Министерство науки и высшего образования
Российской Федерации
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технологический университет «СТАНКИН»
(ФГБОУ ВО «МГТУ «СТАНКИН»)

Институт информационных систем и технологий
Кафедра робототехники и мехатроники, Кафедра сенсорных и управляющих систем
Дисциплина: «Сенсорные и управляющие системы»

ОТЧЁТ
по лабораторной работе №2
на тему:
Подключение цифровых и аналоговых датчиков к микропроцессору. Фильтрация шума.

Выполнили:
студент группы АДМ-21-05 _____ Кайда А.С.
(подпись) (Ф.И.О)

студент группы АДМ-21-05 _____ Абдулзагиров М.М.
(подпись) (Ф.И.О)

студент группы АДМ-21-05 _____ Басов А.Д.
(подпись) (Ф.И.О)

Принял
преподаватель _____ _____ Андреев В.П.
(Дата) (подпись) (Ф.И.О)

Оценка:____ Дата:_____

Москва 2022

Оглавление

Цель работы:

а) исследование аналогового датчика на примере инфракрасного датчика (ИК-датчика) расстояния типа GP2D120 (Sharp);

б) исследование цифрового датчика на примере ультразвукового датчика (УЗ-датчика) расстояния типа HC-SR04.

Задачи:

1) Подключить к микроконтроллеру ИК-датчик расстояния типа GP2D120 (Sharp) и обеспечить отображение показаний датчика на экране ПК при изменении расстояния до объекта.Построить характеристику преобразования ИК-датчика расстояния в диапазоне расстояний, указанном в паспортных данных для плоского препятствия, ориентированного перпендикулярно ориентации датчика (идеальный случай), с использованием фильтрации шума по среднему арифметическому значению при заданном количестве измерений N. Найти аппроксимирующую функцию (степенная, экспоненциальная) и вычислить аппроксимирующей функции. Выполнить фильтрацию шума для заданного расстояния до объекта (для каждой подгруппы студентов задаётся разный набор расстояний) с использованием 5 фильтров: среднее арифметическое, среднее геометрическое, медианный фильтр, среднее гармоническое и фильтр срединной точки. Вычислить отношение сигнал/шум для 5 результатов фильтрации (при едином для всех фильтров окне фильтрации N) и выбрать оптимальный фильтр (максимум отношения сигнал/шум).

2) Подключить к микроконтроллеру УЗ-датчик расстояния типа HC-SR04 и обеспечить отображение показаний датчика на экране ПК при изменении расстояния до объекта. Построить характеристику преобразования УЗ-датчика расстояния в диапазоне расстояний, указанном в паспортных данных для плоского препятствия, ориентированного перпендикулярно ориентации датчика (идеальный случай), с использованием фильтрации шума по среднему арифметическому значению при заданном количестве измерений N. Выполнить фильтрацию шума для 7 значений расстояния до объекта (для каждой подгруппы задаётся разный набор расстояний) с использованием 5 фильтров: среднее арифметическое, среднее геометрическое, медианный фильтр, среднее гармоническое и фильтр срединной точки. Вычислить отношение сигнал/шум для 5 результатов фильтрации (при едином заданном окне фильтрации N) и выбрать оптимальный фильтр (максимум отношения сигнал/шум).

1.1. ИК-датчик расстояния GP2D120 (Sharp).

Технические характеристики ИК-датчика приведены в табл. 1.

Таблица 1 - Технические характеристики датчика GP2D120 (Sharp).

Параметр	MIN.	TYP.	MAX.	Единица измерения
Напряжение питания	4.5	5	5.5	В
Температурный диапазон работы	-10	+20	+60	°C
Температурный диапазон хранения	-40		+70	°C
Диапазон измеряемого расстояния	4	—	30	см
Диапазон напряжения на аналоговом выходе	0.25	—	3.2	В
Потребление тока	—	33	50	мА

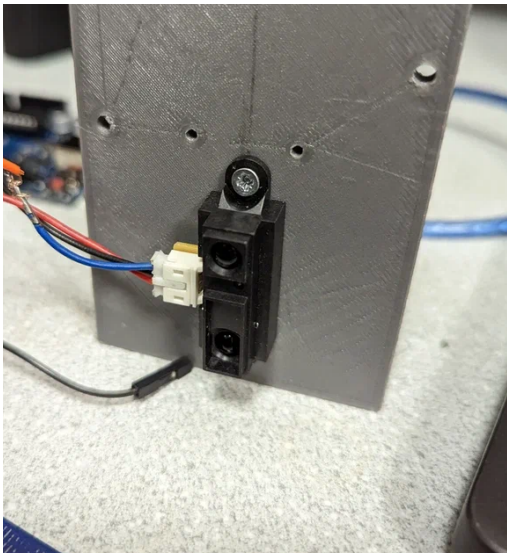

Рисунок 1 – Закрепленный ИК-датчик расстояния.

Схема эксперимента показана на рис. 2.

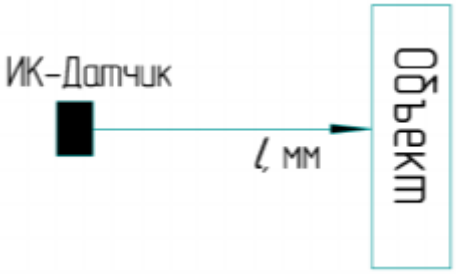

Рисунок 2 – схема эксперимента.

1.2. Подключение ИК-датчиков GP2D120 (Sharp) к микропроцессору.

Таблица 2 - Технические характеристики МК Arduino Uno

Микроконтроллер	ATmega328P
Ядро	8-битный AVR
Тактовая частота	16 МГц
Flash-память	32 КБ
RAM-память	2 КБ
EEPROM-память	1 КБ
Пины ввода-вывода	20
Пины с прерыванием	2
Пины с АЦП	6
Разрядность АЦП	10 бит
Аппаратные интерфейсы	1× UART, 1× I²C, 1× SPI
Напряжение логических уровней	5В
Входное напряжение питания:	- через USB: 5В - через DC-разъём или пин Vin: 7,5–12 В
Максимальный выходной ток пина 3V3	150 мА
Максимальный выходной ток пина 5V	1 А
Размеры	69×53 мм

Принципиальная электрическая схема подключения ИК-датчика показаны на рис. 3

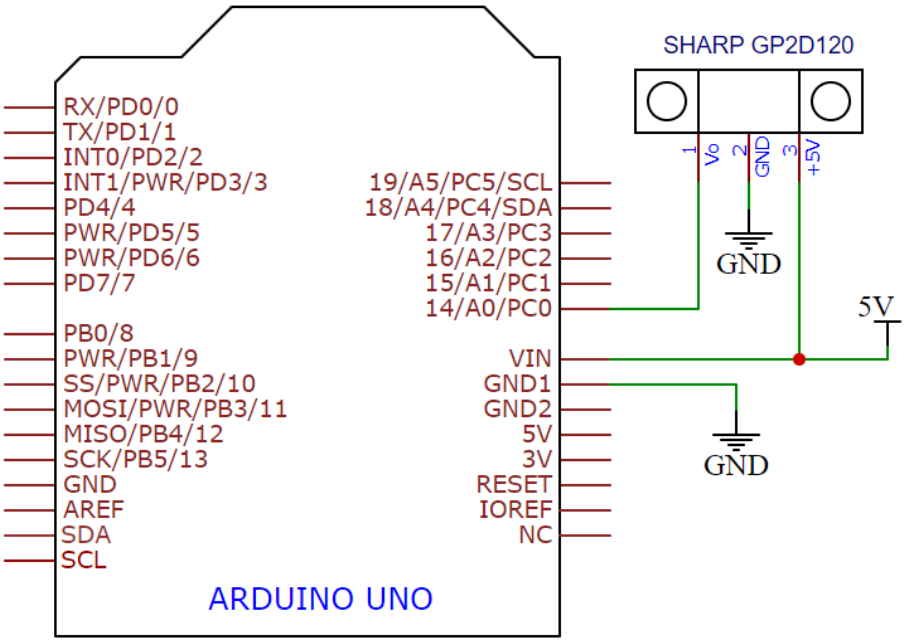

Рисунок 3 - Принципиальная электрическая схема подключения ИК-датчика.

Подключение ИК-датчика к МК во время эксперимента показано на рис. 4

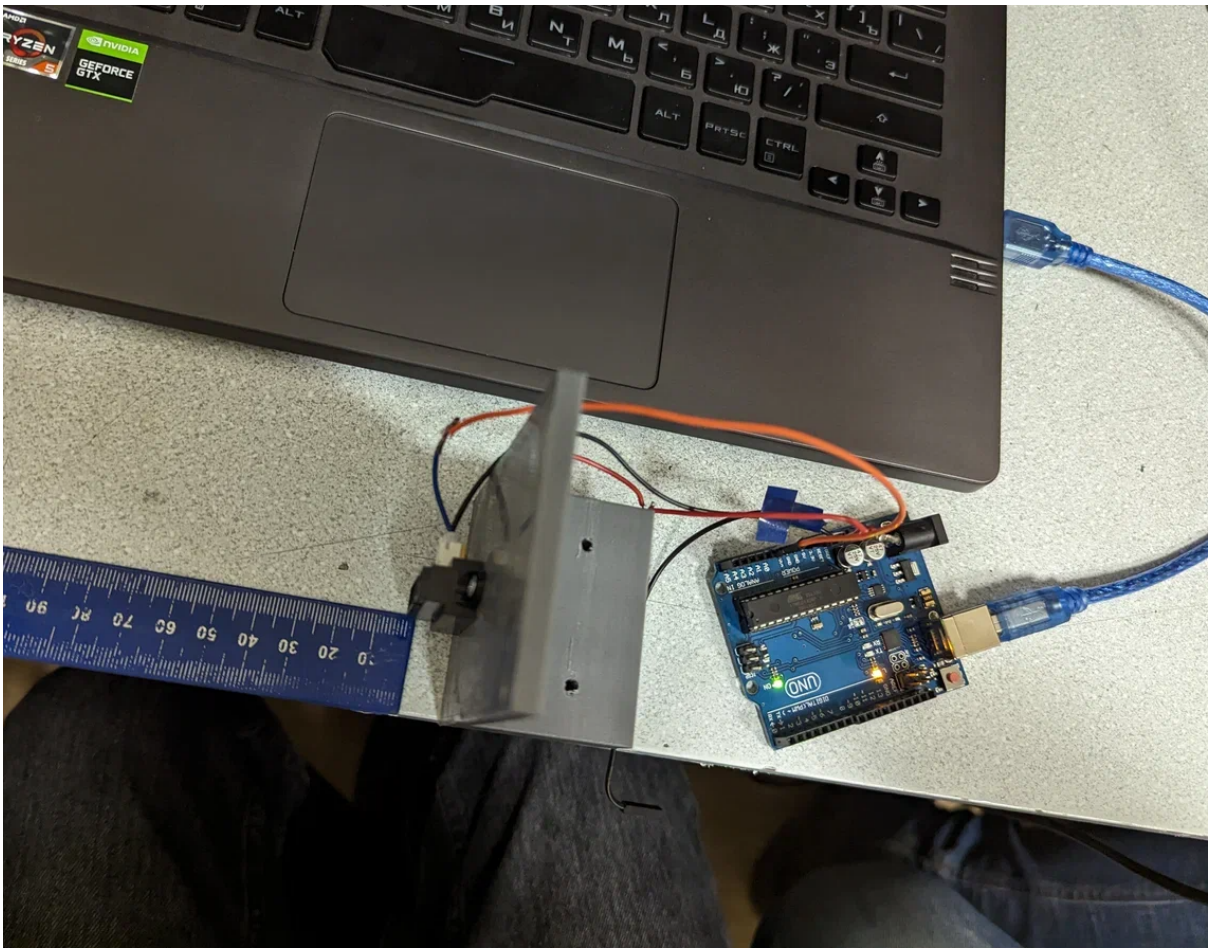

Рисунок 4 – Подключенный ИК-датчик к МК.

Программа для получения показаний ИК-датчика:

Листинг 1 – Код программы управления опроса ИК-датчика.

```
float IRpin = A0;// аналоговый вход для подключения выхода Vo сенсора
void setup()
{
Serial.begin(9600);// запуск последовательного порта
}
void loop(){
// 5V/1024 = 0.0048828125                    //считываем значение сенсора и переводим
float volts = analogRead(IRpin)*0.0048828125; // в напряжение в диапазоне 0-5В
Serial.println(volts); //выдаём значение в порт
delay(50);
}
```

Задание 1 Калибровка датчика.

Цель: построить зависимость показаний (в цифровой форме в диапазоне 10-разрядного АЦП микропроцессора) на выходе датчика Sharp GP2D120 от расстояния до объекта. Определить параметры a и b аппроксимирующей функции (1).

Код программы управления опроса ИК-датчика показан на листинге 2. В данной программе реализуется вычисление математического ожидания и дисперсии на основе выборки из 100 значений.

Листинг 2 – Код программы управления опроса ИК-датчика.

```
#define IRpin A0
void setup()
{
        pinMode(IRpin, 0);
        Serial.begin(9600);
}
void loop()
{
        long val = 0;
        long N = 100; // выборка на основе 100 значений
        int vals[100];

        // считывание показаний АЦП
        for (int i = 0; i < N ; i++)
        {
                vals[i] = (long)analogRead(IRpin);
                delay(10);
        }
        // вычисление математического ожидания
        long val = 0;
        for (int i = 0; i < N ; i++)
```

```
        {
                val += vals[i];
        }
        float M = (float)(val / N ) * 0.0048828125;
        long D = 0;
        for (int i = 0; i < N ; i++)
        {
                D += (vals[i] - (val / N )) * (vals[i] - (val / N ));
        }
        int Df = (D / (N  - 1)); // * 0.0048828125*0.0048828125;
        Serial.print(M);
        Serial.println("\t " + String(Df));

        // Выводим информацию в COM порт
        Serial.print(M);
        Serial.println('\t');
        Serial.print(D);
}
```

Блок схема данного алгоритма показана на рис. 4.

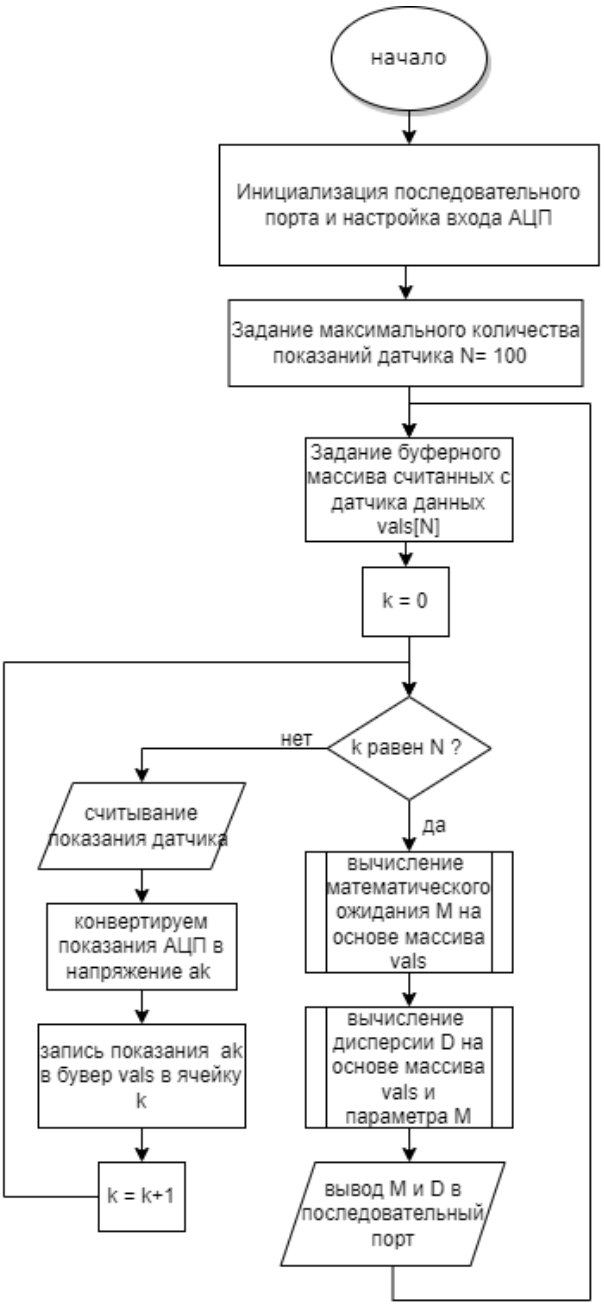

Рис. 5 - Блок-схема программы считывания показаний.

Для каждого значения расстояния l производится вычисление математического ожидания и дисперсии на основе выборки из N = 100 замеров a_k. Математическое ожидание M_l вычисляется по формуле:

$$M_l = \frac{\sum a_k}{N},$$

где a_k = analogRead(IRpin)*3.3/1024 – показание датчика (значение напряжения, полученное с помощью микропроцессора для $k = 1,2, \dots, N$); N − количество замеров.

Дисперсия D_l вычисляется по формуле:

$$D_l = \frac{\sum(a_k - M_l)^2}{N - 1}.$$

Процесс калибровки ИК-датчика показан на рис. 6.

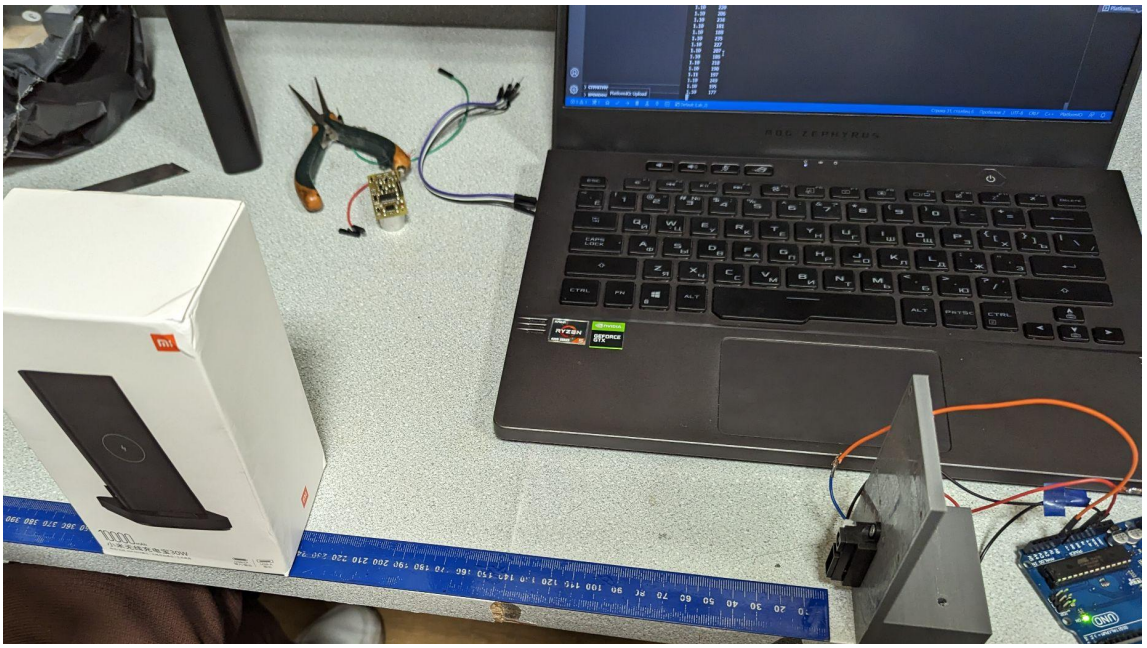

Рис. 6. Проведение эксперимента. Микроконтроллер с подключённым датчиком измеряет расстояние до белой коробки и отправляет данные на ноутбук с запущенным терминалом.

Во время калибровки на определённом расстоянии от датчика устанавливался объект и выводимые на ноутбуке показания записывались в таблицу. Зависимость значений математического ожидания (Ml) и дисперсии (Dl) данных, полученных от ИК-датчика, от расстояния от объекта до ИК-датчика приведены в табл. 3.

Таблица 3- Значения ИК-датчика.

l, мм	***Ml,*** В	***Dl,*** В
20	1,97	0,0012666
40	2,88	0,0027446
60	3,13	0,0017382
80	2,9	0,0024314
100	2,46	0,0013209
120	2,1	0,0015198
140	1,82	0,0020698
160	1,62	0,0013602
180	1,45	0,0010871

200	1,31	0,0017844
220	1,22	0,0015301
240	1,1	0,0015233
260	1,04	0,0011787
280	0,93	0,0019414
300	0,86	0,0011872
320	0,79	0,0007289
340	0,73	0,0016709
360	0,67	0,0011514
380	0,61	0,0017332
400	0,55	0,0008892
420	0,51	0,0009825
440	0,47	0,0010898
460	0,42	0,0007856
480	0,41	0,0018765
500	0,37	0,0012314

На основе полученных данных был построен график, показанный на рис. 7.

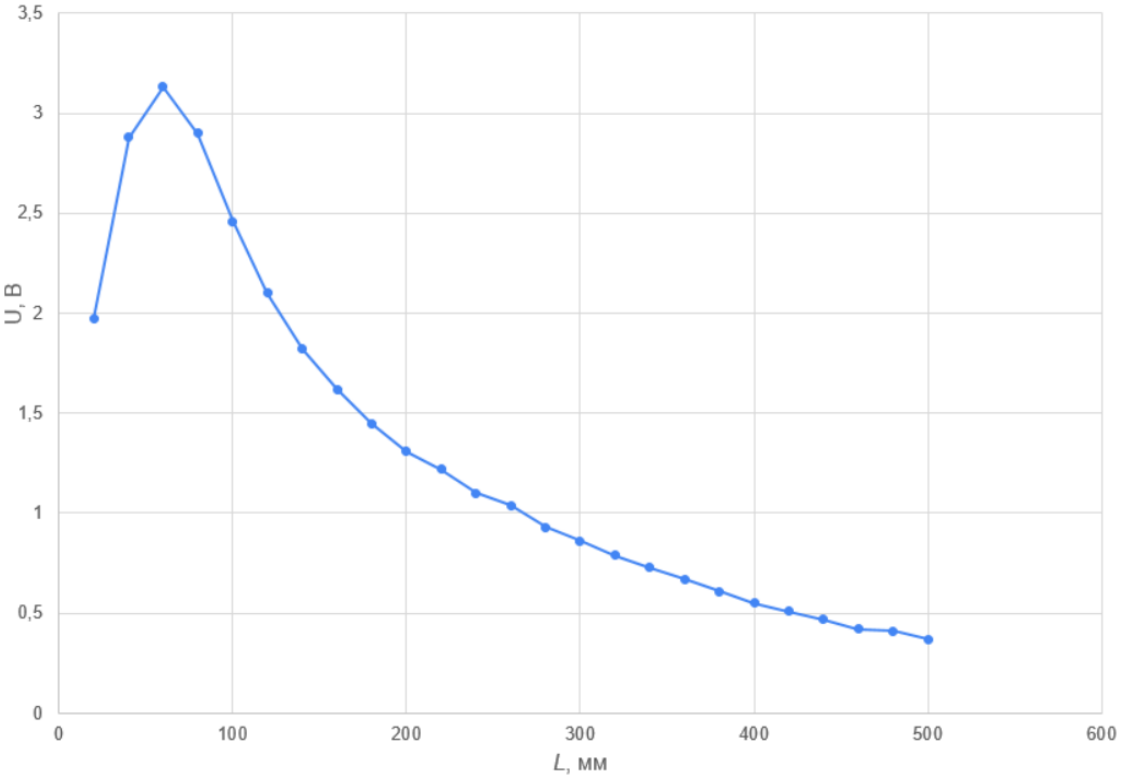

Рис. 7 – График зависимости U(*l*).

По графику видно, что существует неоднозначность получения напряжения от длины. Чтобы избавится от неоднозначности, исключим измерения на расстояниях менее 60 мм.(рис. 8). По оставшимся показаниям была получена линия тренда, изображённая пунктирной линией, и была выведена формула аппроксимирующей функции.

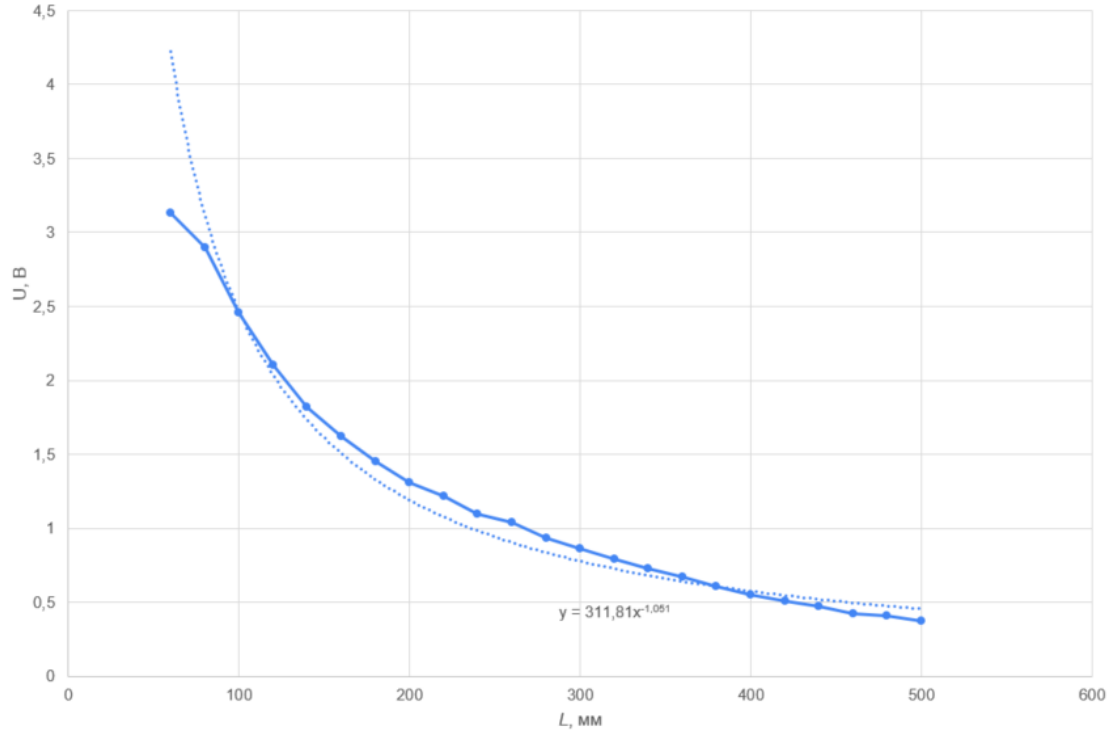

Рисунок 8 – Исправленный график зависимости U(*l*).

Зависимость напряжения(U) от расстояния(L) будет вычисляться по формуле на рис. 9.

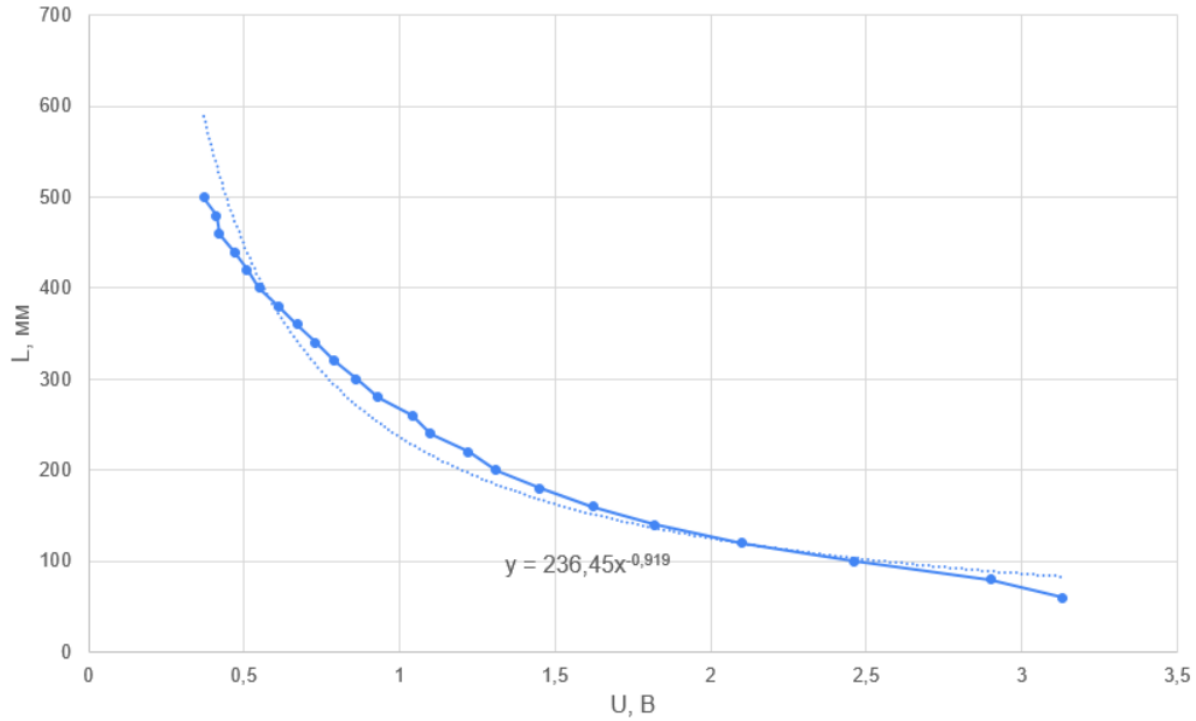

Рисунок 9 - График зависимости l(U).

Наиболее близкая к значениям линия тренда – степенная аппроксимация по формуле:

$$l = 236,45 \cdot U^{-0,919}$$

Выводы: Была выполнена калибровка ИК-датчика и получена формула для однозначного определения расстояния.

Задание 2. Фильтрация шума.

Цель: определить оптимальный фильтр по критерию отношения сигнал/шум.
Параметры расстояния после фильтрации на различных расстояниях измерения приведены в табл. 4.

Код программы управления опроса ИК-датчика показан на листинге 2. В данной программе реализуется вычисление математического ожидания и дисперсии на основе выборки из 100 значений.

Листинг 3 – Код программы управления опроса ИК-датчика.

```
#define IRpin A0
// Фильтр, основанный на вычислении среднего арифметического
int f1(int *vals, int count)
{
  long val = 0;
  for (int i = 0; i < count; i++)
  {
```

```
    val += vals[i];
  }
  // float volts = (float)(val / count) * 0.0048828125;
  return (val / count);
}

int D(int *vals, int count, int Mint)
{
  long D = 0;
  for (int i = 0; i < count; i++)
  {
    D += pow(vals[i] - Mint, 2);
  }
  int Df = (D / (count - 1)); // * 0.0048828125*0.0048828125;
  return Df;
}

// Фильтр, основанный на вычислении среднего геометрического
int f2(int *vals, int count)
{
  uint64_t temp = 1;
  for (int i = 0; i < count / 10; i++)
  {
    uint64_t tempC = 1;
    for (int j = 0; j < count / 10; j++)
    {
      tempC *= vals[i * count / 10 + j] / 10;
    }
    temp *= pow(tempC, 1.0 / ((double)count / 10.0));
  }
  return pow(temp, 1.0 / ((double)count / 10.0)) * 10; // * 0.0048828125;
}
int ArrayCompare(const int *AFirst, const int *ASecond)
{
  if (*AFirst < *ASecond)
    return -1;
  return (*AFirst == *ASecond) ? 0 : 1;
}

// Медианный фильтр
int f3(int *vals, int count)
{
  size_t M_Size = sizeof(vals) / sizeof(vals[0]);
  qsort(vals, M_Size, sizeof(vals[0]), ArrayCompare);
  return vals[50]; //* 0.0048828125;
}

// Фильтр, основанный на вычислении среднего гармонического
```

```
int f4(int *vals, int count)
{
  double val = 0;
  for (int i = 0; i < count; i++)
  {
    val += 1.0 / (double)vals[i];
  }
  return (int)(((double)count) / val); //* 0.0048828125;
}

// Фильтр срединной точки:
int f5(int *vals, int count)
{
  size_t M_Size = sizeof(vals) / sizeof(vals[0]);
  qsort(vals, M_Size, sizeof(vals[0]), ArrayCompare);
  return (vals[0] + vals[count - 1]) / 2; //* 0.0048828125;
}

void SelectionSort(int *sensArr, int count)
{
  for (int j = 0; j < count; ++j)
  {
    int temp = sensArr[j];
    int ind = j;
    for (int i = j + 1; i < count; ++i)
    {
      if (temp > sensArr[i])
      {
        temp = sensArr[i];
        ind = i;
      }
    }
    sensArr[ind] = sensArr[j];
    sensArr[j] = temp;
  }
}

void setup()
{
  pinMode(IRpin, 0);
  Serial.begin(9600);
  Serial.println("Start");
}
void loop()
{
  long count = 100;
  int durations[100];
```

```
  for (int i = 0; i < count; i++)
  {
   durations[i] = (long)analogRead(IRpin);
   delay(10);
  }

  // фильтрация показаний
  int duration1 = f1(durations, count);
  int duration2 = f2(durations, count);
  int duration3 = f3(durations, count);
  int duration4 = f4(durations, count);
  int duration5 = f5(durations, count);

double Df = D(durations, count,duration1);
 // среднеквадратичное отклонение
double Q = pow(Df,2);
// ОСШ
double Relation = M /Q ;
Serial.print(M);
Serial.println("\t " + String(Df));
// среднеквадратичное отклонение
double Q = pow(Df,2);
// ОСШ
double Relation = M /Q ;
  // Выводим информацию в COM порт
  Serial.print(duration1);
  Serial.print('\t');
  Serial.print(duration2);
  Serial.print('\t');
  Serial.print(duration3);
  Serial.print('\t');
  Serial.print(duration4);
  Serial.print('\t');
  Serial.print(duration5);
  Serial.print('\t');
  Serial.print(Df );
  Serial.print('\t');
  Serial.print(Q);
  Serial.print('\t');
  Serial.print(Relation );
}
```

Результаты эксперимента отображены в таблице 4. Обозначения, указанные в таблице:

L – эталонное расстояние до предмета.

L_f – измеренное расстояние до предмета (мм).

$L_f = 236,45 \cdot fm^{-0,919}$, где fm – отфильтрованное показание напряжения на выходе датчика (в), m = 1,2,...,5 – выбранный тип фильтра.

Δy_L – абсолютная погрешность измерений, вычисляемая по формуле:

$$\Delta y_L = |f_m - L|,$$

δ_L – относительная погрешность измерений, вычисляемая по формуле:

$$\delta_L = \frac{\Delta y_L}{f_m}.$$

D_L – дисперсия, вычисляемая по формуле:

$$D_L = \frac{\sum(a_k - f_m)^2}{N-1}.$$

σ_L – среднеквадратичное отклонение, вычисляемая по формуле:

$$\sigma_L = \sqrt{D_L}.$$

$ОСШ_L$ – отношение сигнал/шум, вычисляемая по формуле:

$$ОСШ_L = \frac{f_m}{\sigma_L}.$$

Фильтрация значений производиться следующими фильтрами:
f1 – фильтр, основанный на вычислении среднего арифметического

$$f_1 = \frac{1}{N}\sum_{k=1}^{N} a_k.$$

f2 – фильтр, основанный на вычислении среднего геометрического.

$$f_2 = \left[\prod_{k=1}^{N} a_k\right]^{\frac{1}{N}}.$$

f3 – медианный фильтр.

$$f_3 = med\{a_1, \dots, a_N\}.$$

f4 – фильтр, основанный на вычислении среднего гармонического.

$$f_4 = \frac{N}{\sum_{k=1}^{N}\frac{1}{a_k}}.$$

f5 – фильтр срединной точки.

$$f_5 = \frac{1}{2}[\max\{a_1 \dots a_N\} + \min\{a_1 \dots a_N\}].$$

В данных формулах a_k – значение входного сигнала для *k*-го отсчёта при выборке $N = 100$ $(k = 1, 2, \dots, N)$.

Таблица 4 – Экспериментальные показания.

Тип фильтра	**L, мм (линейка)**	**Lf, мм (датчик)**	**Δy_L, мм**	**δ_L**	**D_L, В^2**	**σ_L, В**	**ОСШ$_L$**
f1	**50**	64	14	0,21875	0,01816	0,1148	25,032
	100	96	4	0,041666666	0,00109	0,0330	61,320
	150	149	1	0,006711409	0,00120	0,0341	44,493
	200	194	6	0,030927835	0,00182	0,0421	26,989
	250	244	6	0,024590163	0,00108	0,0329	30,165
f2	**50**	65	5	0,076927692	0,01783	0,1135	24,690
	100	98	2	0,020408632	0,00107	0,0327	61,720
	150	148	2	0,013513513	0,00107	0,0327	43,295
	200	201	1	0,004975124	0,00209	0,0457	29,466
	250	241	9	0,037344398	0,00106	0,0325	29,128
f3	**50**	68	18	0,264705884	0,00209	0,0458	92,367
	100	99	1	0,010101010	0,00108	0,0328	54,551
	150	147	3	0,020408167	0,00107	0,0327	38,500
	200	198	2	0,010101010	0,00210	0,0458	30,559
	250	248	2	0,008064516	0,00106	0,0325	34,082
f4	**50**	67	17	0,253731343	0,01783	0,1235	22,840
	100	95	5	0,052631578	0,00107	0,0327	49,616
	150	148	2	0,013513513	0,00107	0,0327	36,270
	200	196	4	0,020408163	0,00209	0,0457	28,624
	250	249	1	0,004016064	0,00106	0,0325	32,153
f5	**50**	65	15	0,230769230	0,00836	0,1014	34,866
	100	98	2	0,020408163	0,00523	0,0723	30,860
	150	148	2	0,013513513	0,00315	0,0561	34,748
	200	194	6	0,030927835	0,00418	0,0646	21,381
	250	250	0	0	0,00210	0,0458	19,960

Выводы: Оптимальным по значению отношения сигнал/шум является медианный фильтр.

2.1 Подключение УЗ-датчика HC-SR04 к микропроцессору.

Таблица 5 - Технические характеристики датчика hc-sr04.

Параметр	MIN.	TYP.	MAX.	Единица измерения
Напряжение питания	4.8	5	5.4	В
Температурный диапазон работы	-30	+20	+80	°C
Частота ультразвука	–	40	–	кГц
Диапазон измеряемого расстояния	2	—	400	см
время измерения	–	50	–	мС
Угол измерения	–	15°	30°	В
Потребление тока	2	–	15	мА

Схема подключения УЗ-датчика показан на рисунке 10.

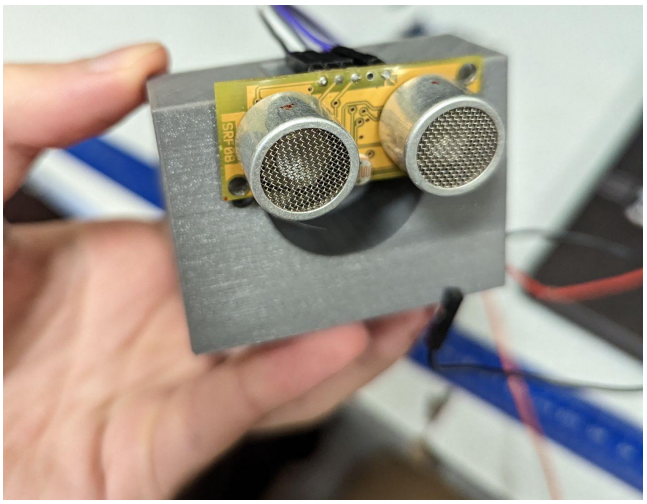

Рисунок 10 – Закрепленный УЗ-датчик расстояния.

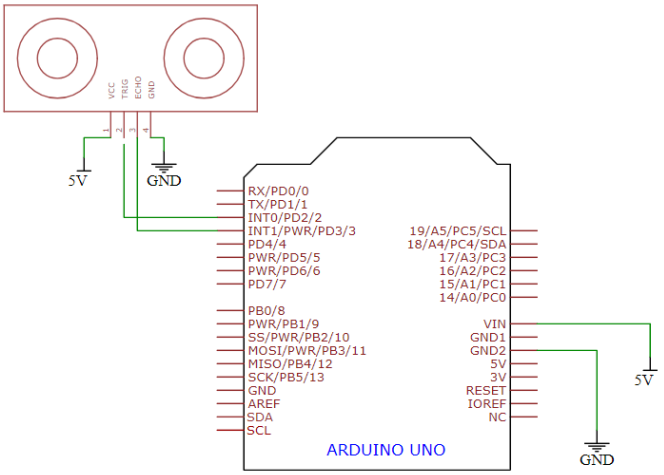

Рисунок 11 - Принципиальная электрическая схема подключения УЗ-датчика.

Программа для получения показаний УЗ-датчика:

Листинг 3 – Код программы управления опроса УЗ-датчика.

```
int echoPin = 8;
int trigPin = 9;
void setup()
{
 Serial.begin (9600); // инициализация последовательного порта
 pinMode(trigPin, OUTPUT);                // задаёт пин (триггер) на выход
 pinMode(echoPin, INPUT);                 // задаёт пин (эхо) на вход
}
void loop()
{
int duration, cm;
digitalWrite(trigPin, LOW);       // подаётся низкий сигнал в течение 2-х микросекунд
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);      // подаётся высокий сигнал в течение 10-ти микросекунд
delayMicroseconds(10);
digitalWrite(trigPin, LOW);       // опять подаётся низкий сигнал
duration = pulseIn(echoPin, HIGH);       // считывается длина принятого импульса
cm = duration / 58;         // определяется дистанция в сантиметрах
Serial.println(cm);         // выводятся данные в последовательный порт
delay (50);                 // задержка между импульсами
}
```

2.2 Задание. Фильтрация шума.

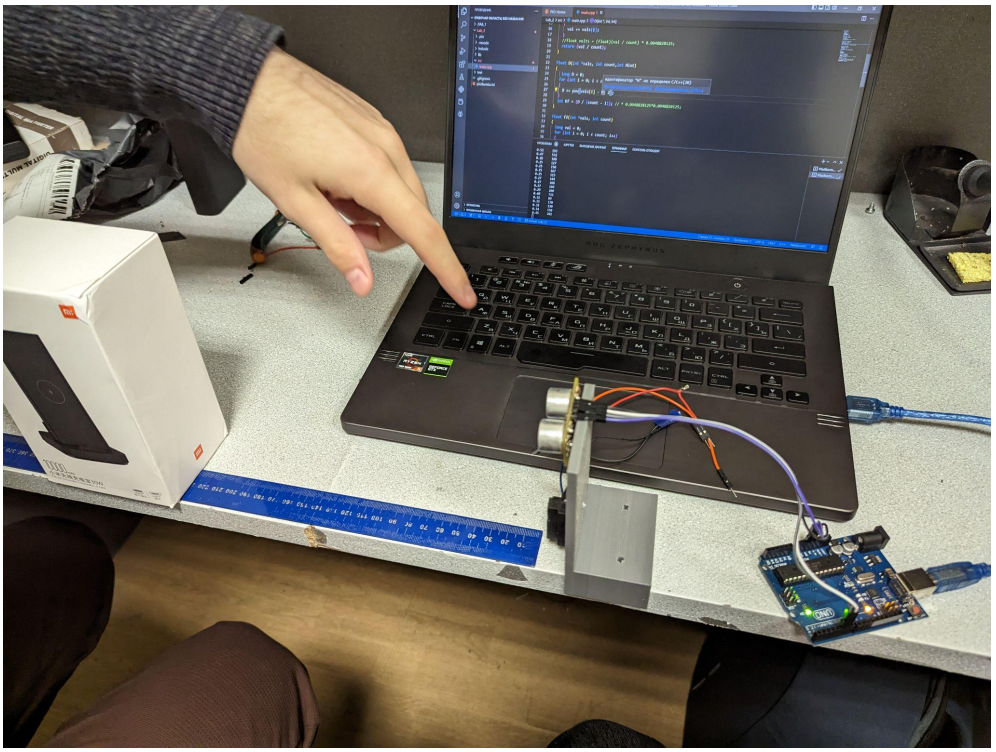

Рис. 6. Проведение эксперимента. Микроконтроллер с подключённым датчиком измеряет расстояние до белой коробки и отправляет данные на ноутбук с запущенным терминалом.

Листинг 4 – Код программы управления опроса УЗ-датчика.

```
#define echoPin 2
#define trigPin 3

// Фильтр, основанный на вычислении среднего арифметического
int f1(int *vals, int count)
{
  long val = 0;
  for (int i = 0; i < count; i++)
  {
    val += vals[i];
  }
  // float volts = (float)(val / count) * 0.0048828125;
  return (val / count);
}

int D(int *vals, int count, int Mint)
{
  long D = 0;
```

```
  for (int i = 0; i < count; i++)
  {
    D += pow(vals[i] - Mint, 2);
  }
  int Df = (D / (count - 1)); // * 0.0048828125*0.0048828125;
  return Df;
}

// Фильтр, основанный на вычислении среднего геометрического
int f2(int *vals, int count)
{
  uint64_t temp = 1;

  for (int i = 0; i < count / 10; i++)
  {
    uint64_t tempC = 1;
    for (int j = 0; j < count / 10; j++)
    {
      tempC *= vals[i * count / 10 + j] / 10;
    }
    temp *= pow(tempC, 1.0 / ((double)count / 10.0));
  }
  return pow(temp, 1.0 / ((double)count / 10.0)) * 10; // * 0.0048828125;
}

int ArrayCompare(const int *AFirst, const int *ASecond)
{
  if (*AFirst < *ASecond)
    return -1;
  return (*AFirst == *ASecond) ? 0 : 1;
}

// Медианный фильтр
int f3(int *vals, int count)
{
  size_t M_Size = sizeof(vals) / sizeof(vals[0]);
  qsort(vals, M_Size, sizeof(vals[0]), ArrayCompare);

  return vals[50]; //* 0.0048828125;
}

// Фильтр, основанный на вычислении среднего гармонического
int f4(int *vals, int count)
{
  double val = 0;
  for (int i = 0; i < count; i++)
  {
    val += 1.0 / (double)vals[i];
```

```
  }
  return (int)(((double)count) / val); //* 0.0048828125;
}

// Фильтр срединной точки:
int f5(int *vals, int count)
{
  size_t M_Size = sizeof(vals) / sizeof(vals[0]);
  qsort(vals, M_Size, sizeof(vals[0]), ArrayCompare);

  return (vals[0] + vals[count - 1]) / 2; //* 0.0048828125;
}

void SelectionSort(int *sensArr, int count)
{
  for (int j = 0; j < count; ++j)
  {
    int temp = sensArr[j];
    int ind = j;
    for (int i = j + 1; i < count; ++i)
    {
      if (temp > sensArr[i])
      {
        temp = sensArr[i];
        ind = i;
      }
    }
    sensArr[ind] = sensArr[j];
    sensArr[j] = temp;
  }
}

void setup()
{
  pinMode(IRpin, 0);
  Serial.begin(9600);
  Serial.println("Start");
}

void loop()
{
  long count = 100;
  int durations[100];

  for (int i = 0; i < count; i++)
  {
    // Генерация короткого импульса длительностью 2 микросекунды
    digitalWrite(trigPin, LOW);
```

```
  delayMicroseconds(2);

  // В течении 10 микросекунд датчик посылает сигналы в 40 кГц
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Получаем время задержки с выхода Echo
  durations[i] = pulseIn(echoPin, HIGH);
  delay(10);
}
for (int i = 0; i < count; i++)
{
  durations[i] = (long)analogRead(IRpin);
  delay(10);
}

// фильтрация показаний
double duration1 = (double )f1(durations, count) / 58.0;
double duration2 = (double )f2(durations, count)/ 58.0;
double duration3 = (double )f3(durations, count)/ 58.0;
double duration4 = (double )f4(durations, count)/ 58.0;
double duration5 = (double )f5(durations, count)/ 58.0;

double Df = D(durations, count, duration1)/ (58.0*58.0);
// среднеквадратичное отклонение
double Q = pow(Df, 2);
// ОСШ
double M = duration1;
double Relation = M / Q;
Serial.print(M);
Serial.println("\t " + String(Df));

// среднеквадратичное отклонение
double Q = pow(Df, 2);
// ОСШ
double Relation = M / Q;

// Выводим информацию в COM порт
Serial.print(duration1);
Serial.print('\t');
Serial.print(duration2);
Serial.print('\t');
Serial.print(duration3);
Serial.print('\t');
Serial.print(duration4);
Serial.print('\t');
Serial.print(duration5);
```

```
  Serial.print('\t');
  Serial.print(Df );
  Serial.print('\t');
  Serial.print(Q);
  Serial.print('\t');
  Serial.print(Relation );
}
```

Результаты эксперимента отображены в таблице 5. Обозначения, указанные в таблице:

L - эталонное расстояние до предмета.

Lf - расстояние с датчика.

Δy_L- абсолютная погрешность измерений.

δ_L - относительная погрешность измерений.

D_L- дисперсия.

σ_L - среднеквадратичное отклонение.

f1 - фильтр, основанный на вычислении среднего арифметического.

f2 - фильтр, основанный на вычислении среднего геометрического.

f3 - медианный фильтр.

f4 - фильтр, основанный на вычислении среднего гармонического.

f5 - фильтр срединной точки.

$ОСШ_L$ -отношение сигнал/шум.

Значение Lf для f1 рассчитывается по формуле.

Lf =

Таблица 5 – Экспериментальные показания.

Тип фильтра	**L, см (линейка)**	**Lf, см (датчик)**	**Δy_L, мм**	**δ_L**	**D_L, мм2**	**σ_L, мм**	**$ОСШ_L$**
f1	**2**	3,09	1,09	0,3527508091	0,00448	0,06698068811	46,132
	4	4,33	0,33	0,07621247113	0,01388	0,1178410587	36,744
	8	8,36	0,36	0,04306220096	0,02157	0,1468928457	56,912
	16	16,53	0,53	0,03206291591	0,03119	0,1766104452	93,595
	32	33,24	1,24	0,03730445247	0,06665	0,2581768243	128,748
	64	65,38	1,38	0,02110737229	0,08631	0,2937856914	222,543
	128	129,24	1,24	0,00959455277	0,09063	0,3010584927	429,285
f2	**2**	3,03	1,03	0,3399339934	0,00491	0,07009771728	43,223
	4	4,26	0,26	0,06103286385	0,01452	0,1205297939	35,3433
	8	8,29	0,29	0,03498190591	0,02435	0,1560602805	53,12

	16	16,84	0,84	0,04988123515	0,033541	0,1831427722	91,95
	32	33,83	1,83	0,05409399941	0,07050	0,2655198092	127,41
	64	65,6	1,6	0,0243902439	0,09058280636	0,3009697765	217,962
	128	129,21	1,21	0,00936460026	0,098434	0,3137419774	411,835
f3	**2**	3,12	1,12	0,358974359	0,004059	0,06371134272	48,97
	4	4,34	0,34	0,07834101382	0,012391	0,1113150676	38,988
	8	8,33	0,33	0,03961584634	0,020082	0,1417111049	58,781
	16	16,57	0,57	0,0343995172	0,034395	0,1854611274	89,344
	32	33,62	1,62	0,04818560381	0,0611	0,2471854902	136,011
	64	65,29	1,29	0,01975800276	0,08823	0,2970400322	219,802
	128	132,02	4,02	0,03044993183	0,11413	0,3378408981	390,775
f4	**2**	3,09	1,09	0,3527508091	0,00405	0,06371134272	48,499
	4	4,33	0,33	0,07621247113	0,01452	0,1205297939	35,924
	8	8,36	0,36	0,04306220096	0,02328	0,1525995337	54,783
	16	16,52	0,52	0,03147699758	0,02969	0,1723246311	95,865
	32	33,24	1,24	0,03730445247	0,06815	0,2610569736	127,328
	64	65,38	1,38	0,02110737229	0,08972	0,2995467436	218,263
	128	129,03	1,03	0,00798263969	0,10377	0,3221412667	400,538
f5	**2**	3	1	0,3333333333	0,00491	0,07009771728	42,797
	4	4,34	0,34	0,07834101382	0,01324	0,1150895262	37,709
	8	8,45	0,45	0,05325443787	0,02115	0,145431189	58,103
	16	16,26	0,26	0,0159901599	0,02755	0,1660102152	97,945
	32	33,21	1,21	0,03643480879	0,06686	0,2585902382	128,427
	64	65,22	1,22	0,01870591843	0,08823	0,2970400322	219,566
	128	131,47	3,47	0,02639385411	0,09603	0,3098880331	424,25

Выводы: Оптимальным по значению отношения сигнал/шум является медианный фильтр.

Список использованных источников.

1. https://hcomp.ru/downloads/arduino/UNOr3/arduino_uno_r3_RUS.pdf
2. https://robotchip.ru/download/datasheet/HC-SR04-Datasheet.pdf
3. http://easyelectronics.ru/img/starters/linear/Range_SHARPGP2D120.pdf