Що рухає серво

Як це працює

Що означає «задати положення»

Звичайний двигун обертається, доки отримує живлення. Позиційному серво ми задаємо, де має опинитися його вал. Наприклад, команда 90 означає приблизно середину ходу. Серво повертається до цього положення й намагається його утримувати. Так можна повертати покажчик, датчик або легку кришку.

Що заховано в корпусі

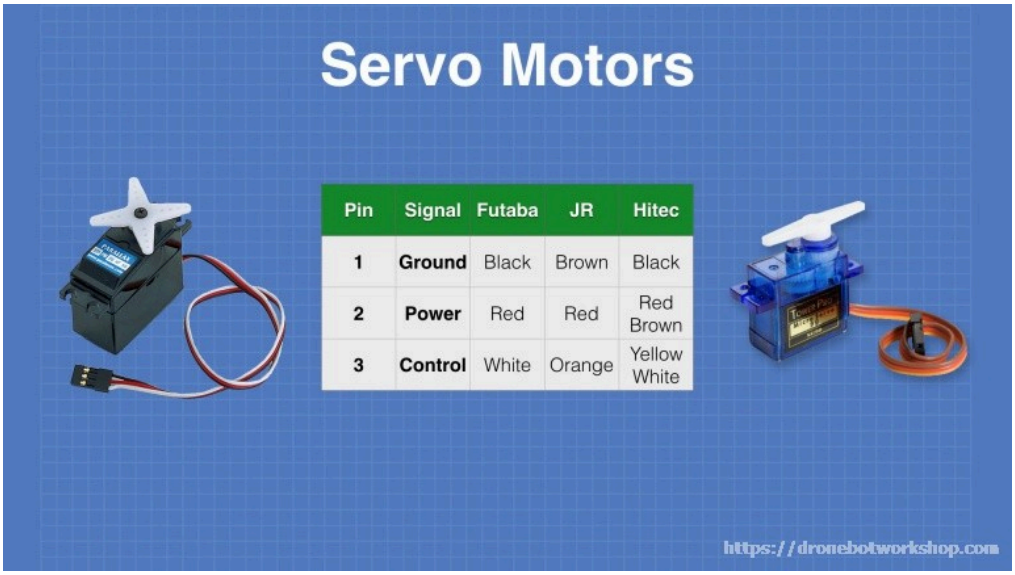
Усередині працюють чотири частини. Двигун створює рух. Шестерні редуктора сповільнюють обертання та збільшують обертальне зусилля на виході. Датчик положення, зазвичай потенціометр, показує електроніці фактичне положення вала. Плата керування порівнює його з командою й вмикає двигун у потрібному напрямку.

Як серво знаходить потрібний кут

Уявімо, що вал стоїть біля 60°, а команда — 90°. Електроніка бачить різницю та рухає вал. Коли він наближається до потрібного положення, різниця зменшується. Таке постійне порівняння називають зворотним зв’язком. Arduino задає ціль, а внутрішня електроніка серво керує самим двигуном.

Вал і важіль

На зубчастий вал встановлюють пластиковий важіль і закріплюють центральним гвинтом. Спочатку задають середнє положення програмою, а вже потім установлюють важіль у потрібному напрямку. Інакше він може дивитися вбік навіть за команди 90.



Знайди на серво вал, корпус і три дроти: `GND`, живлення, сигнал · DroneBot Workshop

Зроби

1. Візьми позиційний SG90 або MG90S, Uno, USB-кабель і дроти.
2. Знайди вал, важіль і три дроти.
3. Залиш вал без важеля для першого запуску.

Очікуваний результат

Ти знайшов рухому частину та роз’єм живлення і сигналу.

Що можна змінити

Поки не підключай живлення.

У Wokwi: відкрий новий Arduino Uno, перенеси diagram.json, sketch/sketch.ino та libraries.txt з ZIP потрібного кроку у відповідні вкладки, натисни ▶. На справжній платі спочатку перевір підключення з учителем. Кожен код замінює весь попередній скетч.

Позиція чи обертання

Як це працює

Схожий корпус — різні команди

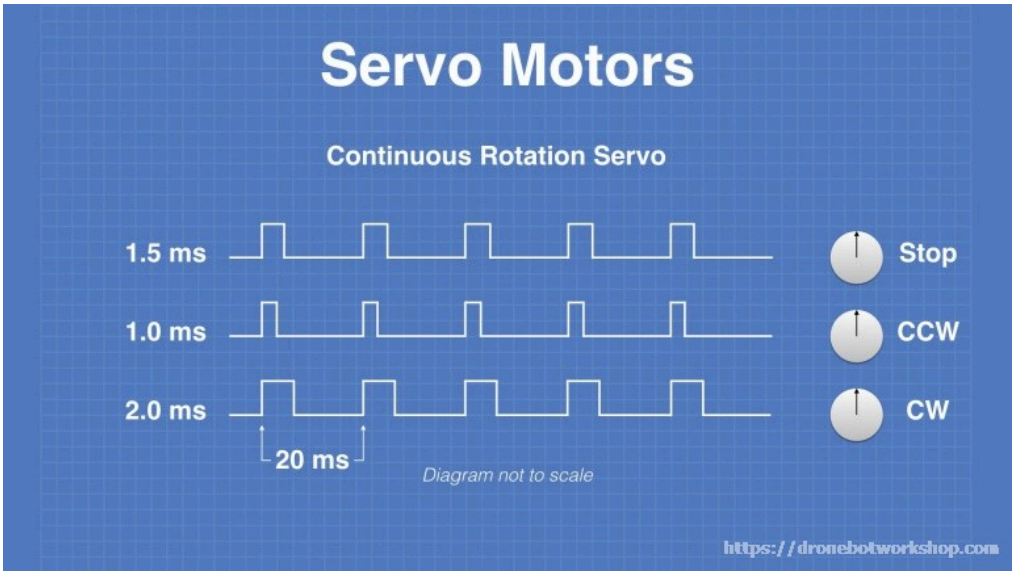
Позиційне серво зупиняється у заданому положенні. Серво безперервного обертання може крутитися багато обертів: схожа команда визначає напрямок і швидкість. Число поблизу 90 для нього зазвичай означає зупинку, але точку зупинки потрібно перевіряти для конкретного екземпляра.

Яке потрібне для нашого досліду

Для покажчика або кришки важливо повторити положення, тому беремо позиційне серво. Для ведучого колеса може бути корисне безперервне обертання. Напис «360°» і колір корпусу самі по собі недостатні: звіряємо точну модель та опис виробника.

Як читати рисунок нижче

На рисунку показано саме безперервне обертання. Короткий, середній і довгий імпульси тут означають різні режими обертання, а не три кути. Не переносимо цю поведінку на наше позиційне серво.



Безперервне обертання: імпульс задає напрямок і швидкість · DroneBot Workshop

Зроби

1. Звір тип свого серво з учителем.
2. Поясни, чому модель безперервного обертання не підійде для точного положення кришки.

Очікуваний результат

Можеш відрізнити команду положення від команди обертання.

Що можна змінити

Тип серво не визначай лише за синім корпусом.

Підключи три дроти

Як це працює

Навіщо три дроти

Живлення V+ постачає енергію двигуну та електроніці. `GND` — зворотний провід живлення і точка відліку напруги. Сигнальний дріт передає команду від Arduino. Через `D9` двигун не живиться: цей пін лише повідомляє, яке положення потрібно зайняти. Червоний дріт зазвичай V+, коричневий або чорний — `GND`, жовтий, білий чи помаранчевий — сигнал; перед підключенням звіряємо модель.

Чому серво має окреме джерело

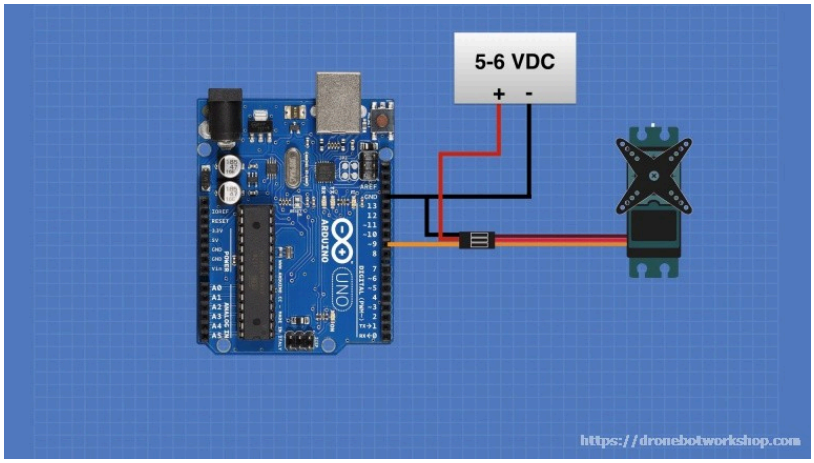
Коли двигун рушає або долає навантаження, йому потрібен більший струм. Якщо живлення не витримує, напруга просідає: серво може смикатися, а Arduino — перезапускатися. Тому в нашому комплекті Arduino працює від USB, а серво — від підготовленого джерела 5 V. Допустиму напругу й потрібний струм перевіряють для конкретного серво.

Чому мінуси з'єднуємо, а плюси — ні

Arduino формує високий і низький рівні сигналу відносно свого `GND`. Серво має бачити ту саму точку відліку. Тому `GND` Arduino з'єднуємо з `GND` джерела серво. Плюси USB/Arduino та окремого джерела у цій схемі залишаємо розділеними: це два різні джерела живлення.

Прочитай схему як два шляхи

Шлях енергії: джерело +5 V → V+ серво; `GND` серво → мінус джерела. Шлях команди: `D9` Arduino → сигнал серво, зі спільним `GND`. На рисунку автора джерело позначене 5–6 V; для нашого підготовленого комплекту беремо 5 V. У Wokwi живлення спрощене — фізичний комплект збираємо за поясненням і таблицею цього кроку.



Окреме живлення серво та спільний `GND` · DroneBot Workshop

Зроби

1. Від'єднай USB та джерело серво.
2. З'єднай дроти за таблицею нижче.
3. Залиш плюсові шини Uno та серво розділеними.
4. Попроси вчителя перевірити полярність.

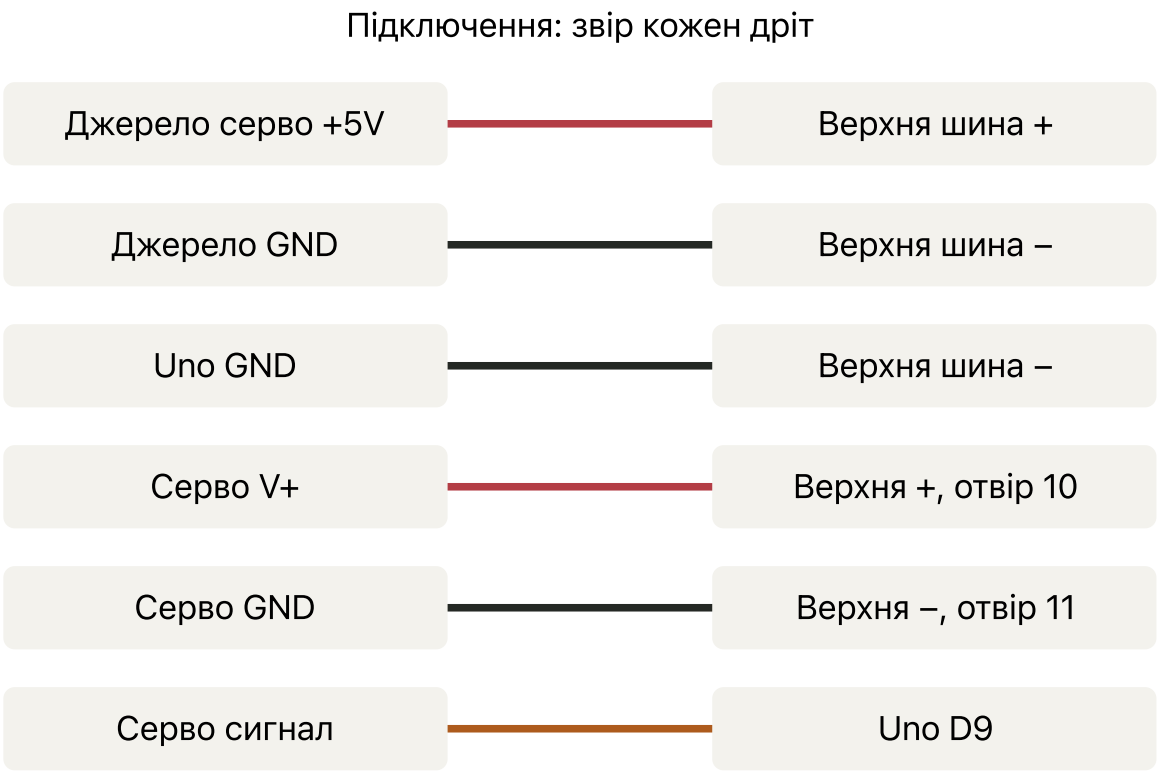
Очікуваний результат

Uno живиться від USB. Серво — від окремого стабілізованого джерела `5V`, яке підготував учитель. `GND` джерела, серво та Uno з'єднані. Плюс джерела серво не з'єднуємо з `5V` Uno. Підключай при вимкнених обох джерелах. У Wokwi серво живиться від Uno як спрощення симуляції; фізичне підключення дивись на схемі уроку.

Що можна змінити

На схемі автора 5–6 V; у нашому досліді використовуємо підготовлені вчителем стабілізовані 5 V.

Підключення справжнього комплекту



Два плюси розділені, GND спільний

На макетній платі верхня + — тільки джерело серво, нижня + — тільки Uno. Об’єднуємо мінусові шини. Перевір розриви шин; за потреби з’єднай половини тієї самої шини. У Wokwi верхня + живиться від Uno лише як спрощення. На справжній платі цей дріт заміни на +5 V окремого джерела; його мінус додай на верхню –.

Задай центральне положення

Як це працює

Що робить бібліотека Servo

Сервопривід очікує повторювані короткі електричні імпульси. Бібліотека Servo формує їх за нас. Після підключення бібліотеки рядок Servo arm створює об’єкт із назвою arm — через нього ми звертаємося до нашого серво.

Розбери три команди

ANGLE зберігає бажаний кут. arm.write(ANGLE) задає це значення бібліотеці. arm.attach(SERVO_PIN) призначає дріт сигналу; SERVO_PIN = 9 означає контакт `D9` на платі. У нашому скетчі спочатку записано початковий кут, а потім увімкнено керування піном. Це число не означає тривалість або швидкість руху.

Чому порожній loop теж працює

Нових команд поки немає, проте бібліотека продовжує надсилати імпульси у фоні. Серво утримує останню ціль. Від’єднання живлення прибирає активне утримання. Для Servo на Uno не обов’язково брати контакт із позначкою ~: тут сигнал створює бібліотека за допомогою таймера.

Команда і справжній кут

90 — зручна команда приблизного центру, а не результат вимірювання транспортиром. Точний кут залежить від серво та встановлення важеля. Починаємо без навантаження й перевіряємо 60–120, щоб не вперти механізм у край ходу.

Зроби

1. В Arduino IDE відкрий менеджер бібліотек і встанови Servo від Arduino.
2. Вибери плату Arduino Uno та її порт.
3. Заміни весь скетч кодом цього кроку.
4. Завантаж код та ввімкни перевірене живлення серво.

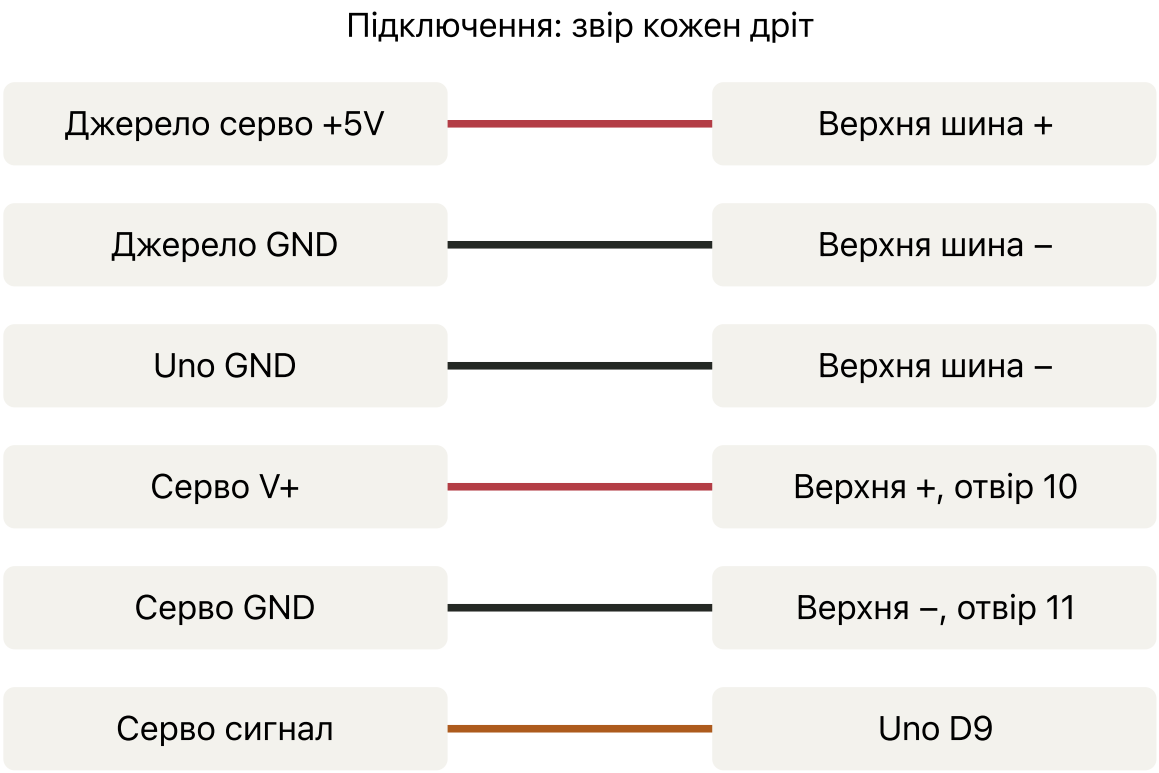
Очікуваний результат

Вал переходить приблизно в середнє положення й утримує його. При гудінні, нагріванні або перезапусках вимкни живлення й поклич учителя.

Що можна змінити

ANGLE: перевір 60, 90, 120 окремими завантаженнями. Не починай з 0 та 180.

Підключення справжнього комплекту



Два плюси розділені, GND спільний

На макетній платі верхня + — тільки джерело серво, нижня + — тільки Uno. Об’єднуємо мінусові шини. Перевір розриви шин; за потреби з’єднай половини тієї самої шини. У Wokwi верхня + живиться від Uno лише як спрощення. На справжній платі цей дріт заміни на +5 V окремого джерела; його мінус додай на верхню –.

Крок 4: код · 1

```
#include <Arduino.h>
#include <Servo.h>
const byte SERVO_PIN = 9;
const int ANGLE = 90; // Початковий кут без стріли
Servo arm;

void setup() {
  arm.write(ANGLE); // Задаємо початкове положення
  arm.attach(SERVO_PIN); // Сигнал керування через D9
}

void loop() {
}
```

Порівняй три положення

Як це працює

Що показує графік

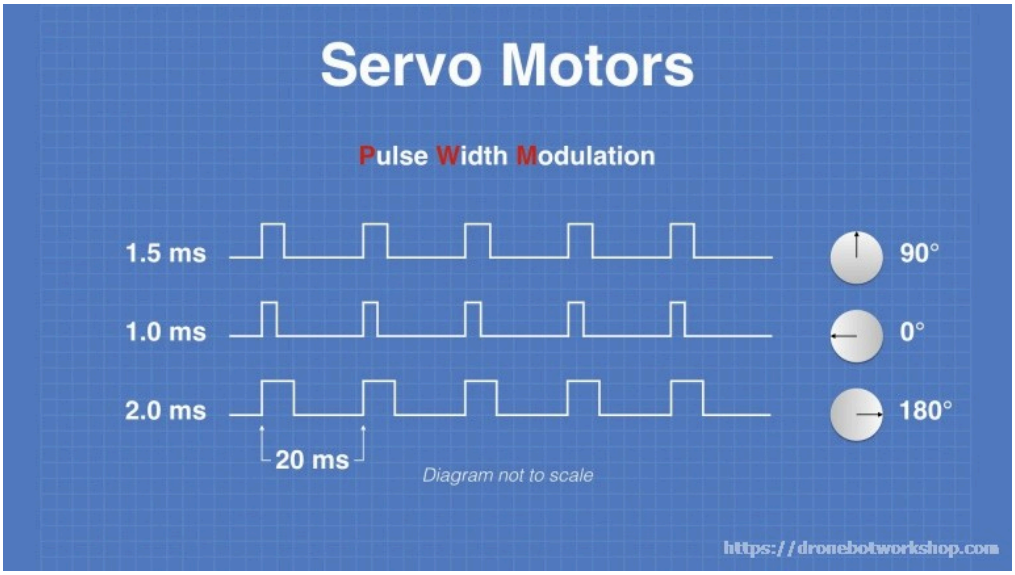
Горизонтальна вісь — час. Висока ділянка лінії означає, що сигнальний дріт на короткий час має високий рівень напруги. Важлива ширина цієї ділянки — тривалість імпульсу. Для типового пояснення часто показують близько 1, 1,5 і 2 мс: вони відповідають різним положенням, а середній імпульс — приблизно центру.

Чому не треба писати імпульси вручну

Команда write переводить кут у тривалість імпульсу, а бібліотека повторює сигнал, зазвичай приблизно кожні 20 мс. Це не звичайне регулювання яскравості LED через analogWrite. Межі імпульсів залежать від моделі та налаштування бібліотеки: підписи на рисунку не є готовими безпечними межами для будь-якого серво.

Що порівнюємо в досліді

Після кожного завантаження змінюється цільове положення. Зверни увагу: команда 120 не просить повернутися ще на 120° від поточного місця. Вона задає конкретну ціль. Повторне надсилання 120 залишає ту саму ціль.



Тривалість імпульсу та положення: типовий приклад · DroneBot Workshop

Зроби

1. Встанови ANGLE = 60 і завантаж код.
2. Повтори для 90 та 120.
3. Намалюй у зошиті три положення вала.
4. Підпиши, яке число ти змінював.

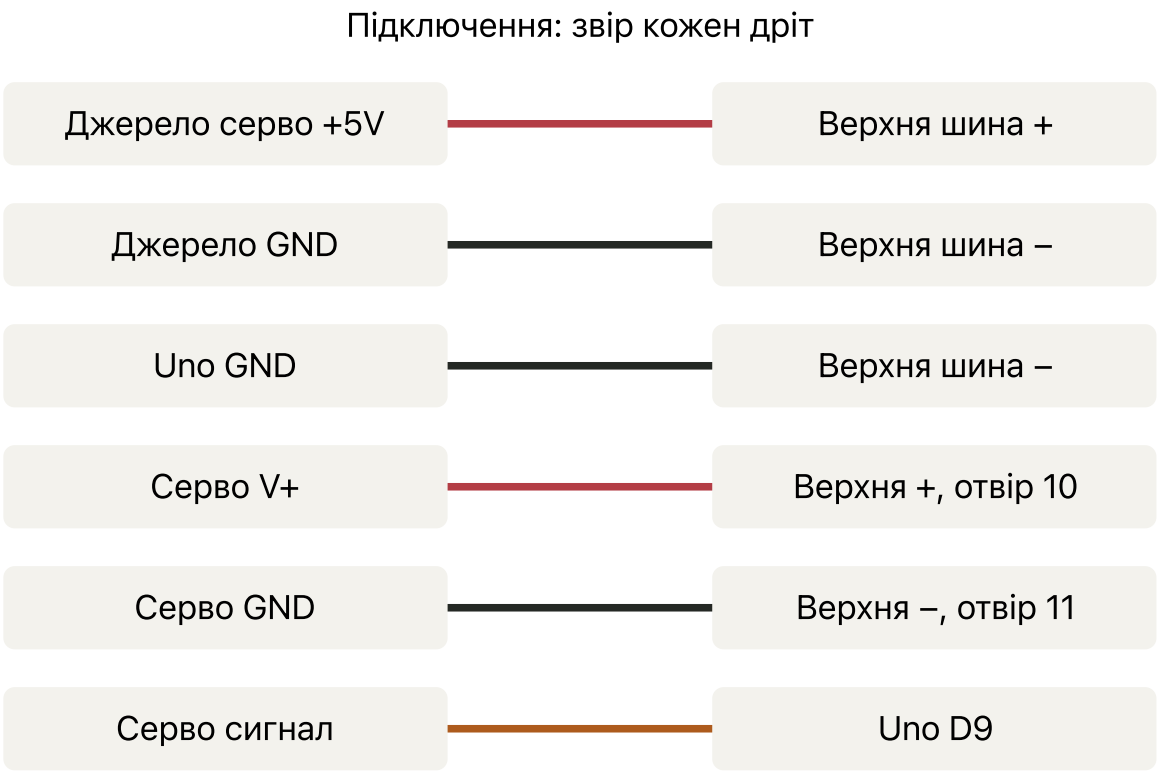
Очікуваний результат

Вал змінює положення, а не обертається безперервно.

Що можна змінити

Залиш діапазон 60–120 до перевірки механіки вчителем.

Підключення справжнього комплекту



Два плюси розділені, GND спільний

На макетній платі верхня + — тільки джерело серво, нижня + — тільки Uno. Об’єднуємо мінусові шини. Перевір розриви шин; за потреби з’єднай половини тієї самої шини. У Wokwi верхня + живиться від Uno лише як спрощення. На справжній платі цей дріт заміни на +5 V окремого джерела; його мінус додай на верхню –.

Крок 5: код · 1

```
#include <Arduino.h>
#include <Servo.h>
const byte SERVO_PIN = 9;
const int ANGLE = 90; // Початковий кут без стріли
Servo arm;

void setup() {
  arm.write(ANGLE); // Задаємо початкове положення
  arm.attach(SERVO_PIN); // Сигнал керування через D9
}

void loop() {
}
```

Запрограмуй два положення

Як це працює

Дві цілі та час між ними

Спочатку програма задає FIRST_ANGLE, потім чекає HOLD_MS. Далі задає SECOND_ANGLE і знову чекає. Оскільки це записано в loop, послідовність повторюється. За HOLD_MS = 2000 між командами минає приблизно дві секунди.

Пауза не означає повільний поворот

Команда з 60 на 120 змінюється одразу. Серво рухається зі швидкістю, яку дозволяють його конструкція, живлення й навантаження. delay(3000) не наказує розтягнути поворот на три секунди. Він лише відкладає наступну команду; імпульси Servo продовжуються.

Передбач, а потім перевір

Якщо збільшити HOLD_MS, важіль довше залишатиметься біля кожної цілі. Щоб зробити сам рух поступовішим, потрібно додати проміжні цілі. Це буде наступний урок.

Зроби

1. Запусти повний код цього кроку.
2. Порівняй HOLD_MS = 1000 та 3000.
3. Збережи свій скетч як Servo_positions.
4. Покажи два положення та поясни роль `GND`.

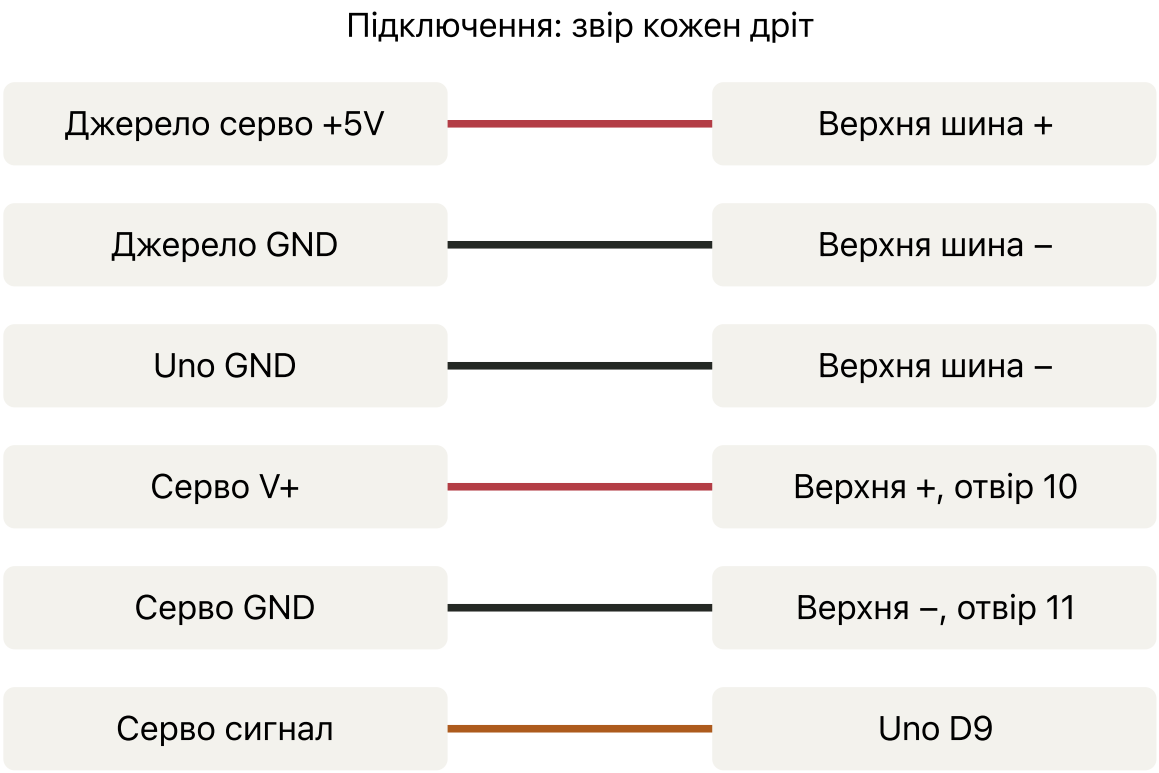
Очікуваний результат

Вал чергує 60 і 120; зміна паузи змінює час утримання.

Що можна змінити

FIRST_ANGLE і SECOND_ANGLE: 60–120; HOLD_MS: 1000–3000 мс.

Підключення справжнього комплекту



Два плюси розділені, GND спільний

На макетній платі верхня + — тільки джерело серво, нижня + — тільки Uno. Об’єднуємо мінусові шини. Перевір розриви шин; за потреби з’єднай половини тієї самої шини. У Wokwi верхня + живиться від Uno лише як спрощення. На справжній платі цей дріт заміни на +5 V окремого джерела; його мінус додай на верхню –.

Крок 6: код · 1

```
#include <Arduino.h>
#include <Servo.h>
const byte SERVO_PIN = 9;
const int ANGLE = 90; // Початковий кут без стріли
Servo arm;
const int FIRST_ANGLE = 60; // Перше положення
const int SECOND_ANGLE = 120; // Друге положення
const unsigned long HOLD_MS = 2000; // Час у кожному положенні

void setup() {
  arm.write(ANGLE); // Задаємо початкове положення
  arm.attach(SERVO_PIN); // Сигнал керування через D9
}

void loop() {
  arm.write(FIRST_ANGLE);
  delay(HOLD_MS);
  arm.write(SECOND_ANGLE);
  delay(HOLD_MS);
}
```

Джерела

Стаття й ілюстрації — DroneBot Workshop · Відео автора. Практичні завдання та програми адаптовано для цього курсу.

Далі: Серво: плавний рух →