Проверка датчиков и сервоприводов

Цель

Научиться задавать скорость и направление работы сервоприводов, а также снимать показания с датчиков.

Описание

Перед тем как начинать работу с манипулятором, необходимо разобраться с некоторыми особенностями управления сервоприводами и с тем, как снимать показания с различных датчиков.

Попробуйте написать программу, которая будет считывать углы поворота потенциометров, включать по кнопке звукоизлучатель, и, при включении внешнего питания, перемещать манипулятор в какое-либо положение (Проверка работоспособности сервоприводов)

Внимание! Запомните, что максимальные значения поворотов для каждого из сервоприводов:

Для 1: 200-800

Для 2: 400-660

Для 3: 200-600

Для 4: 500-1000

Не используйте другие значения, иначе вы можете повредить манипулятор!

Решение

Проверим работоспособность всех датчиков, воспользовавшимся с кодом:

```
//Подключение библиотеки
#include <DynamixelWorkbench.h>
```

```
//Требуется для инициализации
#define DEVICE_NAME  "3"

//Боудрейт, для моторов - 1000000, для Serial порта - 57600
#define BAUDRATE  1000000
#define SERIAL_BAUDRATE 57600

//ID моторов, должны идти по возрастанию
#define DXL_ID1    1
#define DXL_ID2    2
#define DXL_ID3    3
#define DXL_ID4    4

//Пины для потенциометров
#define ANALOG_PIN1  A0
#define ANALOG_PIN2  A1
#define ANALOG_PIN3  A2
#define ANALOG_PIN4  A3

//Пин для ИК датчика
#define IR_SENSOR_PIN  A8

//Пин для кнопки
#define BUTTON_PIN  10

//Пин для звукоизлучателя
#define BUZZ_PIN  11

//Создание рабочего пространства servos
DynamixelWorkbench servos;

void setup() {
    const char *log;

    //Задаем всем аналоговым пинам то, что они работают на вход, также и
для цифровых, кроме звукоизлучателя, для него - выход.
    pinMode(ANALOG_PIN1, INPUT);
    pinMode(ANALOG_PIN2, INPUT);
    pinMode(ANALOG_PIN3, INPUT);
    pinMode(ANALOG_PIN4, INPUT);
    pinMode(IR_SENSOR_PIN, INPUT);
    pinMode(BUTTON_PIN, INPUT);
    pinMode(BUZZ_PIN, OUTPUT);

    Serial.begin(SERIAL_BAUDRATE);

    //Инициализируем сервоприводы
    servos.init(DEVICE_NAME, BAUDRATE);

    //"Включаем" сервоприводы,
    servos.ping(DXL_ID1, 0, 0);
    servos.ping(DXL_ID2, 0, 0);
    servos.ping(DXL_ID3, 0, 0);
    servos.ping(DXL_ID4, 0, 0);

    //Включение сервоприводов для движения: (ID, скорость, ускорение)
    servos.jointMode(DXL_ID1, 40, 40);
    servos.jointMode(DXL_ID2, 40, 40);
    servos.jointMode(DXL_ID3, 40, 40);
    servos.jointMode(DXL_ID4, 40, 40);
```

```
    //Ставим манипулятор в "центральное" положение. Ждем 5 секунд для
выполнения
    servos.goalPosition(DXL_ID1, 500);
    servos.goalPosition(DXL_ID2, 500);
    servos.goalPosition(DXL_ID3, 500);
    servos.goalPosition(DXL_ID4, 500);
    delay (5000);
}

void loop() {
    //Считываем значения с потенциометров и выводим в Serial порт
    Serial.print("\nПотенциометр 1: ");
    Serial.println(analogRead(ANALOG_PIN1));
    Serial.print("Потенциометр 2: ");
    Serial.println(analogRead(ANALOG_PIN2));
    Serial.print("Потенциометр 3: ");
    Serial.println(analogRead(ANALOG_PIN3));
    Serial.print("Потенциометр 4: ");
    Serial.println(analogRead(ANALOG_PIN4));
    Serial.print("ИК датчик: ");
    Serial.println(analogRead(IR_SENSOR_PIN));
    //Если нажать на кнопку, будет сигнал на звукоизлучателе
    if (digitalRead(BUTTON_PIN) == HIGH)
        digitalWrite(BUZZ_PIN, HIGH);
    else
        digitalWrite(BUZZ_PIN, LOW);
    delay (100);
}
```

Не забываем, что все значения пинов вы должны задать в соответствии с подключением их на плате.

Открывая последовательный порт, вы можете получить значения с потенциометров, а нажимая на кнопку, получить звуковой сигнал.

Если вы отключите плату от компьютера и включите внешнее питание, тогда у вас манипулятор сам переместится в центральное положение с открытым захватом.

Крайние положения

Цель

Целью задания является узнать возможности манипулятора, зону его действия

Описание

Вращение — это круто. Но увы, в основе роботов-манипуляторов находятся компоненты, не позволяющие достигать абсолютно любой точки в пространстве. Необходимо выяснить, какие пределы у данного "экземпляра".

Решение

Запустите данный код:

```
//Подключение библиотеки
#include <DynamixelWorkbench.h>

//Требуется для инициализации
#define DEVICE_NAME  "3"

//Боудрейт, для моторов - 1000000, для Serial порта - 57600
#define BAUDRATE  1000000
#define SERIAL_BAUDRATE 57600

//ID моторов, должны идти по возрастанию
#define DXL_ID1    1
#define DXL_ID2    2
#define DXL_ID3    3
#define DXL_ID4    4


//Создание рабочего пространства servos
DynamixelWorkbench servos;

void setup() {
    const char *log;

    Serial.begin(SERIAL_BAUDRATE);

    //Инициализируем сервоприводы
    servos.init(DEVICE_NAME, BAUDRATE);

    //"Включаем" сервоприводы,
```

```
    servos.ping(DXL_ID1, 0, 0);
    servos.ping(DXL_ID2, 0, 0);
    servos.ping(DXL_ID3, 0, 0);
    servos.ping(DXL_ID4, 0, 0);

    //Включение сервоприводов для движения: (ID, скорость, ускорение)
    servos.jointMode(DXL_ID1, 40, 40);
    servos.jointMode(DXL_ID2, 40, 40);
    servos.jointMode(DXL_ID3, 40, 40);
    servos.jointMode(DXL_ID4, 40, 40);

    //Ставим манипулятор в "центральное" положение. Ждем 5 секунд для
выполнения
    servos.goalPosition(DXL_ID1, 500);
    servos.goalPosition(DXL_ID2, 500);
    servos.goalPosition(DXL_ID3, 500);
    servos.goalPosition(DXL_ID4, 500);
    delay (5000);
}

void loop() {
}
```

Теперь, постепенно, меняем для каждого сервопривода, по отдельности, значение в servos.goalPosition с 500 на большее или меньшее, но не сильно, примерно на 10-50, прошиваем и смотрим, как себя поведет сервопривод и манипулятор в целом. Будьте готовы отключить питание, чтобы не сломать его!

Вот примерные значения минимумов и максимумов, полученные на одном из манипуляторов:

Min:

```
    servos.goalPosition(DXL_ID1, 200);
    servos.goalPosition(DXL_ID2, 400);
    servos.goalPosition(DXL_ID3, 200);
    servos.goalPosition(DXL_ID4, 500);
```

Max:

```
    servos.goalPosition(DXL_ID1, 800);
    servos.goalPosition(DXL_ID2, 660);
    servos.goalPosition(DXL_ID3, 600);
    servos.goalPosition(DXL_ID4, 1000);
```

Следующее задание с мануальным управлением позволит это реализовать данную часть задания легче и не сломать манипулятор, если

Вы посчитали максимальные и минимальные углы, так что стоит сначала реализовать его и перейти к этой части:

Аккуратно увеличивайте угол на всех сервоприводах (лучше всего для этого подойдет управление на потенциометрах), наблюдая за изменением положения манипулятора. Какую взаимосвязь вы увидели?

После того, как вы попробовали повращать сервоприводы в хаотичном порядке, стоит попробовать вытянуть манипулятор на максимальную длину.

Мануальное управление

Цель

Целью является управление манипулятором с помощью потенциометров. Это позволяет попробовать себя в роли “микроконтроллера”, осознать, какой колоссальный труд проделывают программисты и инженеры при работе.

Описание

Вы никогда не задумывались, как промышленный манипулятор совершает свои действия? Конечно, всегда существуют специальные программы, в которые заносятся координаты, и манипулятор с легкостью выполняет поставленные задачи.

Для этого есть обратная задача кинематики, которую уже решили до конечного пользователя, но что было бы, если приходилось всегда самому выбирать углы для сервоприводов?

Именно это вы и попробуете сделать в данной задачи! Вам предстоит управлять манипулятором только с помощью четырех потенциометров, но сможете ли вы что-нибудь взять?

Напишите код, который будет решать задачу управления сервомоторами при помощи потенциометров. Попробуйте добавить

Решение

Подключите четыре потенциометра к роботу, чтобы каждый потенциометр отвечал только за один сервопривод. Задав, с помощью команды *map,* максимальные диапазоны для передвижения сервоприводов.

Если Вы сделали все верно, тогда у Вас можно управлять манипулятором только с помощью потенциометров. Закрепите

манипулятор на рабочей поверхности и попробуйте что-нибудь схватить. Лучше всего, если вы попробуете схватить кубик или фигурку коня из шахмат, она Вам еще потребуется в следующем задании.

Если вы желаете посмотреть на пример кода, тогда можете воспользоваться им:

```
//Подключение библиотеки
#include <DynamixelWorkbench.h>

//Требуется для инициализации
#define DEVICE_NAME  "3"

//Боудрейт, для моторов - 1000000, для Serial порта - 57600
#define BAUDRATE  1000000
#define SERIAL_BAUDRATE 57600

//ID моторов, должны идти по возрастанию
#define DXL_ID1    1
#define DXL_ID2    2
#define DXL_ID3    3
#define DXL_ID4    4

//Пины для потенциометров
#define ANALOG_PIN1  A0
#define ANALOG_PIN2  A1
#define ANALOG_PIN3  A2
#define ANALOG_PIN4  A3

//Переменные, в которые попадают значения с сервоприводов и потенциометров
int32_t input1 = 0;
int32_t output1 = 0;

int32_t input2 = 0;
int32_t output2 = 0;

int32_t input3 = 0;
int32_t output3 = 0;

int32_t input4 = 0;
int32_t output4 = 0;

//Создание рабочего пространства servos
DynamixelWorkbench servos;

void setup() {
    const char *log;

    //Задаем всем аналоговым пинам то, что они работают на вход
    pinMode(ANALOG_PIN1, INPUT);
    pinMode(ANALOG_PIN2, INPUT);
    pinMode(ANALOG_PIN3, INPUT);
    pinMode(ANALOG_PIN4, INPUT);

    Serial.begin(SERIAL_BAUDRATE);

    //Инициализируем сервоприводы
```

```
    servos.init(DEVICE_NAME, BAUDRATE);

    //"Включаем" сервоприводы
    servos.ping(DXL_ID1, 0, 0);
    servos.ping(DXL_ID2, 0, 0);
    servos.ping(DXL_ID3, 0, 0);
    servos.ping(DXL_ID4, 0, 0);

    //Включение сервоприводов для движения: (ID, скорость, ускорение)
    servos.jointMode(DXL_ID1, 40, 40);
    servos.jointMode(DXL_ID2, 40, 40);
    servos.jointMode(DXL_ID3, 40, 40);
    servos.jointMode(DXL_ID4, 40, 40);

}

void loop() {
    //Считываем значение с потенциометра
    input1 = analogRead(ANALOG_PIN1);
    //Приводим значение с потенциометра к значениям позиции сервопривода
    output1 = map(input1, 0, 1023, 300, 800);
    //Отправляем команду с позицией на сервопривод
    servos.goalPosition(DXL_ID1, output1);

    input2 = analogRead(ANALOG_PIN2);
    output2 = map(input2, 0, 1023, 400, 660);
    servos.goalPosition(DXL_ID2, output2);

    input3 = analogRead(ANALOG_PIN3);
    output3 = map(input3, 0, 1023, 200, 600);
    servos.goalPosition(DXL_ID3, output3);

    input4 = analogRead(ANALOG_PIN4);
    output4 = map(input4, 0, 1023, 500, 1000);
    servos.goalPosition(DXL_ID4, output4);
}
```

Будьте аккуратны со значениями *map*, чтобы не сломать манипулятор! Лучше поставить значения меньше, чтобы проверить, на сколько безопасно так делать, а потом уже увеличивать диапазон. Ведь мы никуда не спешим!

Перенос кубика

Цель

Целью задачи является испытание захвата манипулятора, чтобы перенести объект - в данном случае кубик.

Описание

Все мы в детстве играли с кубиками. Если нет, то у вас не было детства. У робота манипулятора его тоже не было, он с места производства сразу приехал к Вам, так почему бы не устроить ему небольшое погружение в пучину беззаботности и счастья?

Для этого дадим ему кубик. Пусть попробует поиграть с ним, перенося с места на место.

Не требуется относить его далеко, можно даже просто вернуть его на то же самое место, но подняв при помощи манипулятора.

Решение

Первое задание без примера кода! Вам предстоит потрудиться с ним! На самом деле, ничего сложного нет. Попробуйте при помощи комбинации из двух представленных кодов составить свою программу, которая будет считывать значения потенциометров во время перемещения манипулятора, и все что вам останется в конце – просто перенести кубик по данным координатам.

Но будьте аккуратны! Задайте задержки на передвижение и захват, чтобы робот не сделал задачу очень быстро, сломав кубик или себя, к примеру. Также ограничивайте скорость перемещения, чтобы ничего не повредить!

“Проба пера”

Цель

Цель - попробовать порисовать с помощью робота

Описание

- Может ли робот создать шедевр?...
- Конечно может! Главное - его правильно запрограммировать

Вам предстоит, с помощью ручки, попробовать провести линию, пусть даже и не прямую.

Решение

Придумайте, как можно закрепить ручку или, желательно, фломастер на рабочем органе робота. Вам не нужно пытаться сейчас именно взять его, а надо только попробовать подвести манипулятор к бумаге и, выдерживая высоту, провести линию. Или, как вариант, поднести захват к листу бумаги, дать роботу ручку, подождать пока он схватит ее и проведет линию.

У вас будет множество экспериментов, пока вы будете пытаться не порвать бумагу и в этом нет ничего страшного! Если с бумагой не получается, тогда стоит перейти на жесткий картон.

Если хочется увидеть пример кода, когда робот, на листе А4, рисует дугу, тогда вот он:

```
//Подключение библиотеки
#include <DynamixelWorkbench.h>

//Требуется для инициализации
#define DEVICE_NAME  "3"

//Боудрейт, для моторов - 1000000, для Serial порта - 57600
#define BAUDRATE  1000000
#define SERIAL_BAUDRATE 57600

//ID моторов, должны идти по возрастанию
#define DXL_ID1    1
```

```
#define DXL_ID2     2
#define DXL_ID3     3
#define DXL_ID4     4

//Создание рабочего пространства servos
DynamixelWorkbench servos;

void setup() {
    const char *log;

    Serial.begin(SERIAL_BAUDRATE);

    //Инициализируем сервоприводы
    servos.init(DEVICE_NAME, BAUDRATE);

    //"Включаем" сервоприводы
    servos.ping(DXL_ID1, 0, 0);
    servos.ping(DXL_ID2, 0, 0);
    servos.ping(DXL_ID3, 0, 0);
    servos.ping(DXL_ID4, 0, 0);

    //Включение сервоприводов для движения: (ID, скорость, ускорение)
    servos.jointMode(DXL_ID1, 40, 40);
    servos.jointMode(DXL_ID2, 40, 40);
    servos.jointMode(DXL_ID3, 40, 40);
    servos.jointMode(DXL_ID4, 80, 80);

    //Отправляем команды с начальной позицией на сервоприводы
    servos.goalPosition(DXL_ID1, 500);
    servos.goalPosition(DXL_ID2, 500);
    servos.goalPosition(DXL_ID3, 500);
    servos.goalPosition(DXL_ID4, 500);

    //Ждем 7 секунд, чтобы робот успел достигнуть ее и Вы успели положить
ему маркер в манипулятор
    delay (7000);

    //Захватываем ручку или маркер, даем ему время на захват и чтобы Вы
успели убрать руку
    servos.goalPosition(DXL_ID4, 900);
    delay (5000);

    //Рисуем в две стороны по одной из осей
    servos.goalPosition(DXL_ID1, 400);
    delay (2000);
    servos.goalPosition(DXL_ID1, 600);
}

void loop() {
}
```

Данный код приводит манипулятор в начальное положение, где надо дать ему ручку, которая касается бумаги, а затем просто наблюдать, как манипулятор рисует такую дугу:

Учтите, что все значения захвата или позиции могут меняться, в зависимости от ваших настроек и ручек! Экспериментируйте!

Возможно, у вас появятся некоторые вопросы, к примеру такие:

- Не выходит ничего нарисовать, бумага рвется

Поменяйте бумагу на что-то более жесткое, к примеру картон.

Также можно заменить ручку на фломастер, он мягче и писать им легче.

Также, под лист бумаги можно подложить что-то мягкое, чтобы лист немного пружинил

- Бумага перемещается вместе с ручкой

Лучше всего, чтобы бумага была неподвижной, закрепите ее по углам, с помощью скотча или изоленты, к столу. Так будет легче работать с листом бумаги, так как он не будет перемещаться, пока вы пробуете рисовать на нем.

“Ход конем”

Цель

Взять шахматную фигуру и сделать "ход конем"

Описание

Манипуляторы - очень удобная штука. Они позволяют выполнять различные операции по перемещению объектов из точки А в точку Б. А ещё у нас есть компьютеры, которые очень хорошо умеют играть в шахматы. Так почему бы не объединить эти два прекрасных изобретения? Мы не будем подключать компьютер к манипулятору, но научим его делать "ход конем".

Решение

Кажется, что это одно из сложнейших заданий, но это не так.

Можно просто попробовать записать координаты коня и вариантов хода или решить данную задачу при помощи обратной задачи кинематики, все в ваших руках! Можно даже походить при помощи потенциометров. В любом случае, попробуйте задать вариативность хода. У вас в наборе есть кнопка, и, в зависимости от количества нажатий на нее, должна меняться вариативность хода. К примеру – одно нажатие ходит “влево”, два – “вправо”.

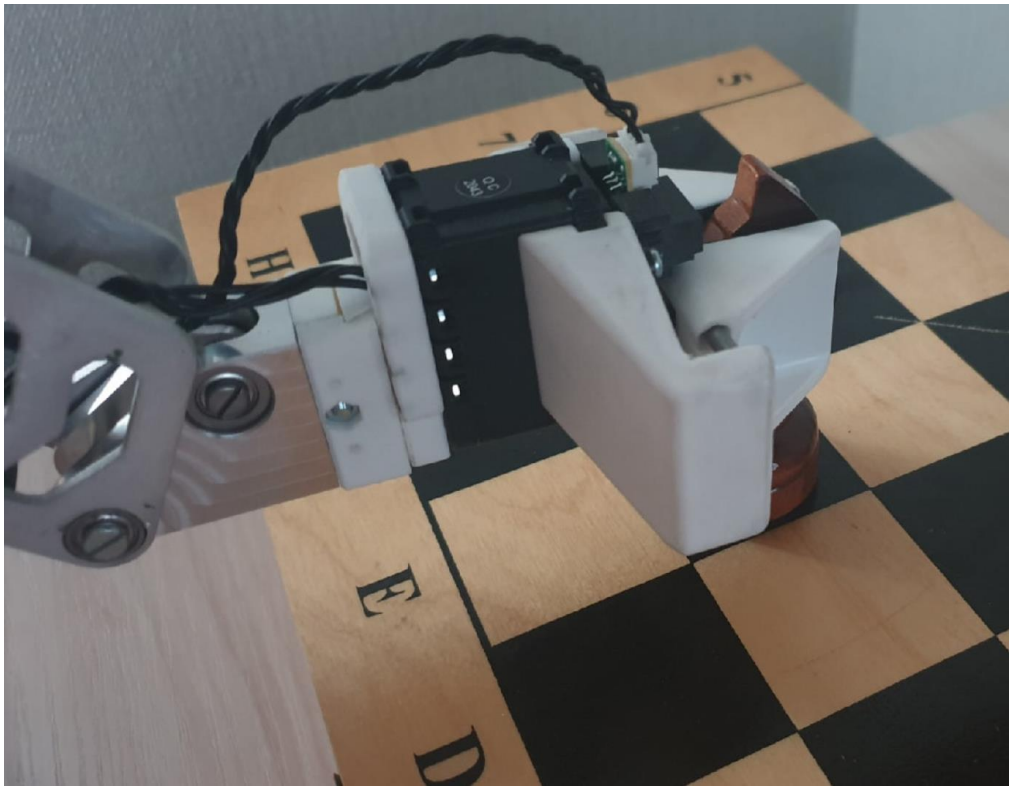
Как это реализовать? Добавить задержку перед началом работы робота, чтобы он считывал нажатия кнопки и начинал работу только через 5-10 секунд после включения.

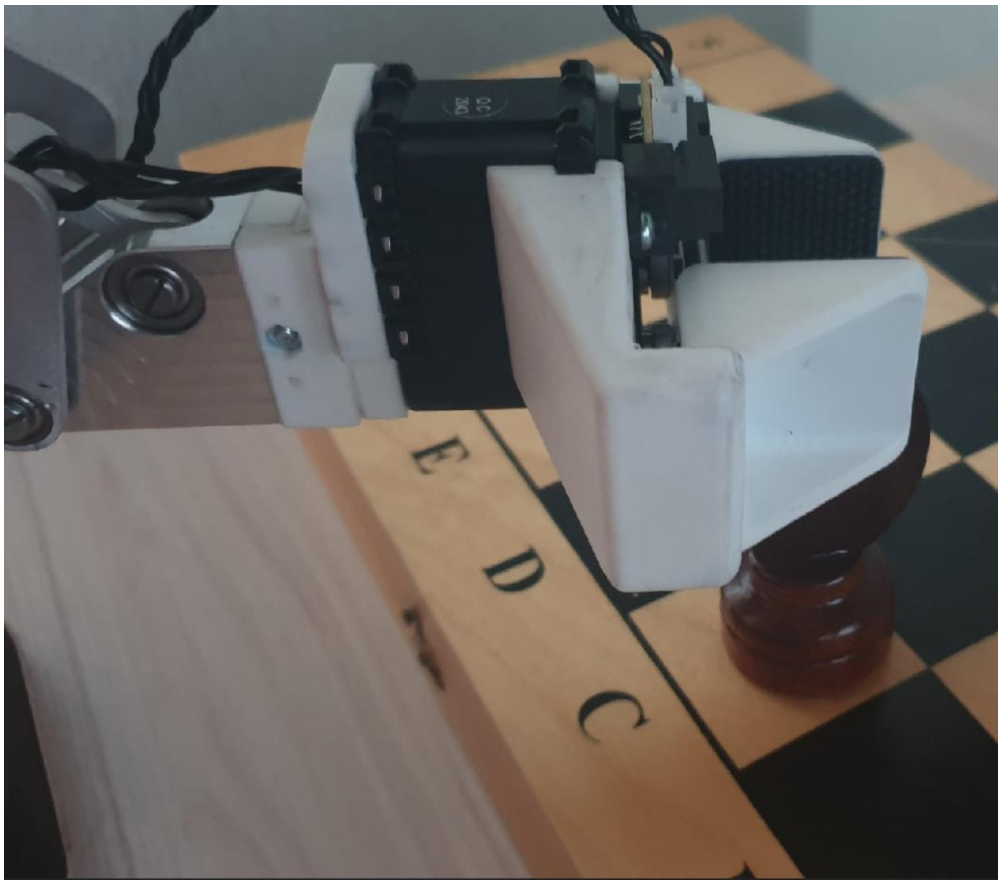
Вот один из примеров кода для решения задачи для одного хода при помощи координат, записанных в код заранее:

```
//Подключение библиотеки
#include <DynamixelWorkbench.h>
```

```
//Требуется для инициализации
#define DEVICE_NAME  "3"

//Боудрейт, для моторов - 1000000, для Serial порта - 57600
#define BAUDRATE  1000000
#define SERIAL_BAUDRATE 57600

//ID моторов, должны идти по возрастанию
#define DXL_ID1    1
#define DXL_ID2    2
#define DXL_ID3    3
#define DXL_ID4    4

//Создание рабочего пространства servos
DynamixelWorkbench servos;

void setup() {
    const char *log;

    Serial.begin(SERIAL_BAUDRATE);

    //Инициализируем сервоприводы
    servos.init(DEVICE_NAME, BAUDRATE);

    //"Включаем" сервоприводы
    servos.ping(DXL_ID1, 0, 0);
    servos.ping(DXL_ID2, 0, 0);
    servos.ping(DXL_ID3, 0, 0);
    servos.ping(DXL_ID4, 0, 0);

    //Включение сервоприводов для движения: (ID, скорость, ускорение)
    servos.jointMode(DXL_ID1, 40, 40);
    servos.jointMode(DXL_ID2, 40, 40);
    servos.jointMode(DXL_ID3, 40, 40);
    servos.jointMode(DXL_ID4, 80, 80);

    //Отправляем команды с начальной позицией на сервоприводы
    servos.goalPosition(DXL_ID1, 500);
    servos.goalPosition(DXL_ID2, 500);
    servos.goalPosition(DXL_ID3, 500);
    servos.goalPosition(DXL_ID4, 500);

    //Ждем 5 секунд, чтобы робот успел достигнуть ее и захватил коня
    delay (5000);

    //Захватываем шахматную фигуру
    servos.goalPosition(DXL_ID4, 900);
    delay (5000);
    //Двигаемся к правильной позиции
    servos.goalPosition(DXL_ID3, 460);
    servos.goalPosition(DXL_ID2, 550);
    servos.goalPosition(DXL_ID1, 420);
    delay (2000);
    //Размыкаем захват с фигурой
    servos.goalPosition(DXL_ID4, 500);
}

void loop() {
}
```

Данный код, для нашей конфигурации манипулятора, делает ход конем с E7 на C8.





«Умный» захват

Цель

Схватить предмет при приближении к манипулятору.

Описание

Движение – это жизнь! Но жизнь без ощущений довольно скучна. И довольно нерезультативна. Было бы гораздо приятнее отдавать себе отчет о том, что мы делаем и что вообще вокруг происходит. При работе с манипулятором принцип в общем-то тот же самый. Работать будет гораздо проще, если мы узнаем, что есть перед манипулятором и есть ли оно вообще там.

Решение

Решение достаточно простое. У нас есть аналоговый ИК датчик. С него, через аналоговый вход микроконтроллера, мы можем снимать показания о характере пространства перед этим самым датчиком. Если перед ним появляется что-то темное и оно близко, тогда смыкаем захват. В ином случае мы его размыкаем. Прямо как краб. Только без ног...

Одна из возможных реализаций:

```
//Подключение библиотеки
#include <DynamixelWorkbench.h>

//Требуется для инициализации
#define DEVICE_NAME  "3"

//Боудрейт, для моторов - 1000000, для Serial порта - 57600
#define BAUDRATE  1000000
#define SERIAL_BAUDRATE 57600

//ID моторов, должны идти по возрастанию
#define DXL_ID1    1
#define DXL_ID2    2
#define DXL_ID3    3
#define DXL_ID4    4

#define IR_SENSOR_PIN  A8
```

```
//Создание рабочего пространства servos
DynamixelWorkbench servos;

void setup() {
    const char *log;

    //Задаем вход инфракрасного датчика
    pinMode(IR_SENSOR_PIN, INPUT);

    Serial.begin(SERIAL_BAUDRATE);

    //Инициализируем сервоприводы
    servos.init(DEVICE_NAME, BAUDRATE);

    //"Включаем" сервоприводы
    servos.ping(DXL_ID1, 0, 0);
    servos.ping(DXL_ID2, 0, 0);
    servos.ping(DXL_ID3, 0, 0);
    servos.ping(DXL_ID4, 0, 0);

    //Включение сервоприводов для движения: (ID, скорость, ускорение)
    servos.jointMode(DXL_ID1, 40, 40);
    servos.jointMode(DXL_ID2, 40, 40);
    servos.jointMode(DXL_ID3, 40, 40);
    servos.jointMode(DXL_ID4, 80, 80);

    //Отправляем команды с начальной позицией на сервоприводы
    servos.goalPosition(DXL_ID1, 500);
    servos.goalPosition(DXL_ID2, 500);
    servos.goalPosition(DXL_ID3, 500);
    servos.goalPosition(DXL_ID4, 500);

    //Ждем 5 секунд, чтобы робот успел достигнуть ее
    delay (5000);
}

void loop() {
    //Если значение с датчика более 900, тогда зажать объект. Иначе -
разжатое состояние
    //Все коэффициенты подбираются под объект, к примеру, если поставить
вместо 700 - 1000, тогда манипулятор полностью сожмет захват
    //Будьте аккуратны!
    if (analogRead(IR_SENSOR_PIN) > 900) {
        servos.goalPosition(DXL_ID4, 700);
        //Для устранения помех, создается короткая задержка, не влияющая на
работу (50-100 мс)
        delay (100);
    }
    else
        servos.goalPosition(DXL_ID4, 500);
}
```

Возможно, что вы столкнетесь с некоторыми проблемами, такими как:

- Не выходит снять показания с датчика.

Помните, что датчик аналоговый, а не цифровой. Для снятия показаний с такого датчика требуются команды, отличные от команд снятия показаний цифровых датчиков.

- Захват смыкаются раньше времени.

Попробуйте сменить границу срабатывания датчика, или добавьте задержку перед командой захвата.

- Захват не смыкается полностью/смыкается слишком сильно

Отрегулируйте позицию, в котором захват принимает крайнее положение. Или возьмите предмет побольше/ поменьше.

“Салочки”

Цель

Экспериментально доказать идею о том, что движение – это жизнь.

Описание

Теперь мы умеем работать с ИК датчиком, так почему бы не пойти чуть дальше простого захвата? В конце концов, можно немного поиграть. Помните, как играть в салочки? Тут принцип такой же. Только водящий всегда вы. Требуется, чтобы манипулятор отводил захват при приближении объекта к ИК датчику. А если ему будет некуда двигаться, то пусть двигается в другую сторону.

Решение

При обнаружении объекта робот приводит себя в движение, уходя от объекта. Если объекта перед ним нет, то ничего не происходит. Ну а если робот достиг одного из крайних положений, то меняется направление движения. Главное двигаться по шагам и не отправлять робота сразу в крайнее положение.

Пример кода, в котором захват убегает от руки, по одной из оси:

```
//Подключение библиотеки
#include <DynamixelWorkbench.h>

//Требуется для инициализации
#define DEVICE_NAME  "3"

//Боудрейт, для моторов - 1000000, для Serial порта - 57600
#define BAUDRATE  1000000
#define SERIAL_BAUDRATE 57600

//ID моторов, должны идти по возрастанию
#define DXL_ID1    1
#define DXL_ID2    2
#define DXL_ID3    3
#define DXL_ID4    4

#define IR_SENSOR_PIN  A8
```

```
//Флаг направления
bool direct = true;
//Текущая позиция
int32_t pos = 500;

//Создание рабочего пространства servos
DynamixelWorkbench servos;

void setup() {
    const char *log;

    //Задаем вход инфракрасного датчика
    pinMode(IR_SENSOR_PIN, INPUT);

    Serial.begin(SERIAL_BAUDRATE);

    //Инициализируем сервоприводы
    servos.init(DEVICE_NAME, BAUDRATE);

    //"Включаем" сервоприводы
    servos.ping(DXL_ID1, 0, 0);
    servos.ping(DXL_ID2, 0, 0);
    servos.ping(DXL_ID3, 0, 0);
    servos.ping(DXL_ID4, 0, 0);

    //Включение сервоприводов для движения: (ID, скорость, ускорение)
    servos.jointMode(DXL_ID1, 50, 50);
    servos.jointMode(DXL_ID2, 40, 40);
    servos.jointMode(DXL_ID3, 40, 40);
    servos.jointMode(DXL_ID4, 80, 80);

    //Отправляем команды с начальной позицией на сервоприводы
    servos.goalPosition(DXL_ID1, pos);
    servos.goalPosition(DXL_ID2, 500);
    servos.goalPosition(DXL_ID3, 500);
    servos.goalPosition(DXL_ID4, 500);

    //Ждем 5 секунд, чтобы робот успел достигнуть ее
    delay (5000);

    //Увеличиваем скорость и ускорение для первой оси до максимального
значения
    servos.jointMode(DXL_ID1, 0, 0);
}

void loop() {
    //Если перед манипулятором объект, такой, что ИК датчик выдает сигнал
высотой 800,
    //тогда проверяем флаг направления и двигаемся до тех пор, пока не
достигнем одного из крайних положений, чтобы сменить направление
    //или до тех пор, пока находится объект перед манипулятором
    //(Если объект есть - есть движение, если объект пропадает -
манипулятор останавливается)
    if (analogRead(IR_SENSOR_PIN) > 800) {
        if (direct == true)
            pos++;
        else
            pos--;
        if (pos > 800)
            direct = false;
```

```
        if (pos < 300)
            direct = true;
        //Для изменения скорости изменения позиции
        delay (20);
    }
    servos.goalPosition(DXL_ID1, pos);
}
```

Попробуйте добавить ему движения и по другим осям и в случайных направлениях. Так будет интереснее с ним играть!

Движение по прямой

Цель

Показать возможность манипулятора совершать прямолинейные движения с учётом вращения звеньев

Описание

Вот манипулятор. И сразу вопрос - возможно ли движение вращения превратить в движение прямолинейное?

Оказывается можно, но нужно для этого приложить некоторые усилия и подумать над конструкцией. Но так как конструкция у нас уже есть, то почему бы не начать двигаться прямо?

Решение

Здесь Вам предстоит вспомнить тригонометрию и решить обратную задачу кинематики.

Обратная задача кинематики – это вычисление углов по заданному положению в пространстве и известной схеме кинематики. Вам повезло, все размеры уже имеются и высчитывать их не придется, но, если есть непреодолимое желание сделать все самому – это даже лучше!

Эксперименты всегда желательны в Вашем деле!

Обычно в интернете разбирают обратные задачи кинематики на плоскости, так что стоит начать с простого и управлять только двумя сервоприводами, этого будет достаточно, чтобы понять суть, но из-за необычной конструкции данного манипулятора, это будет представлять большую сложность, немногие справятся с данным заданием, сможете ли Вы это сделать? Удачи!

- У меня получилось несколько ответов, что делать в такой ситуации?

Такой вариант возможен. Лучше всего – начертить рисунок на бумаге с полученными углами, вы поймете, что один из вариантов – неправильный, он сломает манипулятор.

“Круги Эйлера”

Цель

Показать возможности манипулятора в качестве инструмента для рисования

Описание

Человеку нужен циркуль для рисования кругов? Если вы очень умелый художник, то нет. А манипулятору нужен циркуль для рисования кругов? Если это очень неумелый манипулятор, то да. Однако большинство манипуляторов точно исполняют команды, которые в них заложили программисты. Так почему бы не научить манипулятор рисовать что-то сложнее линии?

Решение

Конечно, рисовать линию очень сложно, так что стоит начать с простого – рисования рабочим органом манипулятора в пространстве, чтобы облегчить задачу.

Вам потребуется вспомнить формулу круга ($(x-x_0)^2+(y-y_0)^2=R^2$) и применить формулу обратной задачи кинематики, вы же получили ее в прошлой задаче, так ведь?

Подумайте, в какой плоскости робот сможет двигать рабочим органом по окружности?

Также не забываем про ограничения, чтобы не сломать манипулятор!