DOI: 10.35598/mcfpga.2020.012

Application of Software Signal Filtering in an Ultrasonic Rangefinder

Artem Khromenko
ORCID 0000-0002-5120-1094
Department of Microelectronics, Electronic Devices and Appliances
Kharkiv National University of Radio Electronics
Kharkiv, Ukraine
artem.khromenko@nure.ua

Liliia Saikivska
ORCID 0000-0002-4139-7732
Department of Microprocessor Technologies and Systems
Kharkiv National University of Radio Electronics
Kharkiv, Ukraine
liliia.saikivska@nure.ua

***Abstract*—This article describes an example of creating an ultrasonic rangefinder on the ARDUINO platform using inexpensive modules and components. The main feature of the device is the use of a software method of filtering values, which significantly increases its accuracy, and makes the device comparable in accuracy with industrial analogues, with a relatively low cost.**

***Keywords*—board, ultrasonic sensor, Arduino, filter**

I. Objective

In everyday life, or in view of the characteristics of the sphere of activity, we often encounter the need to measure the distance to an object or its length, whether it be a wall, ceiling, room height, etc. It is not always convenient to use a classic measuring tool in the form of a ruler, tape measure or carpentry «meter», taking into account the measurement features, scope or other factors.

The aim of this work is to develop a portable pocket rangefinder of sufficient accuracy on an inexpensive element base for use in construction, carpentry or domestic purposes as a universal device for measuring short distances. The device must meet the following requirements:

- compactness;
- ergonomics (low power consumption);
- range of the measured distance to 4-5m;
- error in the measurement of 1 - 2 mm;
- digital display of information.

II. Implementation

To implement this project, the ARDUINO Nano platform on the Atmega 328 microcontroller was used as the main module. The HC SR04 ultrasonic sensor was used as the range finder sensor. Information is displayed on the four-digit seven-segment display of the 74NS595. Conventional AA batteries were used as a power source. In the future, it is planned to equip the device with a lithium-ion battery with the possibility of recharging it. A modular diagram of the device is shown below.

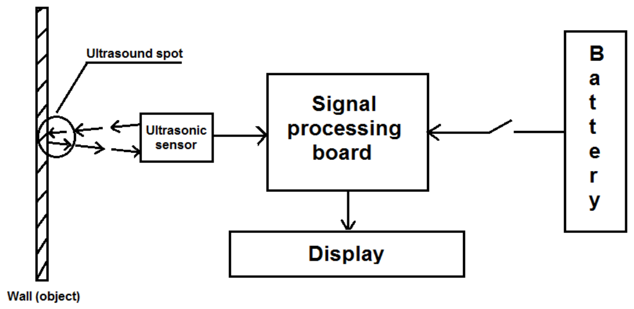

Fig. 1. A modular diagram of the range finder.

III. Principle Of Operation

The operation of the device can be described as follows: the Arduino platform generates a series of consecutive pulses that are transmitted to the emitter of the ultrasonic sensor, the sensor emits these pulses, captures their reflections and generates signals at its output, which are fed to the processor digital inputs. The processor, according to the written algorithm, calculates the time between the signal sent to the sensor and its response to the reflected signal. The received data is generated into a data array and processed using software filtering, in order to avoid accidental outliers of false results, and to increase the measurement accuracy. Software filtering methods will be described below. From a certain frequency, the processed results, digitized and converted to metric format, are displayed on a digital display. The display refresh rate can be varied, the speed of updating the data on the display when the distance from the sensor to the object changes depends on it.

IV. Application Software Filtering

Using the ARDUINO platform, we can write program code using several methods to filter output values. This, in turn, can significantly improve the measurement results, namely, to minimize the noise generated by the ultrasonic sensor due to its design features.

A. Median filter

The appropriateness of using software filtering of signals received from an ultrasonic sensor is due to the fact that as a result of analog-to-digital conversion, a number of values can be obtained in which one or more will significantly differ from neighboring ones, for example: “58, 61, 59, 231 , 60. " A value of 231 is supposedly anomalous and should be ruled out. If you try to use the classic linear filter, you will see that a value of 321 will have a significant effect on the result. The best solution in this case is to use a median filter. For even “n” values, the median is usually defined as the arithmetic average of two average samples of the ordered sequence, that is, in contrast to the arithmetic average, it finds the “average among the average”, and not among all the values, thereby filtering out sharp outliers. The graph below shows the result of the filter against emissions:

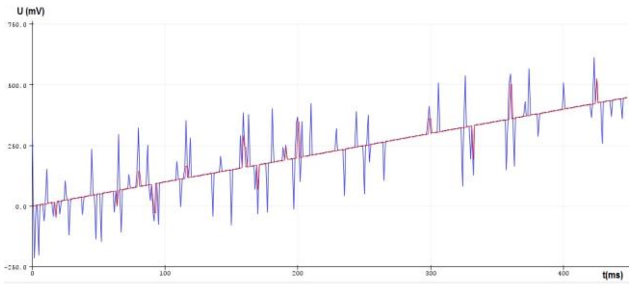

Fig. 2. Is a graph of the median filter, where the blue graph is the real value, the red is filtered.

It should be noted that the higher the dimension of the median filter, the better it cuts off frequent outliers, but at the same time the processor load increases as the number of calculations increases, which entails an increase in the time for updating data and displaying it. In this case, the median filter from the last 3 measurements is used, an example of program code using the median filter is shown below:

```
// Median filter of 3 values
float middle_of_3(float a, float b, float c) {
  if ((a <= b) && (a <= c)) {
    middle = (b <= c) ? b : c;
  }
  else {
    if ((b <= a) && (b <= c)) {
      middle = (a <= c) ? a : c;
    }
    else {
      middle = (a <= b) ? a : b;
    }
  }
  return middle;
}
```

B. *Filter running average*

To achieve maximum accuracy, the running average filter algorithm was also used - this is a method of smoothing time series in order to exclude the influence of a random component. The method consists in replacing the actual values of the members of the series with the arithmetic mean of the values of the several members closest to it. A set of averaged values forms the so-called sliding window. A member whose value is replaced by the window average takes the middle position in the window.

The filter has a setting parameter as a "filtration coefficient", with the help of which a range of filtered values - outliers is adjusted. This coefficient ranges from 0 to 1.

The following is an example of applying a running average filter to combat random outbursts of a uniformly growing signal.

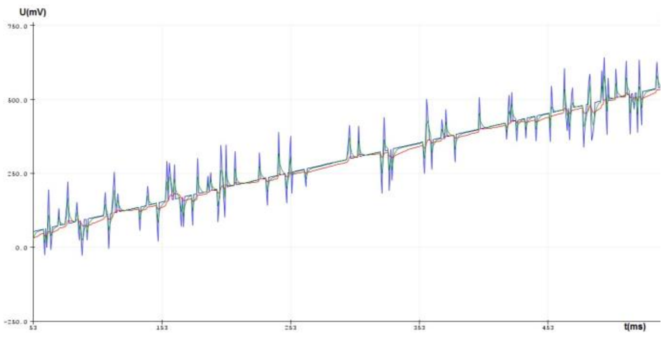

Fig. 3. Graph of the moving average filter, where the blue graph represents the real value, red filtered with a coefficient of 0.1, green with a coefficient of 0.5.

An example of program code using a moving average filter is shown below:

```
if (delta > 1) k = 0.7;
else k = 0.1;

dist_filtered = dist * k + dist_filtered * (1 - k);
```

V. CONCLUSION

The use of two types of filtering in the software and hardware code allowed to significantly increase the accuracy of the device to an error of only 1 mm, and its implementation on a common component base makes it affordable and cheap to manufacture.

This device is not an accurate measuring tool, but it can be used to perform small, often performed measurements in the home or construction.

REFERENCES

[1] Goldenberg, L.M. et al. Digital signal processing. Directory. Radio and communication / L.M. Goldenberg. - Moscow, 1985.- 312 p.

[2] Boxell, J. Learning ARDUINO. 65 do-it-yourself projects / J. Boxel - St Petersburg, 2017 .- 400 p.