Лабораторная работа №9.

Танцевальные движения и их комбинации

Цель:

Закрепить навыки программирования сложных последовательностей движений сервоприводов. Научиться комбинировать готовые функции (примитивы движения, световые и звуковые эффекты) для создания зрелищных и интерактивных анимаций поведения робота. Освоить принципы построения сценариев и реактивного поведения.

Теоретическая часть

Создание сложных анимаций

Ключ к созданию интересного и "живого" поведения робота — не в написании одного гигантского алгоритма, а в **комбинации простых движений** (примитивов) в правильной последовательности.

Основные принципы:

- **Модульность:** Каждая функция (`hello()`, `lunging_side()`, `forward()`) — это отдельный "танец" или "движение".
- **Последовательность:** Движения выполняются одно за другим, образуя танец.
- **Паузы:** Между движениями необходимы паузы (через `delay()` или `millis()`), чтобы зритель мог их воспринять и чтобы робот стабилизировался.
- **Синхронизация:** Движения можно и нужно синхронизировать со световыми (`rgb()`) и звуковыми (`tone()`) эффектами.
- **Интерактивность:** Поведение может зависеть от данных с датчиков (например, ультразвука).

Анализ движения "Выпад в сторону" (lunging_side)

```
 void lunging_side() {
  // Фаза 1: Подготовка - корпус смещается в сторону
  for (int i = 90; i >= 30; i -= 2) {
    servoA.write(180 - i); // Правая передняя лапа идет вперед-всторону
    servoC.write(i);       // Правая средняя лапа идет назад-всторону
    servoE.write(180 - i); // Левая задняя лапа идет вперед-всторону
    servoG.write(i);       // Левая средняя лапа идет назад-всторону
    delay(del);
  }
  // Фаза 2: Активное качание ("тряска") на месте
  for (int j = 0; j < 10; j++) { // Цикл из 10 качаний
    // Поднимаем одни лапы, опускаем другие (создает эффект тряски)
    for (int i = 90; i >= 30; i -= 2) {
      servoB.write(180 - i);
      servoD.write(i);
      servoF.write(i);
      servoH.write(180 - i);
      delay(del * 3); // Медленное движение для эффекта тяжести
    }
    // Обратное движение
    for (int i = 30; i <= 150; i += 2) { // Широкий диапазон для
большой амплитуды
      servoB.write(180 - i);
      servoD.write(i);
      servoF.write(i);
      servoH.write(180 - i);
      delay(del * 3);
    }
    // Возврат к средней точке качания
    for (int i = 150; i >= 90; i -= 2) {
      servoB.write(180 - i);
      servoD.write(i);
      servoF.write(i);
      servoH.write(180 - i);
      delay(del * 3);
    }
  }
```

```
// Фаза 3: Возврат корпуса в нейтральное положение
  for (int i = 30; i <= 90; i += 2) {
    servoA.write(i);
    servoC.write(180 - i);
    servoE.write(i);
    servoG.write(180 - i);
    delay(del);
  }
}
```

Фаза 1: Робот смещает центр тяжести, принимая устойчивую позу для "тряски".

Фаза 2: Основное движение. Обратите внимание на delay(del*3) — оно делает движение подъемных сервоприводов более плавным и весомым.

Фаза 3: Важный этап возврата в исходное положение для последующих движений.

Практическая часть

Задание 1: Анализ и модификация движения

Цель: Понять работу функции и адаптировать ее под свои нужды.

Задание:

1. Загрузите и запустите функцию `lunging_side()`. Опишите своими словами, что делает робот.

2. Измените количество повторений цикла `j < 10` на `j < 3`. Сравните результат.

3. Вопрос: Какие сервоприводы в Фазе 2 отвечают за подъем/опускание правой стороны, а какие — левой?

4. Усложнение: Измените амплитуду "тряски", отредактировав диапазон значений в циклах Фазы 2 (например, с i = 90; i >= 50; ...).

Задание 2: Создание своего танцевального движения

Цель: Написать свою уникальную функцию-движение.

Пример функции "Кивок":

```
void head_bang() {
  // Опускаем аем переднюю часть
  for (int i = 90; i > 45; i--) {
    servoB.write(i);    // Правая передняя
    servoH.write(180 - i); // Левая передняя
    delay(del * 2);
  }
  // "Киваем" несколько раз
  for (int j = 0; j < 2; j++) {
    for (int i = 45; i < 135; i += 5) {
      servoB.write(i);
      servoH.write(180 - i);
      delay(del*3);
    }
    for (int i = 135; i > 45; i -= 5) {
      servoB.write(i);
      servoH.write(180 - i);
      delay(del*3);
    }
  }
  center_servos();
  delay(1000); // Возвращаем все сервы в исходное положение
}
```

Задание:

1. Напишите свою функцию, например, `spin()` (быстрое вращение на месте) или `wiggle()` (быстрое покачивание всеми лапками).

2. Обязательно завершайте функцию вызовом `center_servos()` или плавным возвратом в нейтральную позу.

Задание 3: Танец по сценарию

Цель: Создать последовательность (хореографию) из готовых движений, написанных в предыдущих лабораторных работах.

Код (Сценарий "Заводной краб"):

```
void dance_routine() {
  Serial.println("Начинаем танец!");

  // 1. Приветствие
  hello();
  delay(1000);

  // 2. Два выпада влево и вправо
  for (int i = 0; i < 2; i++) {
    lunging_side(); // Выпад в сторону (по умолчанию, например, вправо)
    delay(300);
    // Здесь должна быть функция выпада в другую сторону (left_lunging_side()),
    // но если ее нет, можно развернуться
    right(); right(); // Быстрый разворот на 180 условных градусов
    delay(300);
  }

  // 3. Кивок
  head_bang();
  delay(500);

  // 4. Кружение на месте
  for (int i = 0; i < 4; i++) {
    right(); // Поворот направо на 90 условных градусов
  }

  // 5. Прощание
  bye();
  Serial.println("Танец окончен!");
}

void loop() {
  // Танец запускается по нажатию кнопки или один раз при старте
  dance_routine();
  delay(10000); // Большая пауза перед повторением
}
```

Задание:

1. Составьте свой сценарий танца, используя функции из предыдущих лаб (`hello`, `forward`, `right`, `left`, `lunging_side`, `head_bang`).

2. Добавьте в сценарий световые эффекты (`rgb(random(255), random(255), random(255))`) и звуковое сопровождение (`tone(8, melody[...], duration)`).

Задание 4: Интерактивный танец (Танцевальный баттл)

Цель: Создать последовательность (хореографию) из готовых движений, написанных в предыдущих лабораторных работах.

Код:

```
// Глобальные переменные для управления состоянием
bool isDancing = false;
int danceMove = 0;

void loop() {
  int dist = getDistance(); // Получаем расстояние с HC-SR04

  if (dist < 25 && !isDancing) {
    // Если рука близко и мы не танцуем — начинаем новый танец
    isDancing = true;
    danceMove = random(1, 4); // Случайно выбираем движение: 1, 2 или 3
    Serial.print("Запускаю движение №: ");
    Serial.println(danceMove);
  }

  if (isDancing) {
    // В зависимости от выбранного движения, выполняем его
    switch (danceMove) {
      case 1:
        rgb(255, 0, 0); // Красный
        lunging_side();
        break;
      case 2:
        rgb(0, 255, 0); // Зеленый
        head_bang();
        break;
```

```
      case 3:
        rgb(0, 0, 255); // Синий
        for (int i = 0; i < 2; i++) { hello(); } // Двойное приветствие
        break;
    }
    // После выполнения движения возвращаемся в исходное состояние
    center_servos();
    rgb(0, 0, 0); // Гасим светодиод
    isDancing = false;
  }

  delay(100); // Небольшая задержка для стабильности
}
```

Задание:

1. Реализуйте интерактивный танец. Поднесите руку — робот выполняет случайное движение.

2. Усложните алгоритм: пусть расстояние определяет не только факт запуска, но и характер движения (например, чем ближе рука, тем быстрее или амплитуднее танец).

3. Вопрос: Почему для управления состоянием (`isDancing`) используется отдельная переменная-флаг, а не просто проверка расстояния внутри функций танца?

Контрольные вопросы

1. Почему после завершения собственной функции движения (например, head_bang()) важно вызывать center_servos()?

2. Предложите структуру данных (например, массивы) для записи последовательности движений (хореографии), чтобы ее можно было легко редактировать и воспроизводить в цикле.

3. Как можно интегрировать в танец данные с "радара" из Лабораторной №7, чтобы робот танцевал, поворачиваясь в сторону самого дальнего препятствия (самого большого свободного пространства)?