МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

КРИВОРІЗЬКИЙ ДЕРЖАВНИЙ ПЕДАГОГІЧНИЙ УНІВЕРСИТЕТ

Факультет педагогічної освіти

Кафедра технологічної та професійної освіти

«Допущено до захисту»

Завідувач кафедри ТПО

_____Олег Цись

«___» _____2024 р.

Реєстраційний № _____

«___» _____ 2024 р.

РОЗРОБКА Й ВИГОТОВЛЕННЯ МОДЕЛІ МАНІПУЛЯТОРА НА БАЗІ ARDUINO ТА МЕТОДИКА ЇЇ ВИКОРИСТАННЯ ПІД ЧАС ВИВЧЕННЯ ТЕХНОЛОГІЇ В ПРОФІЛЬНІЙ ШКОЛІ

Кваліфікаційна робота студента
групи ТНм-23
ступінь вищої освіти магістр
спеціальності
014.10 Середня освіта (Трудове навчання і технології)
Кравчука Ярослава Вікторовича
Керівник: к.пед.н., доц.
Цись Олег Олександрович

Оцінка:
Національна шкала_____
Шкала ECTS_____кількість балів_____
Голова ЕК_____
Члени ЕК_____
_____
_____
_____

Кривий Ріг – 2024

ЗМІСТ

ВСТУП

В даний час на промислових підприємствах все частіше можна спостерігати закономірні процеси заміни ручної і механічної праці з використанням застарілих верстатів на роботу роботизованих машин і механізмів.

На зміну верстатам загального призначення прийшли промислові роботи (англ. industrial robot) – багатоцільові маніпуляційні роботи, що складаються з механічних маніпуляторів і перепрограмованої системи керування, які застосовується для переміщення об'єктів в просторі трьох і більше координат та для виконання різноманітних виробничих процесів.[4]

За ДСТУ 2879-94 промисловий робот – автоматична машина, стаціонарна чи пересувна, з виконавчим пристроєм у вигляді маніпулятора, який має декілька ступенів рухомості, і перепрограмовуваним пристроєм програмного керування для виконання у виробничому процесі рухових і керувальних функцій.[5]

Промислові роботи є важливими компонентами автоматизованих гнучких виробничих систем (ГВС), які дозволяють збільшити продуктивність праці. Типове застосування роботів стосується таких операцій, як зварювання, фарбування, складання, вибірка та встановлення, пакування, контроль продукції та випробування, котрі виконуються з високою надійністю, швидкістю, і точністю. [12]

Промисловий маніпулятор на сучасному виробництві є більш затребуваним ніж робітники, тому що він може виконувати завдання швидше, точніше і ефективніше. Він здатний працювати без відпочинку, не робить помилок, а також може виконувати такі види робіт, які людям не під силу. З точки зору повторюваних робіт промислові маніпулятори мають наступні переваги: висока ефективність, стабільна якість, відсутність впливу

зовнішніх факторів, 24-годинна безперервна робота та тривалий термін служби.[13]

При цьому у галузі середньої освіти склалася ситуація, коли на виробництві рік у рік збільшується кількість промислових роботів, а учні на уроках технології досі вивчають застарілі, верстати з ручним керуванням. І якщо у закладах вищої освіти технічного спрямування активно запроваджується вивчення робототехніки, то у профільній школі ситуація є значно гіршою.

Тому для ознайомлення учнів профільної школи з основами налаштування і керування промисловими роботами з програмним управлінням потрібні діючі моделі роботизованих маніпуляторів, що дозволить їм ознайомитися з конструкцією маніпулятора і навчитися створювати керуючі програми. Також потрібна методика використання даних моделей на уроках технології, що дало б учням можливість сформувати практичні навики.

Це зумовило вибір ***теми*** кваліфікаційної роботи: «***Розробка й виготовлення моделі маніпулятора на базі Arduino та методика її використання під час вивчення технології в профільній школі.»***

Мета роботи: розробити модель маніпулятора на базі Arduino та методику її використання під час вивчення технології в профільній школі.

Об'єкт**:** процес розробки й виготовлення моделі маніпулятора на базі Arduino.

Предмет: процес розробки методики використання моделі маніпулятора на базі Arduino під час вивчення технології в профільній школі.

У відповідності до мети визначено такі ***завдання:***

1. Дослідити можливості застосування апаратно обчислювальної платформи Arduino для робототехнічного проєктування.

2. Розробити й виготовити модель маніпулятора на базі Arduino.

3. Описати методику використання моделі маніпулятора на базі arduino під час вивчення технології в профільній школі.

У процесі виконання кваліфікаційної роботи були застосовані такі ***методи:*** аналіз літератури та інформаційних ресурсів, порівняння та узагальнення вітчизняного й світового досвіду, вивчення технічної документації, проектування, моделювання.

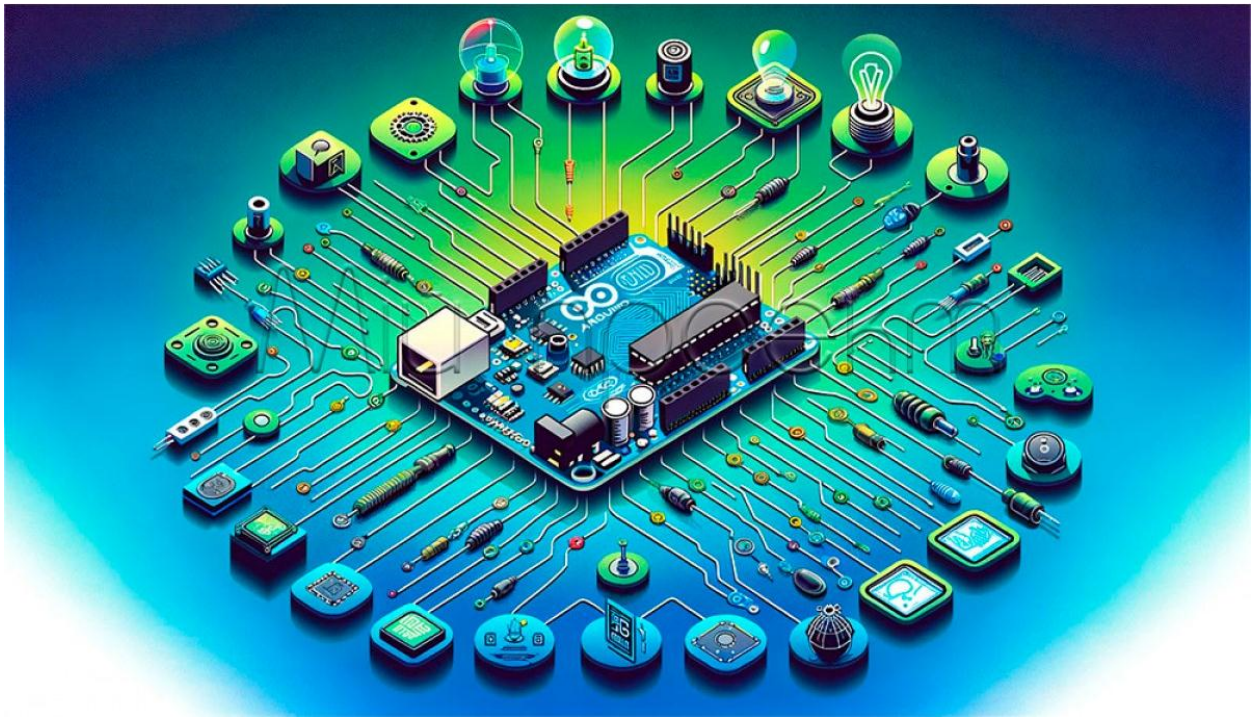
Практична значущість роботи полягає в тому, що розроблена модель маніпулятора на базі Arduino може бути використаний в практиці навчання учнів при вивченні технології у профільній школі.

1. ЗАСТОСУВАННЯ АПАРАТНО ОБЧИСЛЮВАЛЬНОЇ ПЛАТФОРМИ ARDUINO ДЛЯ РОБОТОТЕХНІЧНОГО ПРОЄКТУВАННЯ.

1.1 Огляд платформи Arduino. Основні характеристики і можливості.

Arduino – це електронна платформа з відкритим вихідним кодом, яка базується на простому у використанні апаратному та програмному забезпеченні. Плати Arduino здатні зчитувати різноманітні вхідні дані – датчик світла, натискання кнопки тощо, і перетворювати їх на вихідні дані – активація двигуна, вмикання освітлення та ін. Для цього використовується мова програмування Arduino (на основі Wiring) і програмне забезпечення Arduino (IDE) на основі Processing.[14]

Мал. 1.1.1



Протягом багатьох років Arduino був мозком тисяч проектів, від побутових пристроїв до складних наукових приладів. Навколо цієї платформи з відкритим вихідним кодом зібралося всесвітнє співтовариство творців – студентів, любителів, програмістів і професіоналів. їхній внесок

додав неймовірну кількість доступних знань, які можуть бути дуже корисними як новачкам, так і експертам.

Arduino бере свій початок в Ivrea Interaction Design Institute як простий інструмент для швидкого створення прототипів, орієнтований на студентів, які не мають досвіду в електроніці та програмуванні. Щойно плата Arduino охопила ширше коло користувачів, вона почала змінюватися, щоб адаптуватися до нових потреб і викликів, диференціюючи свою будову від простих 8-розрядних плат до продуктів з керуванням через Інтернет, різноманітних переносних пристроїв, 3D-друку та ін.

Мал.1.1.2

Завдяки простому та доступному користувацькому інтерфейсу Arduino використовувався в тисячах різних проектів і програм. Програмне забезпечення Arduino просте у використанні для початківців, але досить гнучке для досвідчених користувачів. Він працює на Mac, Windows і Linux. Викладачі та студенти використовують його, щоб створювати недорогі наукові прилади для вивчення хімії чи фізики або опанувувати програмування та робототехніку.

Дизайнери та архітектори створюють інтерактивні прототипи, музиканти та художники використовують його для інсталяцій та експериментів з новими музичними інструментами. Виробники, використовують його для створення багатьох проектів, представлених на виставці Maker Faire, наприклад. Arduino є ключовим інструментом для вивчення нового.

Існує багато мікроконтролерів і платформ мікроконтролерів, доступних для фізичних обчислень: Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handyboard та багато інших, які пропонують подібні функції. Усі ці інструменти занадто складні із заплутаним процесом програмування мікроконтролерів для недосвідчених користувачів. Arduino в свою чергу спрощує процес роботи з мікроконтролерами і пропонує деякі переваги для викладачів, студентів і зацікавлених аматорів перед іншими системами.

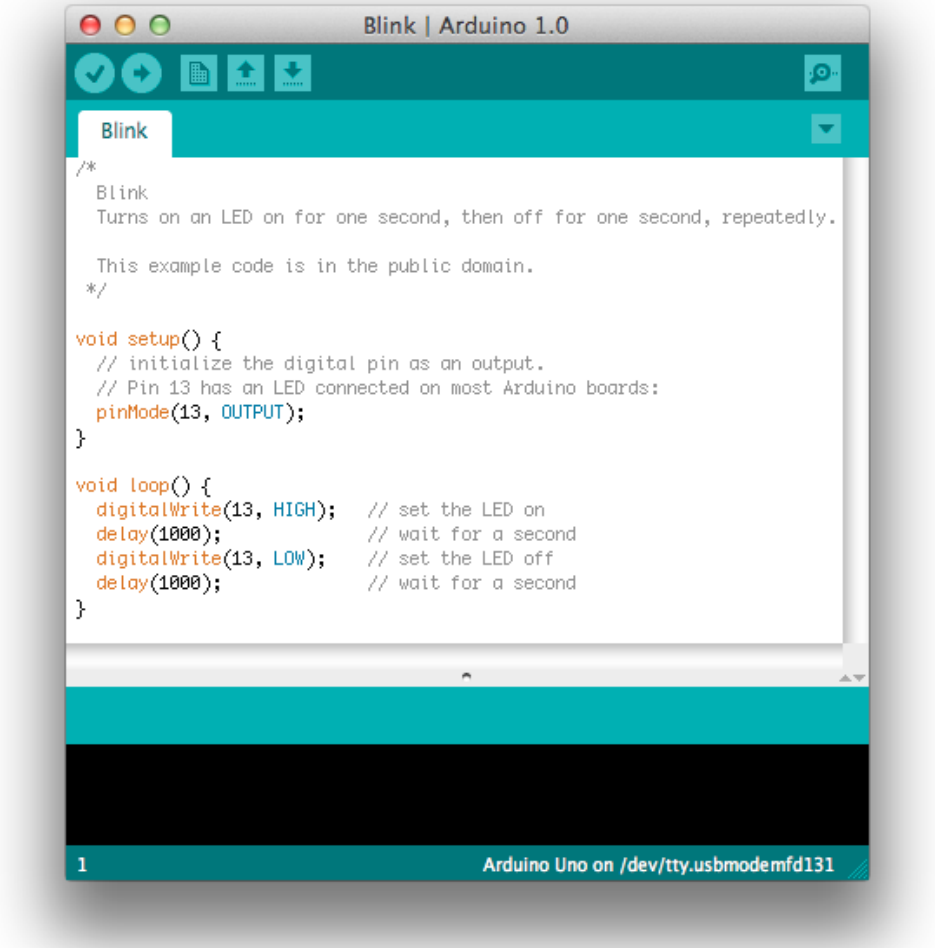
1. Відносно мала вартість – плати Arduino відносно недорогі порівняно з іншими платформами мікроконтролерів. Найдешевшу версію модуля Arduino можна зібрати вручну, і навіть попередньо зібрані модулі Arduino коштують менше $50.

2. Кросплатформенність – програмне забезпечення Arduino (IDE) працює в операційних системах Windows, Macintosh OSX і Linux. Більшість систем мікроконтролерів обмежені Windows.

3. Просте, зрозуміле середовище програмування. Програмне забезпечення Arduino (IDE) просте у використанні для початківців, але досить гнучке, щоб досвідчені користувачі також могли скористатися ним. Воно базується на середовищі програмування Processing, тому в учнів, які навчаються програмувати в цьому середовищі не виникне проблем з тим, як працює Arduino IDE.

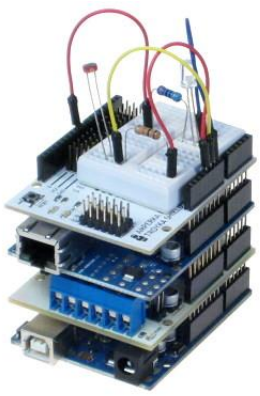
4. Програмне забезпечення з відкритим вихідним кодом. Програмне забезпечення Arduino надається з відкритим кодом і доступне для покращення і розширення досвідченими програмістами. Мову можна розширити за допомогою бібліотек C++, а також мови програмування AVR C, на якій вона заснована. Таким чином можна використовувати код AVR-C безпосередньо у програмах Arduino.

Мал. 1.1.3



5. Розширюване обладнання. Плани плат Arduino опубліковані за ліцензією Creative Commons, тому досвідчені розробники схем можуть створити власну версію модуля, розширивши його та покращивши. Навіть відносно недосвідчені користувачі можуть зібрати макетну версію модуля, щоб зрозуміти, як він працює та заощадити гроші. [14]

Мал.1.1.4



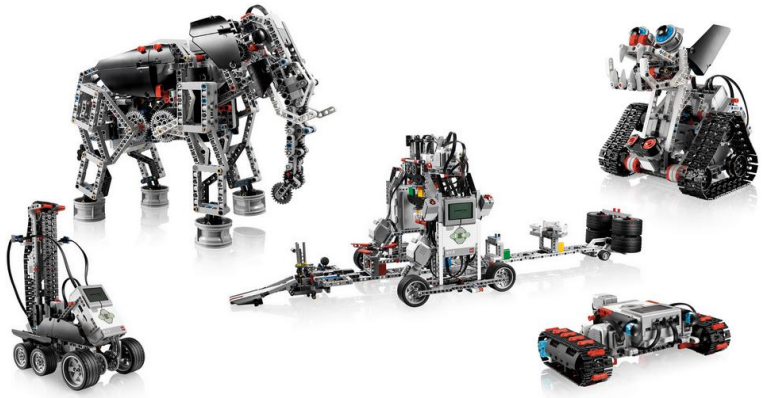
1.2 Можливості апаратно обчислювальної платформи Arduino для проєктування і побудови роботів у порівнянні з іншими платформами.

Сформулюємо основну задачу: підтримка інтересу учнів до світу ІТ в цілому і програмування, зокрема, закріпити і розвинути навики програмування. Додатково – познайомитися з новим розділом світу ІТ – робототехнікою.

Обзор ринку освітньої робототехніки. Розділемо варіанти на категорії:

1. Готові рішення на своїх контролерах – Lego Mindstorms EV3, VEX.

Мал. 1.2.1



Плюси: працює із коробки, є інструкції, можна збирати різних роботів. Найбільш відомі рішення з ними все відносно просто.

Мінуси: Ціна. І це головний мінус.

2. Проекти на Arduino.

Мал. 1.2.2

Має дуже багато переваг, тому що, це цілий світ DIY, але щоб опанувати апаратну частину потрібно гарно розумітися на в електричних схемах.

Плюси: невелика вартість, великий вибір комплектуючих.

Мінуси: готових проектів у продажу не зовсім небагато.

3. Raspberry Pi та аналоги, або Nvidia Jetson Nano +ROSi.

Мал. 1.2.3

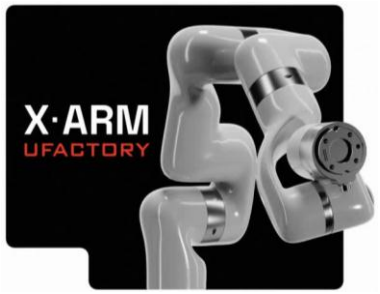
Можливості вже набагато ширші, є розпізнавання відеопотоку, керування великою кількістю пристроїв і дуже багато різних проектів.

Плюси: цілком стабільні в плані роботи і конструкції роботи, можливість серйозно програмувати.

Мінуси: поріг входу в плані необхідних знань тут значно вищий, значить і час, який буде потрібний для занурення в тему – в рази більше.

4. Спеціальні навчальні роботи «високого рівня». Ufactory, Dobot, Applied Robotics

Мал 1.2.4



Плюси: професійна, високоякісна робототехніка

Мінуси: Дуже велика вартість, немає посібників.

За підсумками аналізу зрозуміло: більш-менш якісний невеликий робот за невеликі гроші, без проблем зі складанням і з не надто складною електронікою це Arduino.

Arduino дозволить учням набути навички роботи з цією популярною у всьому світі платформою, а також створити за своїм задумом унікальні геджети, втіливши найнеймовірніші ідеї.

Arduino – це невелика плата з власним процесором та пам'яттю до якої яким можна підключати всілякі компоненти: лампочки, датчики, мотори, чайники, роутери, магнітні дверні замки та взагалі все, що працює від електрики. У процесор Arduino можна завантажити програму, яка керуватиме всіма цими пристроями за заданим алгоритмом. Ще однією відмінністю Arduino є наявність плат розширення, так званих shields або просто «шилдів».

Мал. 1.2.5

Це додаткові плати, які ставляться подібно шарам бутерброда поверх Arduino, щоб дати йому нові можливості.

2. РОЗРОБКА Й ВИГОТОВЛЕННЯ МОДЕЛІ МАНІПУЛЯТОРА НА БАЗІ ARDUINO

2.1 Різновиди і конструкція промислових маніпуляторів.

Промислові роботи це автоматизовані машини, які використовуються у виробничих галузях замість людей. Вони швидше, точніше і можуть працювати протягом багатьох днів. Промислові роботи можна розділити на 7 основних категорій, які представлені нижче.[15]

1. Шарнірні роботи.

Мал. 2.1.1

Шарнірні роботи є найбільш часто використовуваними роботами у виробництві, і вони характеризуються з'єднаннями, що обертаються – осями. Вони варіюються від простих 2-осьових шарнірів до складних з'єднань з більш як 10 осями, які приводяться в дію двигунами. Найбільш поширені шарнірні роботи, які зазвичай використовуються на фабриці, це 6-осьові роботи.

Завдяки своїм осям шарнірні роботи є найбільш гнучкими і тому використовуються в ролях, що вимагають інтенсивної роботи і багатозадачності. Ви знайдете їх в області зварювання, передачі деталей, складання, підбору та розміщення, упаковки та укладання на піддони.

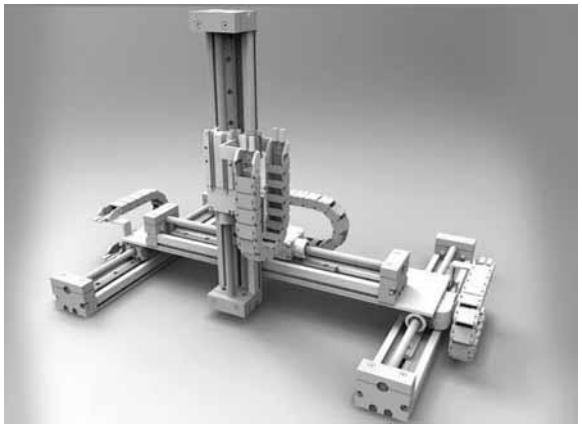
2. Циліндричні роботи.

Мал. 2.1.2

Це роботи, які мають один шарнір, що обертається, в основі і ще один призматичний шарнір, що з'єднує ланки. Цей поворотний шарнір використовує обертальний рух вздовж осі шарніра, а призматичний шарнір використовує лінійний рух. Однак те, що дає їм назву циліндричних – це форма їхньої оболонки, яка має форму циліндра. Деякі з завдань, які найкраще підходять для циліндричних роботів, включають роботизоване покриття та обслуговування машин.

3. Декартові роботи.

Мал. 2.1.3



Як випливає з назви, роботи роботи декартові працюють по лінійній системі рейок. Також відомі як портальні роботи, вони мають кубоподібну оболонку з трьома перпендикулярними осями X, Y та Z. Саме ці осі

визначають кожен спрямований рух робота під час операцій. Декартові роботи найкраще підходять для виконання функцій захоплення і розміщення, тому що вони можуть долати дуже великі відстані, залежно від довжини лінійного галсу. Більшість лінійних доріжок зазвичай знаходяться над головою, що також робить ці типи роботів найкращими у використанні простору.

4. СКАР роботи.

Мал. 2.1.4

SCAR Роботи є одними з передових промислових роботів. Вони найбільш відомі своєю швидкістю, тому їм відводяться складні ролі, такі як збирання складних деталей автомобілів та інших автомобільних пристроїв. Вони мають циліндричну оболонку, а також можуть працювати в площинах X, Y, Z у поєднанні з обертальним рухом. Інші ролі, з якими можуть впоратися роботи SCAR, включають завантаження машин, укладання на піддони та автоматичне упакування.

5. Спільні роботи.

Мал. 2.1.5

Спільні роботи були відповіддю, коли виникла потреба включити людську працю поряд з роботами. Вони розроблені з урахуванням кращих характеристик безпеки, що робить їх ідеальними для роботи поруч із людьми без будь-якої небезпеки нещасних випадків. Вони заокруглені та оснащені сучасними датчиками, які здатні визначати відстань, коли люди знаходяться поблизу.

6. Дельта роботи.

Мал. 2.1.6

Також званий Паралельні роботи, це автоматизовані промислові машини з паралельними шарнірними важелями, які відходять від загальної основи і звернені вниз. Має куполоподібну оболонку. Надзвичайно легкі руки роблять їх ідеальними роботами для завдань, що потребують великої кількості операцій з однаковою точністю. Це тип роботів, яких ви знайдете на заводі з виробництва електроніки, заводі з переробки харчових продуктів або на фармацевтичному заводі. Їх завдання включають вибір місця і передачу деталей.

7. Полярні роботи.

Мал. 2.1.7

Полярні роботыи були винайдені як перші індустріальні роботи. Більшість покращених роботів, які з'явилися після них, були приблизно засновані на кресленнях роботів Polar. Вони мають сферичну робочу оболонку та працюють у полярних координатах. Вони більше не використовуються широко на виробництві окрім місць, де вони були адаптовані для сучасних задач. Найчастіше їх використовують для вирішення таких завдань, як лиття під тиском і обробка матеріалів.

2.2 Розробка й виготовлення моделі маніпулятора на базі Arduino.

Для складання нам знадобиться кріплення:

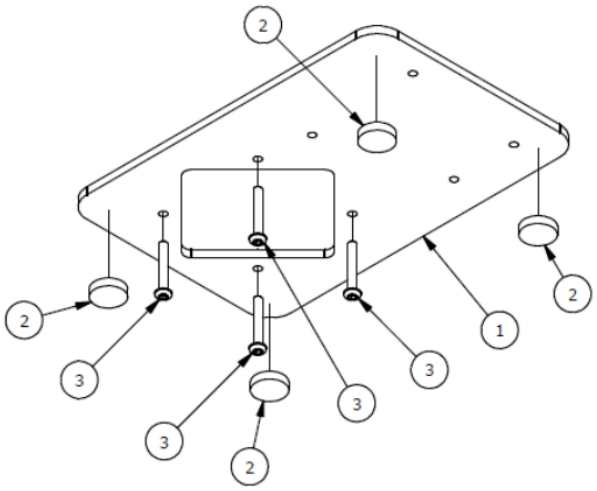
Номер позиції	Назва	Кількість,шт.
1	Гайка М3	10
2	Гвинт М3х6	9
3	Гвинт М3х8	10
4	Гвинт М3х10	5
5	Гвинт М3х12	7
6	Гвинт М3х20	4

Крок 1.

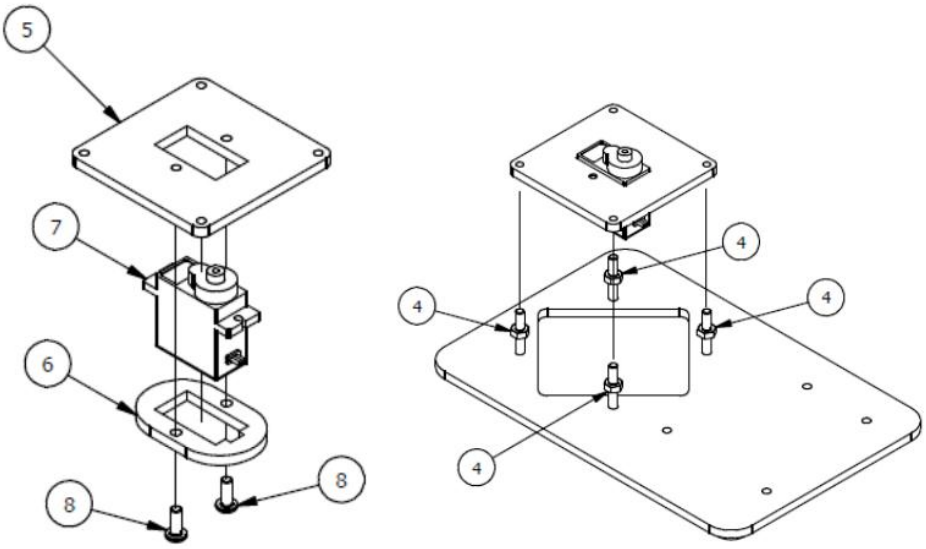
Перелік деталей:

Номер позиції	Кількість	Назва
1	1	Підстава
3	4	М3х20мм гвинт
4	4	М3 гайка
5	1	Опорна пластина
6	1	Кріплення
7	1	Сервопривід
8	2	М3х8мм гвинт

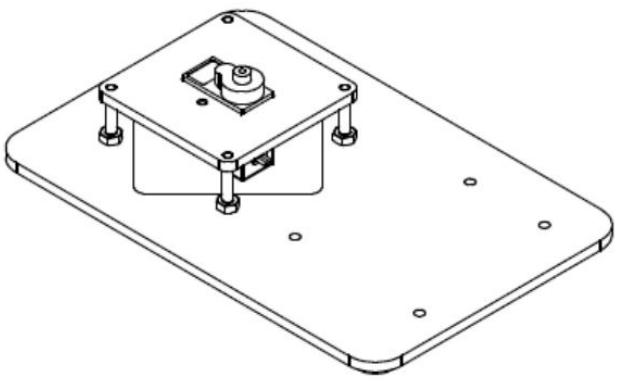
1.



2.



3.

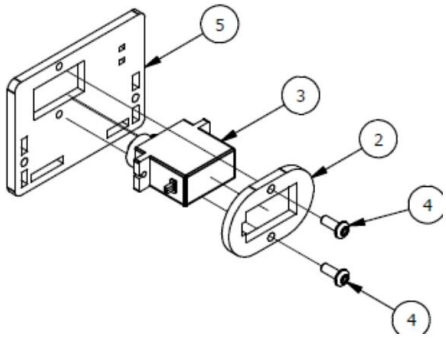


Крок 2.

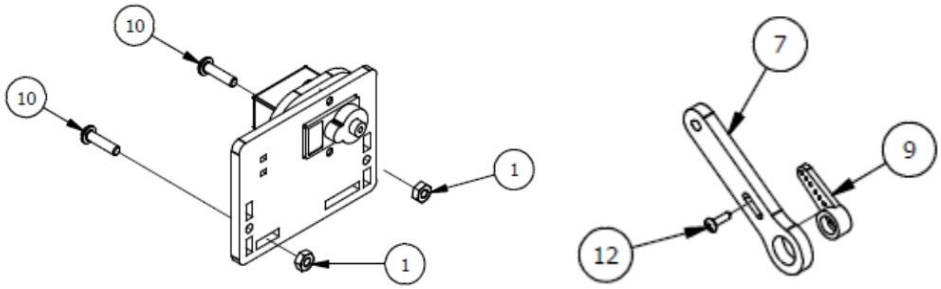
Перелік деталей:

Номер позиції	Кількість	Назва
1	2	М3 гайка
2	1	Кріплення
3	1	Сервопривід
4	2	М3х8гвинт
5	1	Основа лівої руки
6	1	Паралельне кріплення
7	1	Важіль руки
8	1	М3х6мм гвинт
9	1	Серво важіль
10	2	М3х12мм гвинт
11	1	Осьовий серверний гвинт
12	1	Фіксуючий серверний гвинт

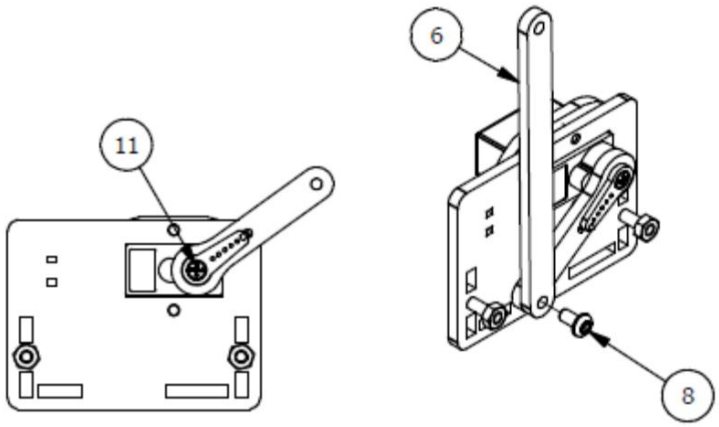
1.



2.



3.

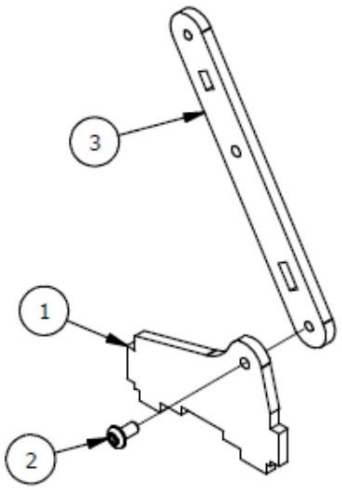


Крок 3.

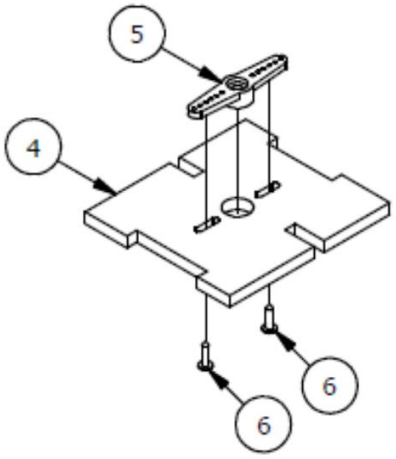
Перелік деталей:

Номер позиції	Кількість	Назва
1	1	Кріплення вкладки лівоїруки
2	1	М3х6мм гвинт
3	1	Балка лівоїруки
4	1	Верхня кришка
5	1	Подвійний сервоважіль
6	2	Фіксуючий серверниййгвинт.

1.



2.

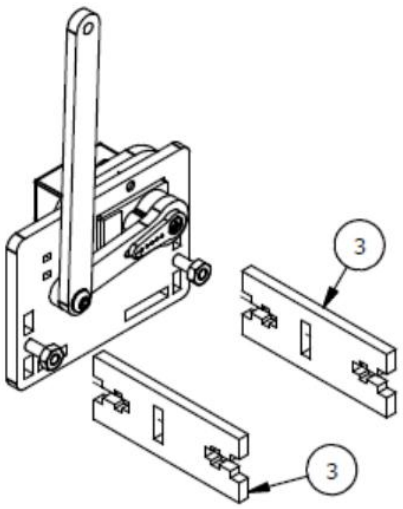


Крок 4.

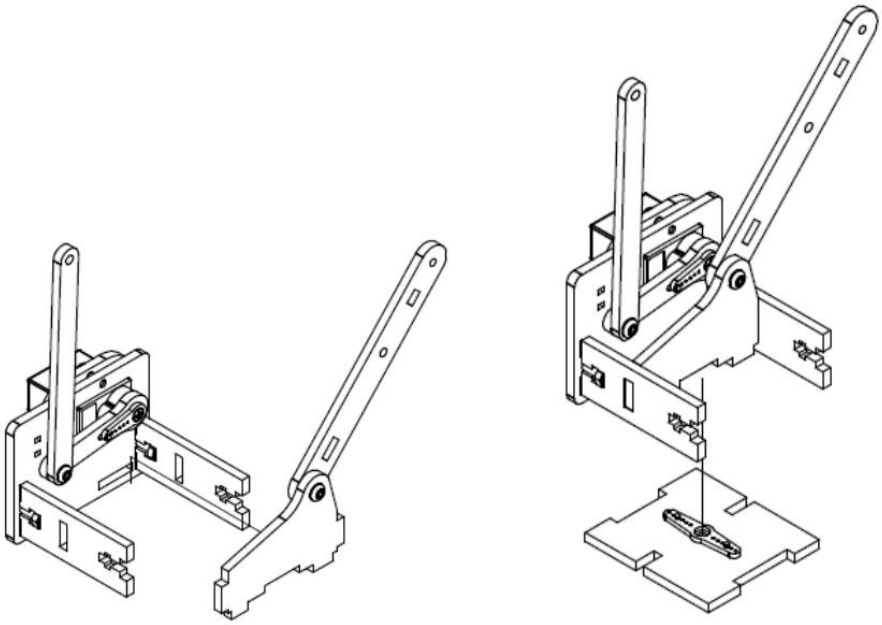
Перелік деталей:

Номер позиції	Кількість	Назва
1	2	М3 гайка
2	2	М3х12мм гвинт
3	1	Траверса основи маніпулятора
4	1	Сполучне ребро жорсткості

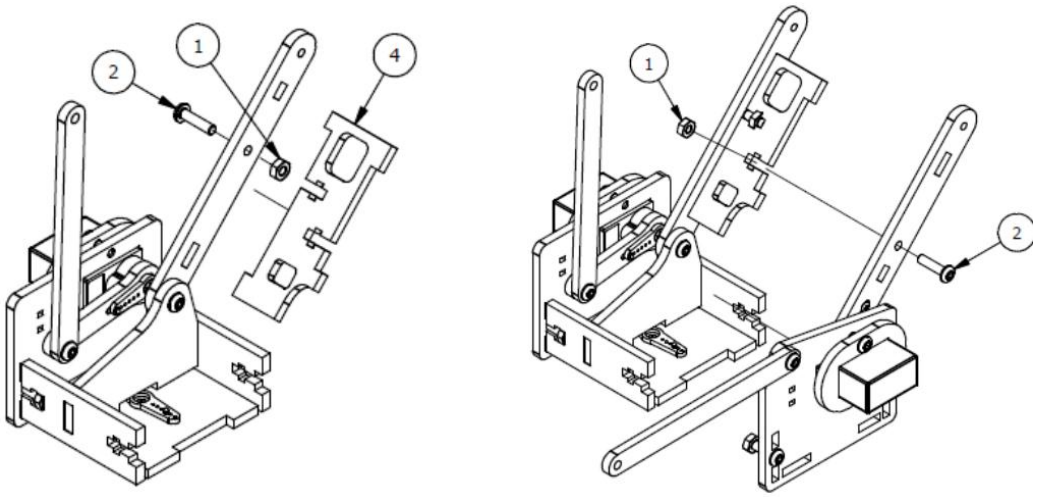
1.



2.



3.

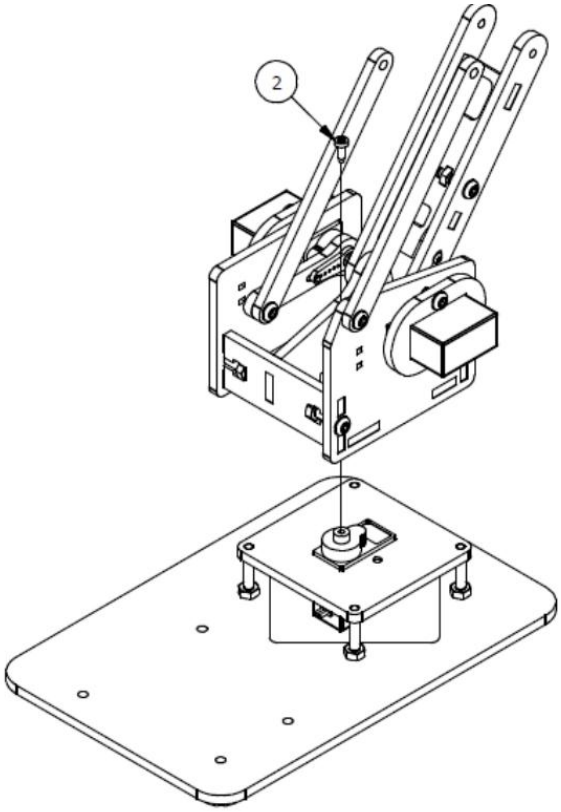


Крок 5.

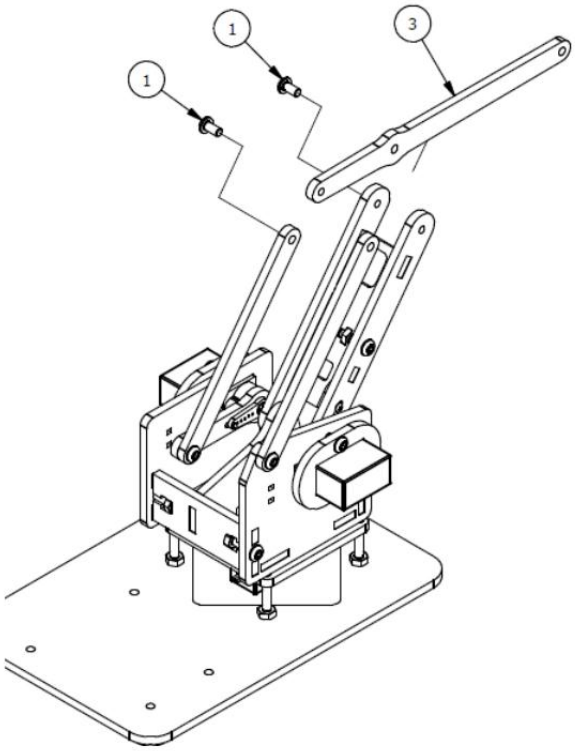
Перелік деталей:

Номер позиції	Кількість	Назва
1	2	М3х6мм гвинт
2	1	Фіксуючий серверний гвинт
3	1	Балка лівого зап'ястя

1.



2.

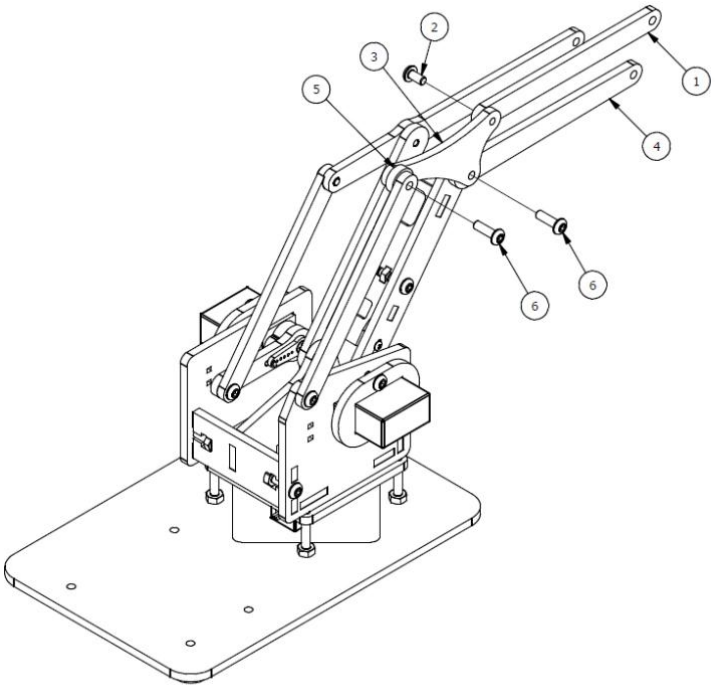


Крок 6.

Перелік деталей:

Номер позиції	Кількість	Назва
1	1	Паралельна балка
2	1	М3х6мм гвинт
3	1	Конектор
4	1	Балка правого зап'ястя
5	1	Прокладка
6	2	М3х10 гвинт

1.



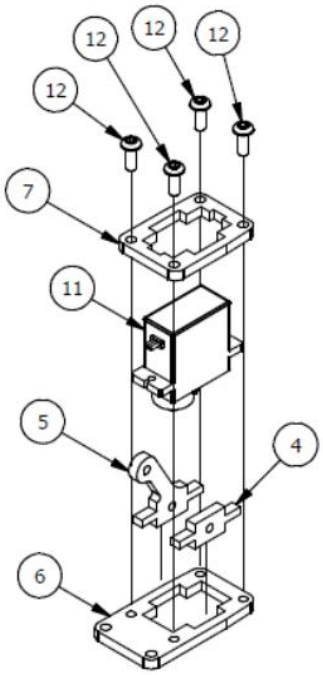
Крок 7.

Перелік деталей:

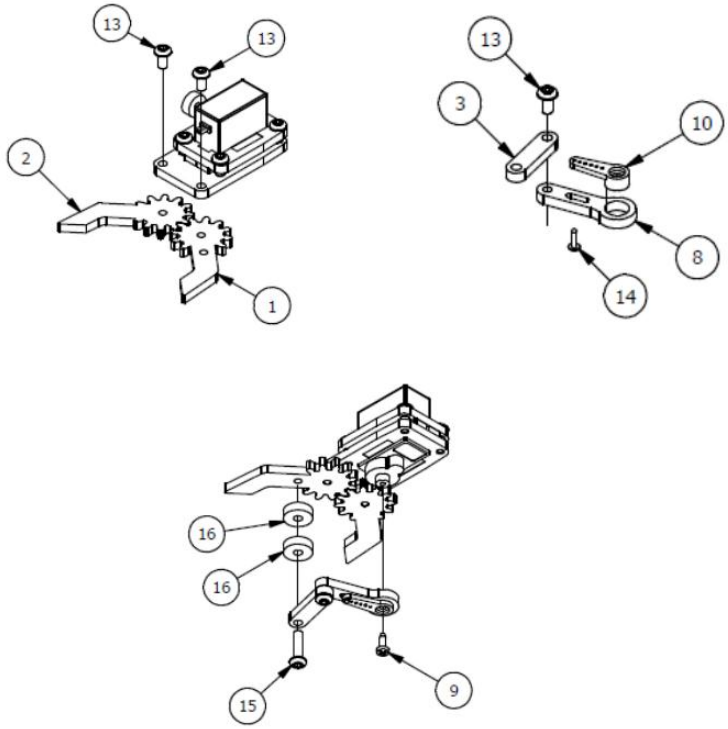
Номер позиції	Кількість	Назва
1	1	Ліве захоплення
2	1	Праве захоплення
3	1	Привідний важіль

4	1	Ліве кріплення зап'ястя
5	1	Праве кріплення зап'ястя
6	1	Нижнє кріплення сервоприводу
7	1	Верхнє кріплення сервоприводу.
8	1	Привідний важіль
9	1	Осьовий серверний гвинт.
10	1	Сервоважіль
11	1	Сервопривід
12	4	М3х8мм
13	3	М3х6мм
14	1	Фіксуючий серверний гвинт
15	1	М3х12мм гвинт
16	2	Прокладка

1.



2.

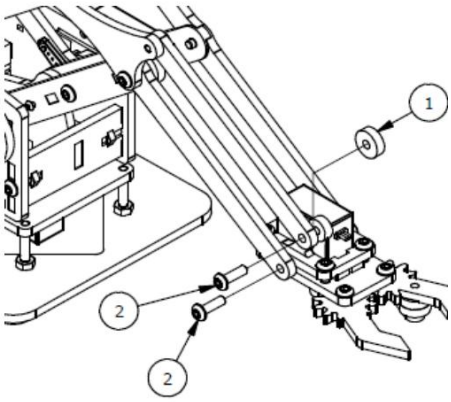


Крок 8.

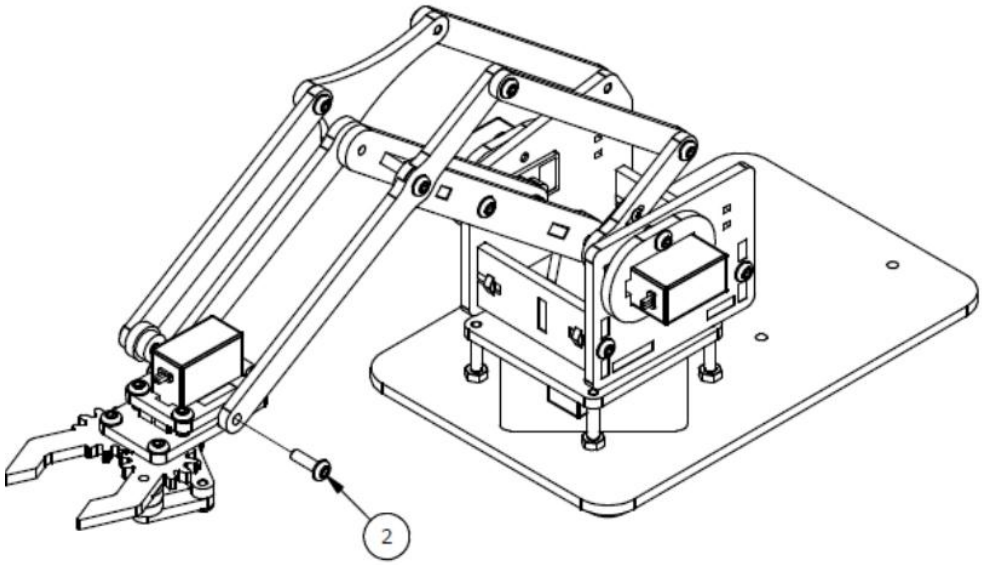
Перелік деталей:

Номер позиції	Кількість	Назва
1	1	Прокладка
2	3	М3х10мм гвинт

1.



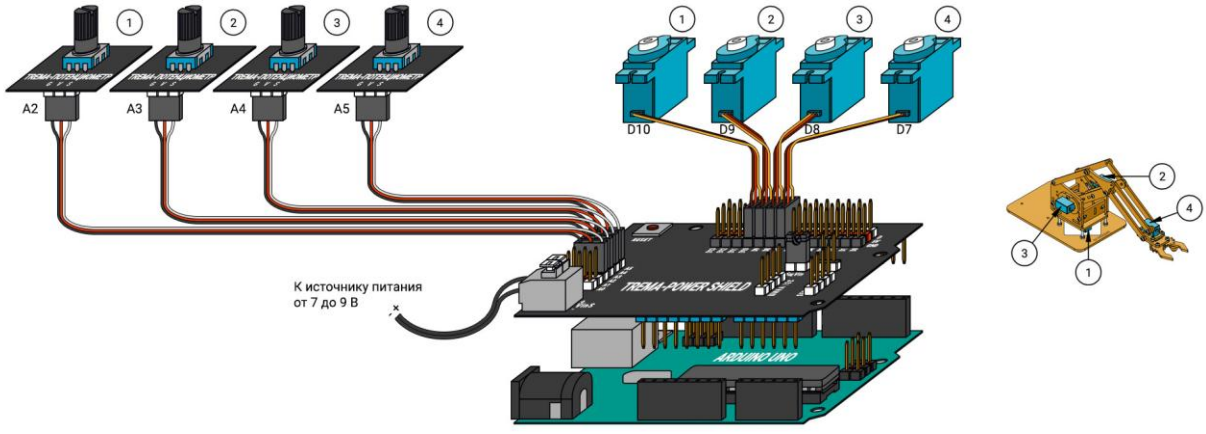
2.



2.3 Підключення і керування маніпулятором на базі Arduino.

Ми пропонуємо підключити сервоприводи маніпулятора Arduino UNO через Trerma-Power Shield, а керувати сервоприводами використовуючи Trema-потенціометри.

Мал. 2.3.1



1. Сервоприводи підключаються до виводів D10-D7 на Trerma-Power Shield (можна міняти у скетчі).

2. Trema-потенціометри підключаються до виводів A2-A5 на Trerma-Power Shield (можна міняти у скетчі).

3. Джампер (перемичка) вибору схеми живлення на платі Trerma-Power Shield встановлюється положення «Заг.Vin».

4. Джерело живлення на 7 - 12 В постійного струму підключається до клемника на Trerma-Power Shield.

5. Підключати живлення до Arduino UNO не потрібно, тому що на Trerma-Power Shield встановлений джампер "Общ.Vin", отже, на вхід Vin Arduino UNO подається живлення з входу "VinS" Trerma-Power Shield.

Живлення:

Trerma-Power Shield має можливість вибору схеми живлення установкою джампера (перемички) на його платі в одне з двох положень: «Заг.Vin» або «Заг.5V». Маніпулятор буде працювати за будь-якої схеми підключення живлення:

1. Якщо у Вас є джерело живлення на 7 - 12 В з проводом на кінці якого є штекер, його потрібно підключити до Arduino UNO і встановити джампер в положення «Заг.Vin».

2. Якщо у Вас є джерело живлення на 7 - 12 В з проводом без штекера, його потрібно підключити до конектора Trerma-Power Shield і встановити джампер в положення «Заг.Vin».

3. Якщо у Вас є джерело живлення на 7 - 30 В з проводом без штекера, його потрібно підключити до конектора Trerma-Power Shield і встановити джампер в положення «Заг.5V».

У наведеному вище прикладі підключення, використовується друга схема вибору живлення: джерело живлення на 7 - 9 підключений до клемника Trerma-Power Shield, а джампер (перемичка) встановлений в положення «Заг.Vin».

Робота маніпулятора:

Алгоритм програми (скетчу) простий: поворот ручки Trema-потенціометра приводить у рух сервопривід.

- Поворот ручки першого Trema-потенціометра призведе до повороту основи.
- Поворот ручки другого Trema-потенціометра призведе до повороту лівого плеча.
- Поворот ручки третього потенціометра Trema призведе до повороту правого плеча.
- Поворот ручки четвертого Trema-потенціометра приведе в рух захват.

У коді програми (скетчі) передбачено захист сервоприводів, який полягає в тому, що діапазон їх обертання обмежений інтервалом (двома кутами) вільного ходу. Мінімальний та максимальний кут обертання вказуються як два останні аргументи функції map() для кожного сервоприводу. А значення цих кутів визначається процесі калібрування, яку потрібно виконати до початку роботи з маніпулятором.

Код програми (Додаток Б):

Якщо ви подаєте живлення, до калібрування, маніпулятор може почати рухатися неадекватно. Спершу необхідно виконати всі кроки калібрування.

Мал. 2.3.2

```
#include <Servo.h>                                                    // Подключаем
Servo servo1;                                                         // Объявляем о
Servo servo2;                                                         // Объявляем о
Servo servo3;                                                         // Объявляем о
Servo servo4;                                                         // Объявляем о
int valR1, valR2, valR3, valR4;                                       // Объявляем п
                                                                      // Назначаем в
const uint8_t pinR1 = A2;                                             // Определяем
const uint8_t pinR2 = A3;                                             // Определяем
const uint8_t pinR3 = A4;                                             // Определяем
const uint8_t pinR4 = A5;                                             // Определяем
const uint8_t pinS1 = 10;                                             // Определяем
const uint8_t pinS2 = 9;                                              // Определяем
const uint8_t pinS3 = 8;                                              // Определяем
const uint8_t pinS4 = 7;                                              // Определяем
void setup(){                                                         // Код функции
    Serial.begin(9600);                                               // Инициируем
    servo1.attach(pinS1);                                             // Назначаем о
    servo2.attach(pinS2);                                             // Назначаем о
    servo3.attach(pinS3);                                             // Назначаем о
    servo4.attach(pinS4);                                             // Назначаем о
}
void loop(){                                                          // Код функции
    valR1=map(analogRead(pinR1), 0, 1024, 10, 170); servo1.write(valR1); // Вращаем осн
    valR2=map(analogRead(pinR2), 0, 1024, 80, 170); servo2.write(valR2); // Управляем л
    valR3=map(analogRead(pinR3), 0, 1024, 60, 170); servo3.write(valR3); // Управляем п
    valR4=map(analogRead(pinR4), 0, 1024, 40,  70); servo4.write(valR4); // Управляем з
    Serial.println((String) "A1 = "+valR1+",\t A2 = "+valR2+", \t A3 = "+valR3+", \t A4
}
```

Калібрування.

Перед початком роботи з маніпулятором його потрібно відкалібрувати.

Калібрування полягає у вказівці крайніх значень кута повороту для кожного сервоприводу, щоб деталі не заважали їх рухам. Порядок калібрування наступний:

1. Від'єднайте всі сервоприводи від Trema-Power Shield, завантажте скетч та підключіть живлення.

2. Відкрийте послідовний монітор.

3. У моніторі відображатимуться кути повороту кожного сервоприводу (у градусах).

4. Підключіть перший сервопривід (керуючий обертанням основи) до виводу D10.

5. Поворот ручки першого Trema-потенціометра (виведення A2) призведе до повороту першого сервоприводу (вивід D10), а в моніторі зміниться значення поточного кута цього сервоприводу (значення: A1 = ...). Останні положення першого сервоприводу будуть лежати в діапазоні, від 10 до 170 градусів (як написано в першому рядку коду loop). Цей діапазон можна змінити, замінивши значення останніх двох аргументів функції map() у першому рядку коду loop на нові. Наприклад, замінивши 170 на 180, Ви збільшите крайнє положення сервоприводу у цьому напрямку. А замінивши 10 на 20, Ви зменшите інше крайнє положення того ж сервоприводу.

6. Якщо Ви замінили значення, потрібно заново завантажити скетч. Тепер сервопривід повертатиметься в нових, заданих Вами межах.

7. Підключіть другий сервопривід (керуючий поворотом лівого плеча) до виводу D9.

8. Поворот ручки другого Trema-потенціометра (вивід A3) призведе до повороту другого сервоприводу (вивід D9), а в моніторі зміниться значення поточного кута цього сервоприводу (значення: A2 = ...). Останні положення

другого сервоприводу будуть лежати в діапазоні, від 80 до 170 градусів (як написано в другому рядку коду loop скетчу). Цей діапазон змінюється так само, як і для першого сервоприводу.

9. Якщо Ви замінили значення, потрібно заново завантажити скетч.

10. Підключіть третій сервопривід (керуючий поворотом правого плеча) до виводу D8. і аналогічно здійсніть його калібрування.

11. Підключіть четвертий сервопривід (керуючий захопленням) до виводу D7. і аналогічно здійсніть його калібрування.

Калібрування достатньо виконати 1 раз, після складання маніпулятора.

3. МЕТОДИКА ВИКОРИСТАННЯ МОДЕЛІ МАНІПУЛЯТОРА НА БАЗІ ARDUINO ПІД ЧАС ВИВЧЕННЯ ТЕХНОЛОГІЇ В ПРОФІЛЬНІЙ ШКОЛІ

3.1 Дидактичні умови вивчення робототехніки на уроках технології.

Вивчення робототехніки у профільній школі перш за все передбачає зміни матеріально-технічної бази та особливу підготовку педагогів. Поступово багато шкіл ретельно обладнують кабінети технологій для занять робототехнікою. Це інтегрована дисципліна, взаємозв'язана з іншими предметами: математикою, фізикою, інформатикою. Тому використовувати навчальні посібники можна відразу для кількох занять.

Важним завданням успішного проведення занять є підготовка вчителя. Сучасним педагогам належить вивчати нові технології, уміти працювати з 3D моделюванням, займатися прототипуванням, електронікою та ін. З метою підвищення кваліфікації для вчителів організовуються профільні курси та готуються спеціальні навчальні матеріали з методичними рекомендаціями.

У школі робототехніку можна починаєть вивчати у 5 класі, але це не означає, що дітям одразу видають набори та просять зібрати свого першого робота. В основі всього – фундаментальна теоретична підготовка. Діти вивчають:

– Конструювання. Учні докладно вивчають, як працювати зі схемами, внутрішню будову робота, взаємодія його елементів між собою.

– Електротехніку. У цьому блоці вчать працювати з електронікою, з'єднувати мікросхеми тощо.

– Програмування. Ази програмування діти отримують під час уроків інформатики. Тут розвивають логіку, вибудовують алгоритми, вивчають конкретні мови програмування.

– Проєктну діяльність. Робототехніка має на увазі комплексну, проектну, роботу. Учні ставлять цілі, розробляють етапи її досягнення і

втілюють на практиці. Крім іншого, цей метод добре зміцнює здатність вибудовувати причинно-наслідкові зв'язки.

– Роботу у команді. Проектна діяльність завершується презентацією. Це спонукає дітей взаємодії один з одним, відточує навички громадського виступу, вчить відстоювати свою думку.

Звичайно, на заняттях дітям знадобляться знання із суміжних областей, творчі здібності та посидючість.

Окрім того, вивчення робототехніки в школі спрощує дітям вступ до профільних ЗВО: вони приходять більш підготовленими. Випускники вищих навчальних закладів у свою чергу не залишаються без роботи. Сьогодні попит на кваліфіковані кадри у галузі робототехніки перевищує пропозицію. Зокрема, діти можуть опанувати такі професії як:

– проєктування роботів та автономних систем;

– моделювання та прототипування;

– створення нового ПЗ;

– діагностика та усунення несправностей у системі;

– навчання персоналу роботі з інноваційними технологіями та багато іншого.

Під час вивчення робототехніки діти можуть також брати участь у різноманітних тематичних змаганнях. Рівень різноманітний: від міських до міжнародних. Змагання чудово доповнюють класичний варіант навчання та ще більше мотивують школярів продовжувати освоювати робототехніку.

3.2 Методика використання моделі маніпулятора на базі Arduino у профільній школі.

План-конспект уроку на **тему** «Вивчення основних бібліотек платформи Arduino»

Навчальна мета: знати поняття бібліотек, їх функцій. Розуміти структуру програми. Застосовувати навички роботи – використовувати гарячі клавіші, працювати з документами, ключові точки.

Розвиваюча мета: розкрити творчий потенціал учня, розвинути пам'ять та мислення.

Виховна мета: розвинути розуміння та комп'ютерну грамотність учня.

Знання, які необхідно актуалізувати під час заняття: Комп'ютерна графіка, мультимедіа, знання мови програмування.

Тип заняття: Комбінований.

Необхідні наочні посібники та технічні засоби: комп'ютер, проектор, мадель маніпулятора на базі Arduino.

Порядок проведення заняття:

1.Підготовка учнів до виконання практичної роботи. (5хв)

2. Інструктаж. З прикладом виконання роботи (25хв).

3. Виконання практичного завдання (40 хв)

4. Перевірка практичного завдання (20хв)

Хід заняття

1.Підготовка учнів до виконання практичної роботи.

На цьому етапі проводиться перекличка, розподіляються робочі місця за комп'ютерами між учнями. Педагогом проводиться оцінка знань учнів по темі, щоб підібрати оптимальний темп навчання.

2. Інструктаж. З прикладом виконання.

На даному етапі ставляться цілі та завдання практичної роботи, викладач показує бібліотеки, розповідає їх зміст та корисність, для чого вони потрібні та наводить приклади.

1. Бібліотека Servo.

Ця бібліотека функцій Arduino контролера надає набір функцій для керування сервоприводами. Стандартні сервоприводи дозволяють повертати

привід на певний кут від 0 до 180 градусів. Деякі сервоприводи дозволяють здійснювати повні оберти заданої швидкості. Бібліотека Servo дозволяє одночасно керувати 12 сервоприводами на більшості плат Arduino і 48 на Arduino Mega. На контролерах, відмінних від Mega, використання бібліотеки відключає можливість використовувати виходи 9 і 10 в режимі ШІМ навіть якщо привід не підключений до цих виводів. На платі Mega можуть бути використані до 12 сервоприводів без втрати функціоналу ШІМ. При використанні Mega для керування від 12 до 23 сервоприводів не можна буде використовувати виходи 11 та 12 для ШІМ.

Функції:

1) attach ()

2) write ()

3) writeMicroseconds ()

4) read ()

5) attached ()

6) detach ()

У загальному випадку сервопривід підключається трьома проводами: живлення, земля та сигнальний. Зазвичай живлення – червоний провід і може бути підключений до виведення +5V на платі Arduino. Чорний дріт земля підключається до GND виводу Arduino, сигнальний, зазвичай жовтий, провід підключається до цифрового виводу котроллера Arduino. Слід зазначити, що потужні сервоприводи можуть створювати велике навантаження, в цьому випадку він повинен бути окремо (не через вихід +5V Arduino). Теж саме правильно для випадку підключення відразу кількох сервоприводів.

2. Бібліотека EERPOM.

Мікроконтролери ATmega мають свою енергонезалежну пам'ять, тобто користувачі Ардуїно мають можливість зберігати дані в цій пам'яті і вони можуть бути використані після вимкнення-вмикання або перезавантаження

контролера. Arduino бібліотека EERPOM надає зручний та простий інтерфейс роботи з цією пам'яттю. Різні моделі мікроконтролерів відрізняються обсягом EERPOM пам'яті, так ATMega328 має 1024 байт, 512 байт у ATmega168 та ATmega8 і по 4Кб (4096 байт) у ATmega1280 та ATmega2560.

Функції:

1) read ()

2) write ()

3. Бібліотека SPI.

Бібліотека SPI дозволяє контролеру Arduino взаємодіяти з пристроями, які підтримують SPI протокол. Arduino в даному випадку виступає як провідний пристрій. Послідовний периферійний інтерфейс (SPI) – це послідовний синхронний протокол передачі даних, що використовується мікроконтролерами для обміну даними з одним або декількома периферійними пристроями на невеликих відстанях. Для організації з'єднання SPI необхідний один провідний пристрій, зазвичай це мікроконтролер, який управляє з'єднанням з веденими пристроями. Зазвичай підключення здійснюється трьома загальними лініями та лінією вибору периферійного (відомого) пристрою:

1) Master In Slave Out (MISO), перекладається як «вхід ведучого вихід веденого», використовується передачі даних від веденого до ведучого.

2) Master Out Slave In (MOSI) – вихід ведучого вхід веденого, передачі даних від ведучого до периферійних пристроїв.

3) Serial Clock (SCK) – синхронізуюча лінія, синхросигнал генерується провідним пристроєм.

4) Slave Select pin – вхід на ведених пристроях за допомогою якого ведучий може ініціювати обмін даними з периферійним пристроєм. Якщо цьому вході LOW, то ведений взаємодіє з провідним, якщо HIGH, то ведений ігнорує сигнали від ведучого.

При роботі з SPI пристроями слід враховувати такі моменти:

1) Який порядок виведення даних використовується: Most Significant Bit (MSB – старший біт (розряд)) або Least Significant Bit (LSB – молодший біт) перший. Порядок може змінюватися функцією SPI. setBitOrder().

2) Рівень сигналу синхронізації – за яким синхронізуючим сигналом (HIGH або LOW) передаються дані.

3) Фаза синхронізації – впливає на послідовність встановлення та вибірки даних. Фаза синхронізації SPI та рівень сигналу визначається функцією SPI. setDataMode().

4) Швидкість, на якій працює SPI, встановлюється функцією SPI. setClockDivider().

Виробники SPI пристроїв по-різному реалізують протокол, тому необхідно уважно ознайомитися з технічним описом до пристрою. Комбінація фази синхронізації (CPHA) та рівня сигналу синхронізації (CPOL) задають режим логіки роботи інтерфейсу SPI. Режим встановлюється SPI. setDataMode(). На контролерах Arduino Duemilanove та інших на базі ATmega168/328, шина SPI використовує виходи 10 (SS), 11 (MOSI), 12 (MISO) та 13 (SCK). На Arduino Mega – 50 (MISO), 51 (MOSI), 52 (SCK), та 53 (SS). Зверніть увагу, що навіть якщо ви не використовуєте вихід SS, він повинен бути встановлений як вихід, інакше інтерфейс може опинитися в режимі керування і бібліотека не буде працювати як слід. Як SS виходу може бути використаний відмінний вихід від 10-го. Наприклад, при роботі з Arduino Ethernet shield контролер використовує вихід 4 для взаємодії з картою SD по SPI і вихід 10 для роботи з Ethernet контролером.

Функції:

1) begin ()

2) end ()

3) setBitOrder ()

4) setClockDivider ()

5) setDataMode ()

6) transfer ()

4. Бібліотека Stepper

Бібліотека Stepper надає зручний інтерфейс керування біполярними та уніполярними кроковими двигунами. Для управління кроковим двигуном, залежно від його типу.

ВИСНОВКИ

Виконана кваліфікаційна робота передбачала розробку моделі маніпулятора на базі Arduino та методики її використання під час вивчення технології в профільній школі. За її результатами можна зробити наступні висновки:

1. Досліджено можливості застосування апаратно обчислювальної платформи Arduino для робототехнічного роєктування. Arduino – це електронна платформа з відкритим вихідним кодом, яка базується на простому у використанні апаратному та програмному забезпеченні. Плати Arduino здатні зчитувати різноманітні вхідні дані –датчик світла, натискання кнопки тощо, і перетворювати їх на вихідні дані – активація двигуна, вмикання освітлення та ін. Для цього використовується мова програмування Arduino (на основі Wiring) і програмне забезпечення Arduino (IDE) на основі Processing.

2. Розроблено й виготовлено модель маніпулятора на базі Arduino. За основу взято шарніпний робот. Шарнірні роботи є найбільш часто використовуваними роботами у виробництві, і вони характеризуються з'єднаннями, що обертаються – осями. Вони варіюються від простих 2-осьових шарнірів до складних з'єднань з більш як 10 осями, які приводяться в дію двигунами. Найбільш поширені шарнірні роботи це 6-осьові роботи.

Завдяки своїм осям шарнірні роботи є найбільш гнучкими і тому використовуються в ролях, що вимагають інтенсивної роботи і багатозадачності.

3. Описана методика використання моделі маніпулятора на базі Arduino під час вивчення технології в профільній школі. Вивчення робототехніки у профільній школі перш за все передбачає зміни матеріально-технічної бази та особливу підготовку педагогів. Поступово багато шкіл ретельно обладнують

кабінети технологій для занять робототехнікою. Це інтегрована дисципліна, взаємозв'язана з іншими предметами: математикою, фізикою, інформатикою. Тому використовувати навчальні посібники можна відразу для кількох занять. Розроблено план-конспект уроку на тему «Вивчення основних бібліотек платформи Arduino»

Таким чином, використання моделі маніпулятора на базі Arduino під час вивчення технології в профільній школі може стати ефективним інструментом для розвитку потенціалу дітей у різних галузях та сприяти створенню умов для їхнього творчого та політехнічного розвитку.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бондар В. І. Дидактика: навч. посіб. Київ: Либідь, 2005. 264 с.
2. Биков В. Ю. Інформаційні технології і засоби навчання. Київ: Атіка, 2008. 684 с.
3. Вакуленко І. В. Управління самостійною роботою студентів з використанням інформаційно-комунікаційних технологій. *Науковий часопис Національного педагогічного університету імені М. П. Драгоманова. Серія 2: Комп'ютерно-орієнтовані системи навчання.* Київ. 2016. Вип. 18 (25). С. 50-64.
4. Вікіпедія. Промисловий робот. [Електронний ресурс]. Wikipedia. Режим доступу: https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%BC%D0%B8%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D0%B9_%D1%80%D0%BE%D0%B1%D0%BE%D1%82 (дата звернення: 20.11.2024)
5. ДСТУ 2879-94 Маніпулятори, автооператори, роботи промислові та системи виробничі гнучкі. Терміни та визначення.
6. Лещук С. О. Навчально-інформаційне середовище як засіб організації пізнавальної діяльності учнів. *Науковий часопис Національного педагогічного університету ім. М. П. Драгоманова. Серія 2: Комп'ютерно-орієнтовні системи навчання: До 170-річного ювілею.* Київ: НПУ, 2004. С. 305-313.
7. Ортинський В. Л. Педагогіка вищої школи: навч. посіб. для студ. вищ. навч. закл. Київ: Центр учбової літератури, 2009. 472 с.
8. Остапчук С.А. До проблеми використання платформи Arduino у вивченні робототехніки. Наукові записки ЦДПУ. Серія: Педагогічні науки. Кропивницький: РВВ ЦДПУ ім. В. Винниченка, 2018. Вип. 168. С. 178-181.

9. Петрицин І. О. Застосування комп’ютерного моделювання у процесі електротехнічної підготовки майбутнього вчителя технологій. *Молодь і ринок.* 2017. № 1. С. 60-64.
10. Плата Arduino Uno R3: схема, опис, підключення пристроїв [Електронний ресурс]. ArduinoMaster. Режим доступу: arduinomaster.ru/platy-arduino/plata-arduino-uno/#__Arduino_Uno_R3(дата звернення: 20.11.2024).
11. Програмування Arduino [Електронний ресурс]. Arduino. Режим доступу: arduino.com/Reference (дата звернення: 20.11.2024).
12. Промислові роботи. Кафедра автоматизації технологічних процесів і виробництв Тернопільского національний технічний університет. [Електронний ресурс]. Kaf-av. Режим доступу: https://kaf-av.tntu.edu.ua/index.php/mn-abiturient/mn-articles/676-art-industrial-robots (дата звернення: 20.11.2024).
13. Промислові маніпулятори. [Електронний ресурс] Tlmanipulator. Режим доступу: https://uk.tlmanipulator.com/news/how-much-do-you-know-about-industrial-manipulators/ (дата звернення: 20.11.2024).
14. What is Arduino. [Електронний ресурс]. Arduino. Режим доступу: https://www.arduino.cc/en/Guide/Introduction (дата звернення: 20.11.2024).
15. 7 кращих промислових роботів. [Електронний ресурс] Evsint. Режим доступу: https://www.evsint.com/ru/top-7-industrial-robots/ (дата звернення: 20.11.2024).
16. Michael Margolis. Arduino Cookbook: Recipes to Begin, Expand, and Enhance Your Projects. 2020. 3rd Edition. P. 795
17. David McGriffy. Make: Drones: Teach an Arduino to Fly. 2016. P. 220
18. J. Hughes. Arduino: A Technical Reference: A Handbook for Technicians, Engineers, and Makers. 2016. P. 636

ДОДАТКИ

Додаток А

Розроблена модель маніпулятора на базі Arduino

1.

2.

3.

4.

5. 

Додаток Б

Код програми для роботи маніпулятора на базі Arduino

```
#include <Servo.h>
int pinLB=14;     // Define a 14 Pin
int pinLF=15;     // Define a 15 Pin

int pinRB=16;    // Define a 16 Pin
int pinRF=17;    // Define a 17 Pin

int MotorLPWM=5;  //Define a 5 Pin
int MotorRPWM=6;  //Define a 6 Pin

int inputPin = 9;  // Define the ultrasound signal receiving a Pin
int outputPin =8;  //Define the ultrasound signal emission Pin

int Fspeedd = 0;      // go
int Rspeedd = 0;      // The right to
int Lspeedd = 0;      // Turn left to
int directionn = 0;   // After the former = 8 = 2 left = 4 right = 6
Servo myservo;        // Set up the myservo
int delay_time = 250; // After the servo motor to the stability of the time

int Fgo = 8;         // go
int Rgo = 6;         // The right to
int Lgo = 4;         // Turn left to
int Bgo = 2;         // astern

void setup()
 {
  Serial.begin(9600);     // Initialize
  pinMode(pinLB,OUTPUT); // Define 14 pin for the output (PWM)
  pinMode(pinLF,OUTPUT); // Define 15 pin for the output (PWM)
  pinMode(pinRB,OUTPUT); // Define 16 pin for the output (PWM)
  pinMode(pinRF,OUTPUT); // Define 17 pin for the output (PWM)

  pinMode(MotorLPWM,  OUTPUT);  // Define 5 pin for PWM output
  pinMode(MotorRPWM,  OUTPUT);  // Define 6 pin for PWM output

  pinMode(inputPin, INPUT);    // Define the ultrasound enter pin
```

```
  pinMode(outputPin, OUTPUT);  // Define the ultrasound output pin

  myservo.attach(11);    // Define the servo motor output 10 pin(PWM)
 }
void advance(int a)     // go
   {
    digitalWrite(pinRB,HIGH);  // 16 feet for high level
    digitalWrite(pinRF,LOW);   //17 feet for low level
    analogWrite(MotorRPWM,220);//Set the output speed(PWM)
    digitalWrite(pinLB,HIGH);  // 14 feet for high level
    digitalWrite(pinLF,LOW);   //15 feet for high level
    analogWrite(MotorLPWM,220);//Set the output speed(PWM)
    delay(a * 1);
   }

void right(int b)        //right
   {
    digitalWrite(pinRB,LOW);
    digitalWrite(pinRF,HIGH);
    analogWrite(MotorRPWM,220);
    digitalWrite(pinLB,LOW);
    digitalWrite(pinLF,LOW);
    delay(b * 100);
   }
void left(int c)         //left
   {
    digitalWrite(pinRB,LOW);
    digitalWrite(pinRF,LOW);
    digitalWrite(pinLB,LOW);
    digitalWrite(pinLF,HIGH);
    analogWrite(MotorLPWM,220);
    delay(c * 100);
   }
void turnR(int d)        //right
   {
    digitalWrite(pinRB,HIGH);
    digitalWrite(pinRF,LOW);
    analogWrite(MotorRPWM,220);
    digitalWrite(pinLB,LOW);
    digitalWrite(pinLF,HIGH);
    analogWrite(MotorLPWM,220);
```

```
    delay(d * 50);
   }
void turnL(int e)        //left
   {
    digitalWrite(pinRB,LOW);
    digitalWrite(pinRF,HIGH);
    analogWrite(MotorRPWM,220);
    digitalWrite(pinLB,HIGH);
    digitalWrite(pinLF,LOW);
    analogWrite(MotorLPWM,220);
    delay(e * 50);
   }
void stopp(int f)         //stop
   {
    digitalWrite(pinRB,LOW);
    digitalWrite(pinRF,LOW);
    digitalWrite(pinLB,LOW);
    digitalWrite(pinLF,LOW);
    delay(f * 100);
   }
void back(int g)          //back
   {

    digitalWrite(pinRB,LOW);
    digitalWrite(pinRF,HIGH);
    analogWrite(MotorRPWM,220);
    digitalWrite(pinLB,LOW);
    digitalWrite(pinLF,HIGH);
    analogWrite(MotorLPWM,220);
    delay(g * 100);
   }

void detection()        //Measuring three angles(0.90.179)
   {
     int delay_time = 200;   // After the servo motor to the stability of the time
     ask_pin_F();            // Read in front of the distance

    if(Fspeedd < 10)         // If the front distance less than 10 cm
     {
     stopp(1);               // Remove the output data
     back(2);                // The back two milliseconds
```

```
    }

    if(Fspeedd < 25)         // If the front distance less than 25 cm
    {
      stopp(1);              // Remove the output data
      ask_pin_L();           // Read the left distance
      delay(delay_time);     // Waiting for the servo motor is stable
      ask_pin_R();           // Read the right distance
      delay(delay_time);     //  Waiting for the servo motor is stable

      if(Lspeedd > Rspeedd)   //If the distance is greater than the right distance on the left
      {
       directionn = Lgo;      //Left
      }

      if(Lspeedd <= Rspeedd)   //If the distance is less than or equal to the distance on the right
      {
       directionn = Rgo;      //right
      }

      if (Lspeedd < 15 && Rspeedd < 15)   /*If the left front distance and distance and the right distance is
less than 15 cm */
      {

       directionn = Bgo;      //Walk backwards
      }
    }
    else                    //If the front is not less than 25 cm (greater than)
    {
      directionn = Fgo;        //Walk forward
    }

  }
void ask_pin_F()   // Measure the distance ahead
  {
    myservo.write(90);
    digitalWrite(outputPin, LOW);   // For low voltage 2 us ultrasonic launch
    delayMicroseconds(2);
    digitalWrite(outputPin, HIGH);  // Let ultrasonic launch 10 us high voltage, there is at least 10 us
    delayMicroseconds(10);
    digitalWrite(outputPin, LOW);    // Maintaining low voltage ultrasonic launch
```

```
    float Fdistance = pulseIn(inputPin, HIGH);  // Read the time difference
    Fdistance= Fdistance/5.8/10;       // A time to distance distance (unit: cmJ©
    Serial.print("F distance:");      //The output distance (unit: cm)
    Serial.println(Fdistance);         //According to the distance
    Fspeedd = Fdistance;
  }
 void ask_pin_L()   // Measure the distance on the left
  {
    myservo.write(5);
    delay(delay_time);
    digitalWrite(outputPin, LOW);   // For low voltage 2 us ultrasonic launch
    delayMicroseconds(2);
    digitalWrite(outputPin, HIGH);  // Let ultrasonic launch 10 us high voltage, there is at least 10 us
    delayMicroseconds(10);
    digitalWrite(outputPin, LOW);    // Maintaining low voltage ultrasonic launch
    float Ldistance = pulseIn(inputPin, HIGH);  // Read the time difference
    Ldistance= Ldistance/5.8/10;       // Will be time to distance distance (unit: cm)
    Serial.print("L distance:");       //The output distance (unit: cm)
    Serial.println(Ldistance);         //According to the distance
    Lspeedd = Ldistance;               // Will reading Lspeedd distance
  }
void ask_pin_R()   // Measure the distance on the right
  {
    myservo.write(177);
    delay(delay_time);
    digitalWrite(outputPin, LOW);   // For low voltage 2 us ultrasonic launch
    delayMicroseconds(2);
    digitalWrite(outputPin, HIGH);  // Let ultrasonic launch 10 us high voltage, there is at least 10 us
    delayMicroseconds(10);
    digitalWrite(outputPin, LOW);    // Maintaining low voltage ultrasonic launch
    float Rdistance = pulseIn(inputPin, HIGH);  // Read the time difference
    Rdistance= Rdistance/5.8/10;       //Will be time to distance distance (unit: cm)
    Serial.print("R distance:");       //The output distance (unit: cm)
    Serial.println(Rdistance);         //According to the distance
    Rspeedd = Rdistance;
  }

void loop()
 {
   myservo.write(90);  /*Make the servo motor ready position Prepare the next measurement */
   detection();        //Measuring Angle And determine which direction to go to
```

```
 if(directionn == 2)  //If directionn (direction) = 2 (back)
 {
  back(8);                  //  back
  turnL(2);                 //Move slightly to the left (to prevent stuck in dead end lane)
  Serial.print(" Reverse ");   //According to the direction (reverse)
 }
 if(directionn == 6)          //If directionn (direction) = 6 (right)
 {
  back(1);
  turnR(6);                 // right
  Serial.print(" Right ");    //According to the direction (Right)
 }
 if(directionn == 4)         //If directionn (direction) = 4 (left)
 {
  back(1);
  turnL(6);                // left
  Serial.print(" Left ");     //According to the direction (Left)
 }
 if(directionn == 8)         //If directionn (direction) = 8 (forward)
 {
  advance(1);               // go
  Serial.print(" Advance ");   //According to the direction (Advance)
  Serial.print("   ");
 }
}
```