УДК 621.979.067 https://doi.org/10.26160/2309-8864-2020-9-89-92

УЛУЧШЕНИЕ ТРЕХКОЛЕСНОГО ПОГРУЗЧИКА ДЛЯ ДОСТАВКИ СТАЛЬНЫХ ЗАГОТОВОК В ТЕСНЫХ ЦЕХАХ

Назаров А.А., Красило М.С., Исраелян Г.М.
Донской государственный технический университет, г.Ростов-на-Дону

Ключевые слова: робототехника, мехатронный объект, пульт управления, ODROID, матрицы.
Аннотация. Цель данной статьи – показать один из основных методов улучшения позиционирования роботов в пространстве за счет добавления дополнительных коррекционных функций сравнения ожидаемых координат и реальных при переносе материала из одной линии производства в другую. Был приведён программный код, учитывающий изменение в конструкции прототипа и протестирован в рамках соревнований.

IMPROVEMENT OF A THREE-WHEEL LOADER FOR THE DELIVERY OF STEEL BILLETS IN CRAMPED WORKSHOPS

Nazarov A.A., Krasilo M.S., Israelyan G.M.
Don state technical University, Rostov-on-don

Keywords: robotics, mechatronic facility, control panel, matrix, ODROID, matrices.
Abstract. The purpose of this article is to show one of the main methods for improving the positioning of robots in space by adding additional correction functions for comparing the expected coordinates and real ones when transferring material from one production line to another. The program code was given taking into account the change in the prototype design and tested in the framework of the competition.

При производстве или ввозе заготовок возникает задача перевоза их в цех для дальнейшей обработки, но из-за массы и огромного количества такая задача накладывает трудности при перемещении. Для автоматизации процесса погрузки и выкладки на линию был придуман трехколесный робот доставщик заготовок в САПРе – Autodesk Fusion 360 (рисунок 1).

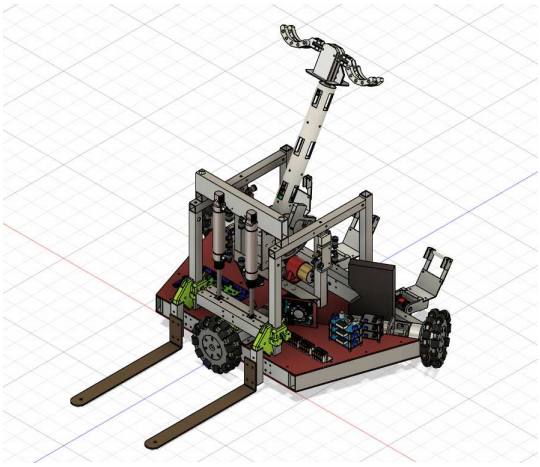


Рис. 1. Модель погрузчика

Он имеет лыжи, работающие от пневмоцилиндров, для перевоза коробок с грузом и оснащен манипулятором для их выкладки на конвейерную линию. Но для массовости производства потребуется занять все большее пространство цеха. В связи с этим данная конструкция робота в таком тесном помещении приведет к неизбежному столкновению с функциональными частями производственных станков, конвейерных линий. Тщательный анализ решения проблемы привел к модификации уже имеющейся конструкции (рисунок 2). В основание было добавлено два колеса-ленивца. Сами по себе такие колеса не двигаются от двигателя постоянного тока, они выполняют роль считывания угла от установленного на них энкодеров.

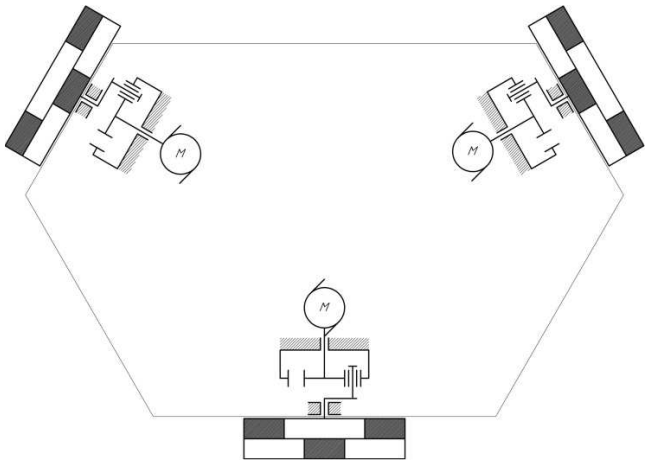


Рис. 2. Основание трехколесной платформы (кинематическая схема)

Таким образом, доработанная модель будет выглядеть иначе (рисунок 2). Она потребует также и программной доработки.

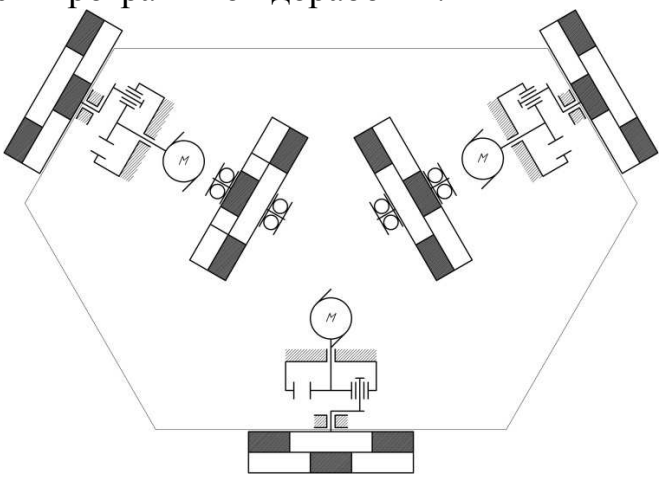


Рис. 2. Модификация погрузчика(кинематическая схема)

Для осуществления управления использован уже ранее установленный микрокомпьютер ODROID (рисунок 3).

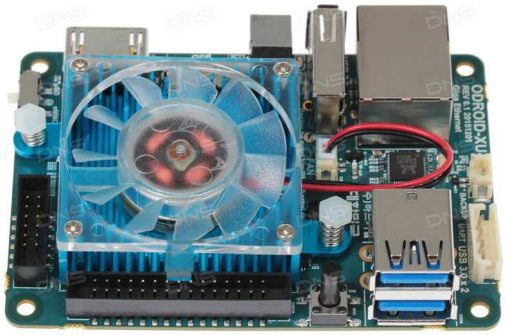


Рис. 3. ODROID, внешний вид

В программном продукте EmBitz 1.11, в котором была написана программа для ранней модели погрузчика, вносим корректировку.

Установленные два энкодера на счётных колесах должны как-то учитываться программой.

Начнём с их инициализации:

uint32_t * encCnt[5] ={ENCODER1_CNT, ENCODER2_CNT, ENCODER3_CNT, ENCODER4_CNT, ENCODER5_CNT};

Четвертый энкодер контролирует правильность данных первого, в свою очередь пятый контролирует второй. Малейшая разность между показаниями сразу будет учтена и поменяется значение скорости одного или двух двигателей.

Так как счётные колеса находятся ближе к центру по радиусу, соответственно при одной скорости двигателя их угловая скорость будет меньше, это нужно учитывать в программном коде.

encCnt[4]=1,2*encCnt[4];

encCnt[5]=1,2*encCnt[5];

Коэффициент 1,2 - отношение радиуса движущегося колеса к счётному.

Теперь разность будет считаться как для работающих колес. Следующим шагом будет настройка регулятора, поскольку перед расчетом нужно вычислить разность в показаниях.

encCntError[1]=encCnt[1]-encCnt[4];

encCntError[2]=encCnt[2]-encCnt[5];

encCnt[1]= encCnt[1]- encCntError[1];

encCnt[2]= encCnt[2]- encCntError[2];

Выше написанный участок кода будет получать разницу углов при проскальзывании колёс и вносить корректировку в данные. Следующим шагом угол энкодера будет преобразовываться в скорость на двигателе через регулятор ПИД (рисунок 4).

```
////////////////////////////////////////////////////////////////////////////////
void pidCalc(PidStruct *pid_control)  //расчет ПИД
{
  float error, dif_error;
  error = pid_control->target - pid_control->current;
  dif_error = error - pid_control->prev_error;
  pid_control->prev_error = error;
  pid_control->sum_error += error;

  if (pid_control->sum_error > pid_control->max_sum_error)
    pid_control->sum_error = pid_control->max_sum_error;
  if (pid_control->sum_error < -pid_control->max_sum_error)
    pid_control->sum_error = -pid_control->max_sum_error;

  if (pid_control->pid_on)
  {
    pid_control->output = ((float)(pid_control->p_k * error)+(pid_control->i_k * pid_control->sum_error)+
           (pid_control->d_k * dif_error));


    if (pid_control->output > pid_control->max_output)
      pid_control->output = pid_control->max_output;
    else if (pid_control->output < -pid_control->max_output)
      pid_control->output = -pid_control->max_output;

    if (pid_control->output < pid_control->min_output && pid_control->output > -pid_control->min_output)
      pid_control->output = 0;

    if ((pid_control->output <= pid_control->pid_output_end) &&(
        pid_control->output >= -pid_control->pid_output_end) &&(
        error <= pid_control->pid_error_end) && (error >= -pid_control->pid_error_end))
      pid_control->pid_finish = 1;
    else
      pid_control->pid_finish = 0;
  }
  else
  {
    pid_control->output = 0;
    pid_control->pid_finish = 0;
  }
}
```

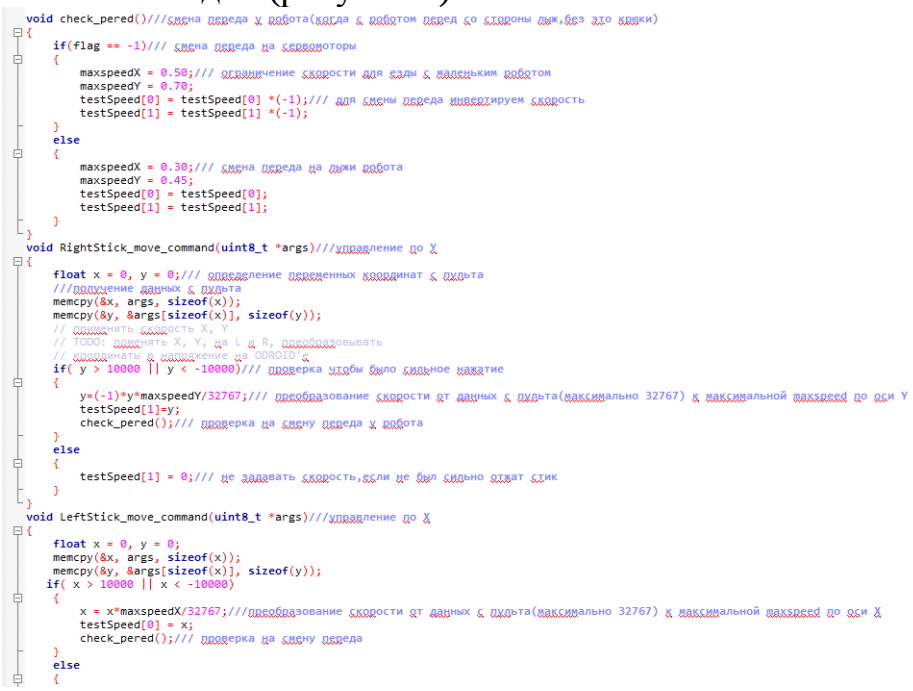
Рис. 4. Участок кода рассчёта ПИД

На выходе регулятора регулятора получим разность между требуемыми значениями двигателя и текущими показаниями.

Программный код гораздо массивнее приложенного выше, в рамках исследования был приведен нужный участок.

Для безопасности погрузчика было принято решение при перевозке с грузом вводить режим с пониженной скорости движения, когда груз будет доставлен, скорость снова увеличится. Такое решение позволит не только быстро

переместится до груза, но и с максимально возможной скоростью доставить заготовки до зоны выкладки (рисунок 5).



```
void check_pered()///смена переда у робота(когда с роботом перед со стороны лыж,без это крюки)
{
    if(flag == -1)/// смена переда на сервомоторы
    {
        maxspeedX = 0.50;/// ограничение скорости для сады с маленьким роботом
        maxspeedY = 0.70;
        testSpeed[0] = testSpeed[0] *(-1);/// для смены переда инвертируем скорость
        testSpeed[1] = testSpeed[1] *(-1);
    }
    else
    {
        maxspeedX = 0.30;/// смена переда на лыжи робота
        maxspeedY = 0.45;
        testSpeed[0] = testSpeed[0];
        testSpeed[1] = testSpeed[1];
    }
}
void RightStick_move_command(uint8_t *args)///управление по Х
{
    float x = 0, y = 0;/// определение переменных координат с пульта
    ///получение данных с пульта
    memcpy(&x, args, sizeof(x));
    memcpy(&y, &args[sizeof(x)], sizeof(y));
    // ограничить скорость X, Y
    // TODO: принимать X, Y, да L и R, комбинировать
    // команды с нажатием на ODROID
    if( y > 10000 || y < -10000)/// проверка чтобы было сильное нажатие
    {
        y=(-1)*y*maxspeedY/32767;/// преобразование скорости от данных с пульта(максимально 32767) к максимальной maxspeed по оси Y
        testSpeed[1]=y;
        check_pered();/// проверка на смену переда у робота
    }
    else
    {
        testSpeed[1] = 0;/// не задавать скорость,если не был сильно отжат стик
    }
}
void LeftStick_move_command(uint8_t *args)///управление по Х
{
    float x = 0, y = 0;
    memcpy(&x, args, sizeof(x));
    memcpy(&y, &args[sizeof(x)], sizeof(y));
   if( x > 10000 || x < -10000)
    {
        x = x*maxspeedX/32767;///преобразование скорости от данных с пульта(максимально 32767) к максимальной maxspeed по оси Х
        testSpeed[0] = x;
        check_pered();/// проверка на смену переда
    }
    else
    {
```

Рис. 5. Программное решение для пульта управления

Робот-погрузчик был успешно протестирован в рамках всероссийских робототехнических соревнований «Econet18+» и занял призовое место. Подбор в рамках фестиваля груз в виде перерабатываемых материалов можно применить в будущем для стальных заготовок и/или готовых изделий на производстве.

Возможное решение для модификации такого прототипа это добавление счётных колёс на пружинах, что позволит считывать угол энкодера с наименьшим трением колёс об поверхность, по которой происходит движение.

Список литературы

1. Марченко А.Л. Основы электроники. Учебное пособие для вузов. – М.: ДМК Пресс, 2008. – 296 с.
2. Подбельский В.В., Фомин С.С. Курс программирования на языке Си: учебник. – М.: ДМК Пресс, 2012. – 384 с.
3. Литвиненко Н.А. Технология программирования на C++. Win32 API-приложения. – СПб.: БХВ-Петербург, 2010. – 288 с.
4. Назаров А.А., Красило М.С. Робот-погрузчик // Научный потенциал молодежи и технический прогресс: Материалы III международной студенческой научно-практической конференции 15 мая 2020 г. – СПб: СПбФ НИЦ МС, 2020. – С. 58-60.
5. Красило М.С., Назаров А.А. Ввод команд управления с помощью ПУ XBOX 360 // Научный потенциал молодежи и технический прогресс: Материалы III международной студенческой научно-практической конференции 15 мая 2020 г. – СПб: СПбФ НИЦ МС, 2020. – С. 21-24.

Сведения об авторах:

Назаров Александр Александрович – студент, ДГТУ, г.Ростов-на-Дону;

Красило Михаил Сергеевич – студент, ДГТУ, г.Ростов-на-Дону;

Исраелян Гарри Михайлович – студент, ДГТУ, г.Ростов-на-Дону.