Плотер по послідовному з'єднанню (б)

Приклад програми виконання арифметичних дій

Створимо програму для послідовності арифметичних дій з отриманням результату у форматі float (з плаваючою комою) згідно з формулою:

$$z = x / (x^2 + y^2)^{1/2}$$

sketch_arithmetic_20_02_25

```
float Var1; //вихідні дані X
float Var2; //вихідні дані  Y
float Var3; //результат Z

void setup() {
Serial.begin(9600);
}

void loop() {
// Коли в буфері є дані, які були введені з монітору
while (Serial.available() > 0)
{
Var1 = Serial.parseFloat(); // Отримане число
Var2 = Serial.parseFloat(); // Отримане число
if (Serial.read() == '\n') // Кінець передачі
{
Var3 = Var1 /(sqrt(sq(Var1) + sq(Var2)));
Serial.print("X = \t");
Serial.println(Var1, 4); //вихідні дані X
Serial.print("Y = \t");
Serial.println(Var2, 4); //вихідні дані  Y
Serial.print("Z = \t");
Serial.println(Var3, 4); //результат Z
}
}
}
```

На рис. 1 наведені дані, що були отримані на моніторі за допомогою утиліти Монітор порту при обчисленні наведеної вище функції для значень x = 45, y = 56 (результат z =0,62639) . Числа, які вводяться у полі для введення даних, розділяються комою.

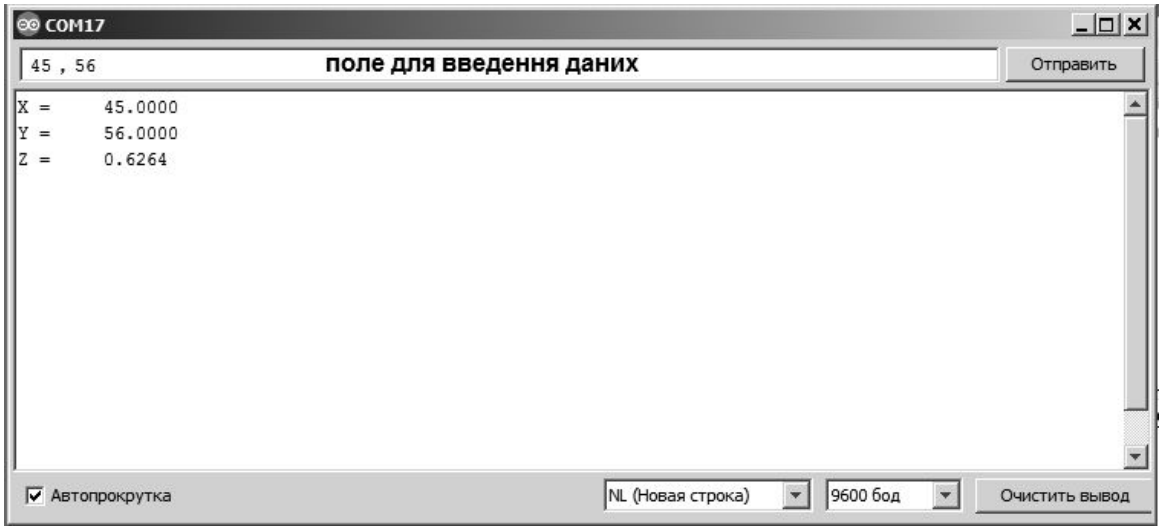


Рис. 1. Результат, отриманий на моніторі