Article

Design of an open-source software system for water temperature monitoring

João E. M. Perea Martins
Computer Science Department, School of Sciences (FC), São Paulo State University (UNESP), Bauru-SP, Brazil
E-mail: joao.perea@unesp.br

Received 1 December 2024; *Accepted 5 January 2025*; *Published online 25 January 2025*; *Published 1 September 2025*

Abstract
This article presents the software and hardware components of an open-source system designed for water temperature monitoring. The system can handle measurements from up to eight sensors at user-defined time intervals. It was designed with the waterproof temperature sensor model DS18B20, an Arduino board and a computer. The Arduino-based program reads the sensors data and transmits the temperature values to the computer in real-time via a serial communication. The computer software has an intuitive user interface to ensure an easy operation, and it displays the temperatures numerically, generates graphs, and allows the user to save the data to a disk at any time. Besides the Arduino and the sensors, the hardware setup requires only a single additional resistor making assembly straightforward. This article details the Arduino and the computer source code programs, and the satisfactory system results have motivated the work presentation.

Keywords water temperature; data acquisition; temperature sensors; DS18B20; Arduino; measurements.

Computational Ecology and Software
ISSN 2220-721X
URL: http://www.iaees.org/publications/journals/ces/online-version.asp
RSS: http://www.iaees.org/publications/journals/ces/rss.xml
E-mail: ces@iaees.org
Editor-in-Chief: WenJun Zhang
Publisher: International Academy of Ecology and Environmental Sciences

1 Introduction

This work focuses on an automatic system for water temperature measurement, which is an important parameter associated with several subjects, such as global climate, agriculture, ecosystem, aquaculture, hydrology, biodiversity, melting ice, water quality, industrial activities, human health, wastewater treatment, etc.

Among the various associated subjects, the climate change is a special topic that has become increasingly important because it presents significant consequences and affects the global water temperatures. 71% of the Earth surface is covered by water from oceans, lakes, and rivers, which act as a heat sink, absorbing and storing heat from the sun. This process efficiency also depends on the water temperature and can influence the environmentwith consequences such as the evaporation rateincreasing, causing water shortages and increasing the frequency and intensity of droughts.

Venegas et al. (2023) and Xu (2021) showed the oceans water have warmed at rates of up to 0.13ºC per decade, according to the region, but water rivers are also undergoing warming, which a more pronounced growth. Liu et al. (2020) verified the water temperature of some rivers in the large tropical region, between latitudes 30°S and 30°N, has increased at 0.5°C per decade, but the problem occurs even outside this region, such as in rivers of USA, where warming reached up to 0.08ºC per year (Kaushal, 2010; Johnson et al., 2024). The water temperature increasing affects all the aquatic life, influencing the fish body temperature that varies according to the water temperature and influences changes in theirphysiological parameters, such as biochemical pattern, growth, size, and age at maturity (Islam, 2020). Poloczanska et al. (2013) andVehanen (2023) presented analysis showing that as ocean temperatures rise, some aquatic life species are moving towards cooler regions, causing changes in marine systems and even indicting future species extinction problems in regions that are becoming warmer. According to the Food and Agricultural Organization, fish can represent more than 60% of the per capita intake of animal protein in some countries (FAO 2020), and therefore, fish migration can negatively affect these regions.

Water is also important in cooling processes, and in the adiabatic cooling process, which is based on water vaporization into the hot air, the energy consumption depends directly on the water temperature, which requires its precise measurement (Mihai and Rusu, 2022). This process saves a large amount of water, and the ambient air temperature can be controlled by adjusting the air flow and is used in industrial plants, data centers, HVAC systems, agricultural greenhouses, etc.

Water temperature historical data isalso avery important variablein water temperature predication processes, which are based on mathematical models. They allow the water temperature predication and consequently permit early decision-making in several areas. However, the development, validationand use of mathematical models of predication requires true input data of some parameters and, therefore, the existence of historical databases of water temperature becomes animportant factor for predictions (Loubna et al., 2007, Zhu and Piotrowski, 2020; Feigl et al., 2021, Padrón et al., 2024)).

In agriculture, plants absorb water through their roots, and water temperature can affect the rate of nutrient uptake, and therefore, the right temperature causing faster uptake and improved plant metabolism. The water temperature can also affect the solubility and availability of nutrients in the soil, the rate at which seeds germinate and roots develop, and the efficiency of photosynthesis by plants. In addition, water temperature also affects evapotranspiration, where higher water temperatures increase the rate of evaporation from soil and plant surfaces, leading to greater water demand (Wen et al., 2023; Barros et al., 2022; Sghaier et al., 2022)

The influence of water temperature on the different fields allows the discussion of numerous issues associated with scientific, social and economic aspects. However, it isn’t the main purpose of this article, and the contextualization above becomes a way to prove the water temperature monitoring relevance, which is the true focus of this work. Therefore, the next sections show the proposed system structure and detail the measurement hardware interface, the waterproof sensor analysis, and the data acquisition software operation and code.

2 The System Structure

A data acquisition system is a process that includes the gathering, analyzes, and recording information concerned to a specific phenomenon (Song et al., 2024), and therefore, in addition to the hardware components, its software should include stages as data acquisition, data analysis, and data storage (Li, 2021). Fig. 1 shows the data acquisition system structure proposed in this work, which is called *Water Temperature Datalogger*, or simply *WTLog*, and includes hardware and software components, where the hardware components are:

1. The waterproof temperature sensors model DS18B20, whose quantity can vary from 1 to 8 according to the user's needs.
2. An Arduino board. This work used the Arduino board called "nano", but it can be replaced by several other boards without software or hardware adjustments.
3. A computer.

The software components used in this work are summarized as:

1. Arduino software, *that controls the sensors*, get their measurement values and sends them to the computer.
2. A computer software that receives the data from the Arduino board, analyses them, shows their values in numerical and graphical forms, and allows their storage on a disk.

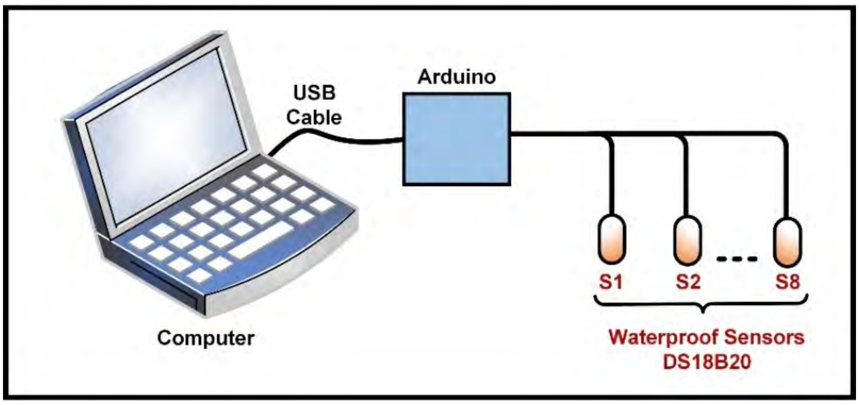

Fig. 1 The data acquisition system structure.

3 The Measurement Interface

Fig. 2 shows the temperature measurement interface includesan Arduino board and up to eight DS18B20 temperature sensors, which requires only one additional 4.7 kΩ resistor for pull-up connection. In fact, there are several works that uses the DS18B20 sensor in operation with embedded processing systems for watertemperature measurement in fields as water quality, agriculture, food industry, microclimate monitoring, etc (Simanjuntak et al., 2022; Al Mamun et al., 2024; Bogdan et al., 2023; Rebaudo et al., 2023), which indicates the sensor quality. However, the present work presents a differential, which is the inclusion of software techniques to allow the user to choose the sensors number from 1 up to 8, without hardware or software adjusts. This sensor has only three wires for connections, where two are connected to the power source (5 V and GND) that are derived directly from the Arduino board, and the third is for data flow between the Arduino and the sensor, and this work connects it to the Arduino pin2.

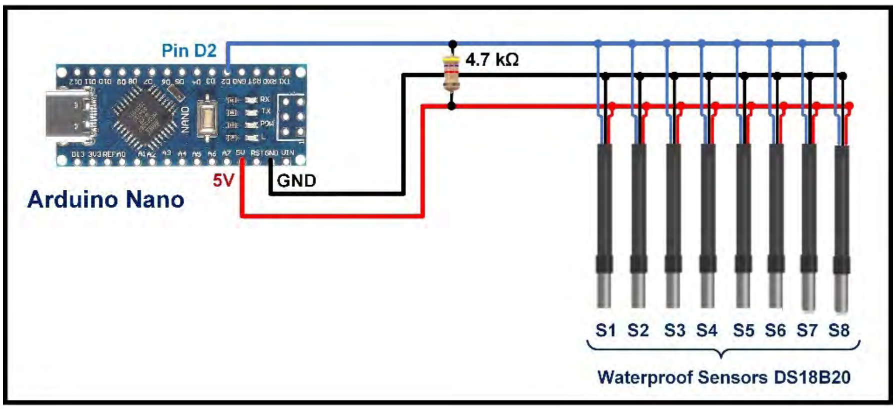

Fig. 2 The measurement hardware.

The DS18B20 sensor has two commercial versions, the first is a simple electronic sensor, and the second covers it with a metallic probe to operate as a waterproof device, as shown in figure 2.Among the DS18B20 sensor characteristics, is interesting to highlight:

1. Measures temperatures in the range from -55°C to +125°C, with accuracy at ±0.5°C from -10°C to +85°C.
2. Programmable Resolution by software, up to 0.0625°C
3. Low current consumption, typically at 1mA in operation mode, and 750nA when in standby mode.

There are different types of sensors for invasive temperature measurement, where the sensor is in direct contact with the medium in analysis, which varies since traditional Liquid-in-glass thermometers to sensors that allow their connection to electronic circuits and basically can be classified as:

1. Resistive sensors, where the sensor electrical resistance varies according to the temperature and includes types as RTD (Resistance Temperature Detector) and Thermistors classified as NTC and PTC.
2. Thermocouples: They are a type of sensors designed with two metal wires made of different materials and joined at their ends to form a junction where a small output voltage is naturally generated according to the temperature, which occurs physically due to the called Seebeck effect.
3. Integrated Circuit (IC) Temperature Sensors that produce an output voltage proportional to temperature and are designed with several built-in auxiliary electronic components, which allows signal processing with just a minimum of external electronic components and usually allows a more accurate measurement.

Resistive sensors and thermocouples can measure temperatures in higher ranges up to hundreds or even a few thousand degrees, which makes them preferable in certain scientific or industrial applications, but their connection to electronic circuits, which may be theoretically simple, in practice, may require special care to avoid noise and signal conditioning problems. IC sensors usually measure temperatures up to a few hundred degrees, which is satisfactory for environmental applications, and usually allow an easier interfacing with existing electronic circuits. Besides, typically they offer high reliability and long-term stability, with less need for calibration or maintenance over time, ratability, small physical dimensions, and low power consumption, and therefore become an ideal option for environmental measurements and were chosen for this work design.

The Arduino board requires software to manage the DS18B20 sensors, and figure3 shows thisprogram flowchart, which configures the Arduino system in the first block, wait to receive a byte from the computer through serial communication in the second block, and next, in the third block, gets the temperature of all sensors and send to a computer. Note that, the Arduino waits for a byte to arrive from the computer, and therefore, the computer is the one that determines when the measurements will be done.

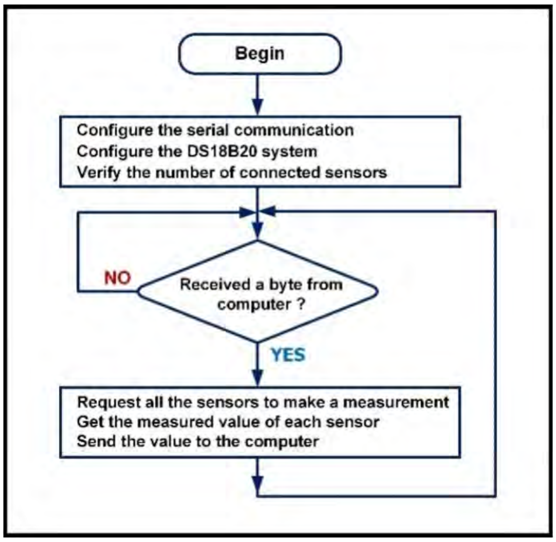

Fig. 3 The Arduino program algorithm.

Fig. 4 shows the program that runs on the Arduino board, which follows exactly the sequence of the flowchart shown in Figure 3. The program is simple and is fully documented to facilitate its understanding.The number of sensors physically connected to the system is checked in the “void setup” and is stored in the variable “deviceCount”. In the “void loop” the command “if” verify the serial byte reception from the computer and then start all the measurements.

```
// Arduino Program to controle up to 8 temperature sensors model DS18B20

#include <OneWire.h>                // Include library
#include <DallasTemperature.h>      // Include library
#define ONE_WIRE_BUS 2              // Define pin 2 for sensors connection

float tempC;
int s, deviceCount, inByte;

OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature Sensors(&oneWire);

void setup(void)
{ Serial.begin (9600);                        // Serial port BPS
  Sensors.begin ();                           // DS18B20 sensors initialization
  Sensors.setResolution (11);                 // Sensors resolution at 11 bits (0.125ºC)
  deviceCount = Sensors.getDeviceCount (); // Verify the number of sensors
}

void loop(void)
{ if (Serial.available() > 0) {              // Verify the arrival of a byte via serial
    Sensors.requestTemperatures();           // Determines that each sensor measures temperature
    inByte = Serial.read();                  // Flush the serial buffer
    for (s = 0; s <deviceCount; s++) {
      tempC = Sensors.getTempCByIndex(s);   // request temperature measured by the sensor 's'
      Serial.print(s + 1);                   // Send sensor number to computer
      Serial.print(" ");
      Serial.print(tempC);                   // Send measured temperature to computer
      Serial.print(" ");
     }
    Serial.println("");
   }
}
```

Fig. 4 The Arduino program.

This work verified the waterproof DS18B20 sensor performance immersing two sensors in water at a constant temperature of 29.80ºC and performed 1500 measurements with each sensor simultaneously, with an interval of one second between each measurement. Fig. 5 shows the measurement graph of both sensors, named "SA" and "SB".

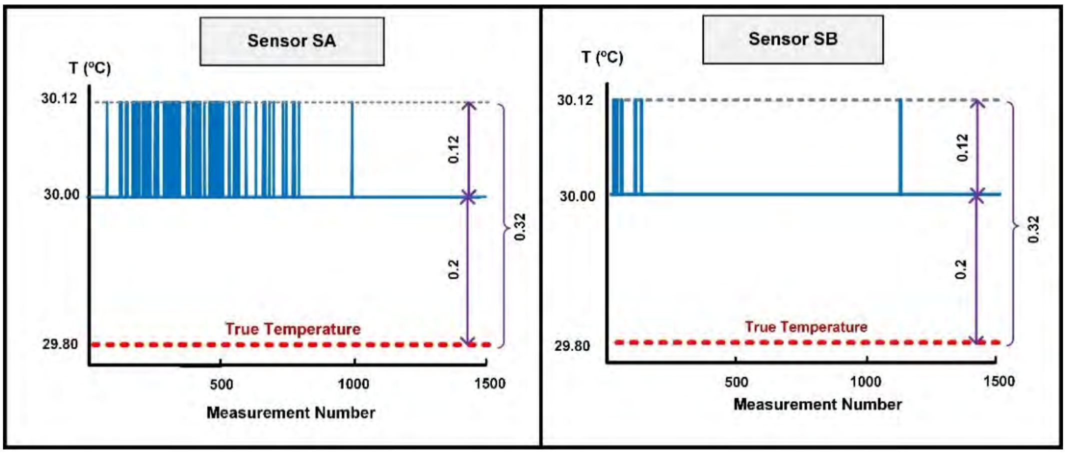

Fig. 5 Measurements with two DS18B20 sensors, refereed as "SA" and "SB".

The first analyzed parameter was the sensor accuracy, which the International Vocabulary of Metrology (VIM), published by International Bureau of Weights and Measures in association with other notorious institutions, defines as "closeness of agreement between a measured quantity value and a true quantity value of a measurand" (JGCM, 2012). According to the sensor DS18B20 datasheet, its accuracy is ±0.5ºC, and it means the maximum difference between the true temperature and the measured temperature that can occur is ± 0.5ºC. Note, it is the maximum difference value allowed, but, for practical approaches, each sensor piece can have an accuracy equal or smaller than it, and, besides, the accuracy of each sensor piece can vary from others.

In this experiment, the true water temperature was 29.8ºC, and therefore any measured value should be in the range 29.8ºC ± 0.5ºC, or from 29.3ºC up to 30.3ºC. Fig. 5 shows the measurements of both sensors are in thisrange, and presented a good accuracy level, which, coincidentally, had a maximum value of 0.32ºC for both sensors.

The sensor precision was the second analyzed parameter, which the VIM defines as "closeness of agreement between indications or measured quantity values obtained by replicate measurements on the same or similar objects under specified conditions". In other words, it indicates the measurement result repeatability for several measurements under the same conditions, which can be analyzed with the statistical analysis of a data set. Table 1 shows this analysis for the Fig. 5 data set, and, in the worst case, the coefficient of variation (cf) was only 0.09%, which indicates the variation around a mean value is extremely small, and therefore, it proves the DS18B20 temperature sensor hasan excellent precision. Table 1 details the arithmetic mean value (mean) for 1500 measurements with each sensor, and also includes the standard deviation (std), the coefficient of variation (cf), the smallest data set value (min), the biggest data set value (max), and the number of occurrences of the min and max values that are referred as Nmin and Nmax, respectively.

Table 1 Analysis of the measurements of two sensors.

Sensor	mean (ºc)	std (ºc)	cf (%)	min (ºc)	max (ºc)	Nmin	Nmax
SA	30.01	0.027	0.09	30	30.12	1421	79
SB	30.0003	0.0077	0.026	29.87	30.12	1495	5

Assuming that resolution is the difference and precision is its repetition over time, then, in cases where the sensor precision is good, with a high degree of repeatability, the measured value can be easily later corrected mathematically, by adding or subtracting a correction constant value. For example, 97.7% of the sensor SA measurements were 30ºC, which increases to 99.6% for sensor SB. The difference between the majority measurement values (30ºC) and the true temperature value (29.8ºC) is 0.2ºC, and therefore, if the measured values are later subtracted by 0.2, then at least 97.7% of the measurement values would have a value equal to the true value, which this work considers a very good result. In the present case, both sensors required the same numerical correction value at 0.2ºC, but it be analyzed individually for each sensor piece because it may vary slightly between them.

Another interesting fact to be analyzed is the sensor ability to detect abrupt changes in temperature, which occur in some processes, such as in pasteurization,where the food is heated to tens or even hundreds of degrees Celsius, and then rapidly cooled to temperatures that may be just a few degrees (Indumathy et al., 2022; Lan et al., 2024). In practice, sensors have a parameter called *response time* (RT) that is an undesired delay in the detection of the phenomenon intensity variation when it occurs abruptly as a step-function. Scientifically, it is defined as the time the sensor takes to reach 63.2% of the measured phenomenon value when it changes abruptly, which is at about 10 seconds a mercury-in-glass thermometer and decreases to 7 seconds for the waterproof sensor DS18B20 (Perea Martins, 2022).

4 The Data Acquisition Software

This work used the Delphi programming language to develop the WTLog main software that runs on a computer, and this language was chosen because it is very efficient and easy to use in creating human-machine interfaces, allowing the creation of screens where the programmer can easily position, align, and size the components, adjusting quickly the interface layout according to their needs and accelerating the development process.This language may not be the most popular language today, but since its release in 1995, the Delphi language has evolved continuously (Cornelius, 2021), and in 2024, the most recent Delphi version 12.1 was released.Besides, it remains a leader in Rapid Application Development (RAD) for desktop applications, which is a software design methodology that focuses on developing applications rapidly through frequent iterations and continuous feedback, and uses an iterative and incremental approach, dividing the development process into specific stages.

Another important justification for using Delphi is its easy code understanding. It is based on the Object Pascal programming language that is an extension of the Pascal language and was created to be a teaching language, with a clear and easy-to-learn syntax (Zhang, 2023; Zhang and Qi, 2024). Thus, it is expected that a Delphi code can be easily understood by programmers of other languages, such as Python or C. Besides, the Delphi language also has an IDE (Integrated Development Environment) that includes excellent debugging tools, which allows the programmer to easily identify problems and errors in the code development. Delphi is a commercial development environment product, but there is a software development system called Lazarus that is similar to Delphi and can be used on several platforms for free.

Fig. 6 shows the screen of the WTLogprogram designed in this work and that runs on a computer. This program structure includes four main parts that are:

1. A main menu with seven options in the upper left part to configure the system.
2. A temperature data box that showsnumerically the measured temperatures.
3. A temperature chart box to plot the temperatures in real time.
4. Three small display boxes on the left side that shows the current configuration.

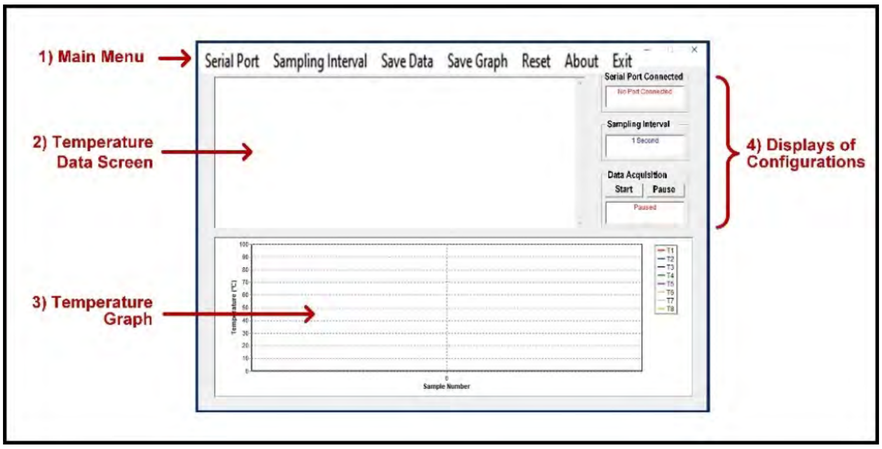

Fig. 6 The data acquisition software screen.

The main menu has seven options, which are:

1. *Serial Port*. It allows access to two different procedures associated with the serial communication port, which are:
   1.1 *Connect the Serial Port*. When the user clicks on this subitem, a box for communication port configuration appears. In practice, Fig. 7 shows the user must only set the first item to define the serial port, referred as COM, that the Windows has reserved for connecting the Arduino board.

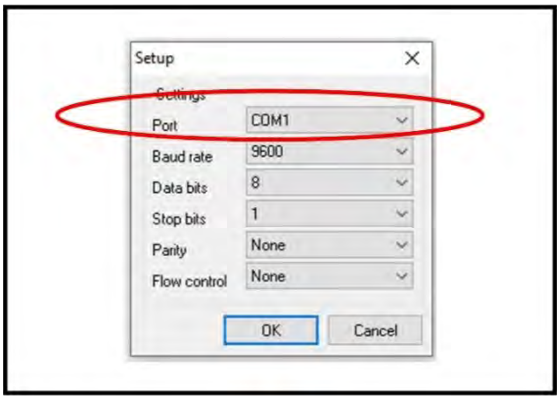

Fig. 7 box for communication port configuration.

1.2 *Close the Opened Serial Port*. This procedure closes the connected (opened) serial port between the computer and the Arduino, which is useful in cases where this port needs to be used by other software.

2. *Sampling Interval:* This procedureallows the user to define the time interval between two consecutive temperature measurements.
3. *Save Data:* This option saves all data shown in the temperature data box to a disk file. Figure 8 shows the message shown after the saving operation, where the file name is composed according to the current date and time, and therefore, there will never be two files with the same name, avoiding the overlapping of data files and the consequent loss of older data. Remember, the user can rename any file later. At any time, the user can save the data to a text file (TXT) without interrupting the data acquisition process.

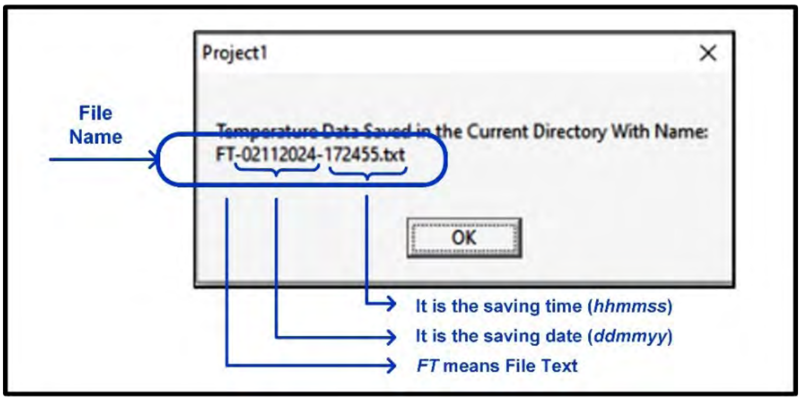

Fig. 8 Name of the data set text file.

4. *Save Graph:* This option saves the graph from the temperature chart box to a figure file, type BMP. The graph file name is also composed according to the current date and time, such as “GT-02112024-171442”, where GT means Graph of Temperature and the next numerical information represents the saving date and time. This operation doesn’t interrupt the data acquisition process and also avoids the overlapping of files.
5. *Reset*: This procedure resets the entire data acquisition system, erasing all data and configurations to allow the running of a new data acquisition process.
6. *About*: It shows the software system designer’s name and email to allow comments and questions from users.
7. *Exit*: This function ends the processing and closes the program. It has the same effect as clicking on the 'X' symbol in the top right corner.

Fig. 9 shows the information layout to show the temperatures on the data screen in real time, where each line represents a set of measurements done ata specific moment.

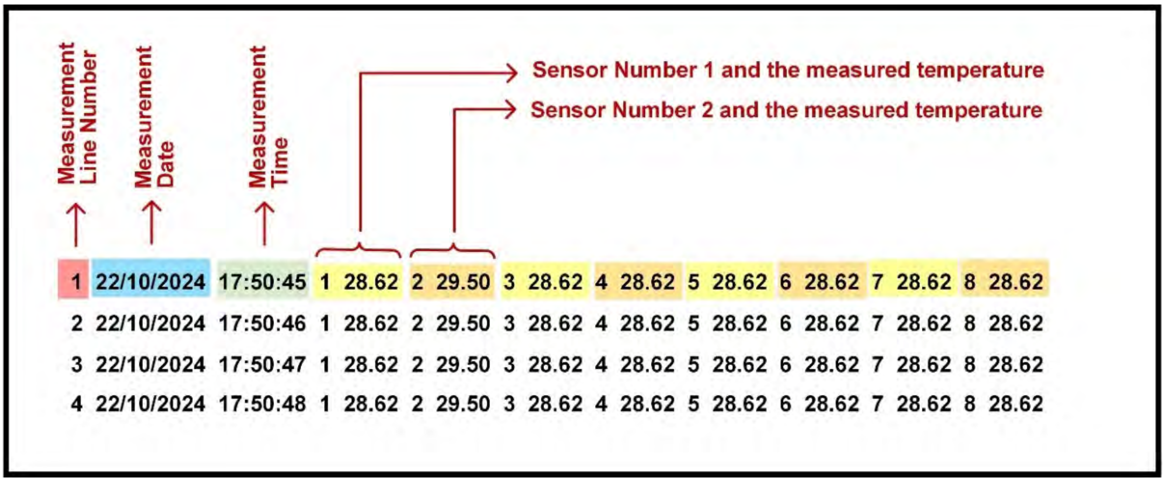

Fig. 9 Temperature data layout on the software screen.

The first data row in Fig. 9 was colored only to aid its explanation. The first number, highlighted in red, is the number of the measurement set, or the measurement line, which increases sequentially. The second information, highlighted in blue, is the line measurements date, and the third information, highlighted in green, is the measurements time. Therefore, it shows the measurement moment. Note, in this figure, the measurement time difference between two sequential lines is one second, because it was the sampling interval defined by the user. Next, highlighted in yellow or orange, is the measured temperature by each sensor, where first appearsthe sensor number and next its measured temperature. In the figure 20, there are 8 sensors, but the system is intelligent and automatically verifies the exact number of sensors physically connected to the Arduino board, and shows only thedata associated with the existing sensors.

The box “Temperature graph” shows the graphs of temperatures measured withall sensors, which is updated in real time. Fig. 10 shows an example when the system had two sensors.

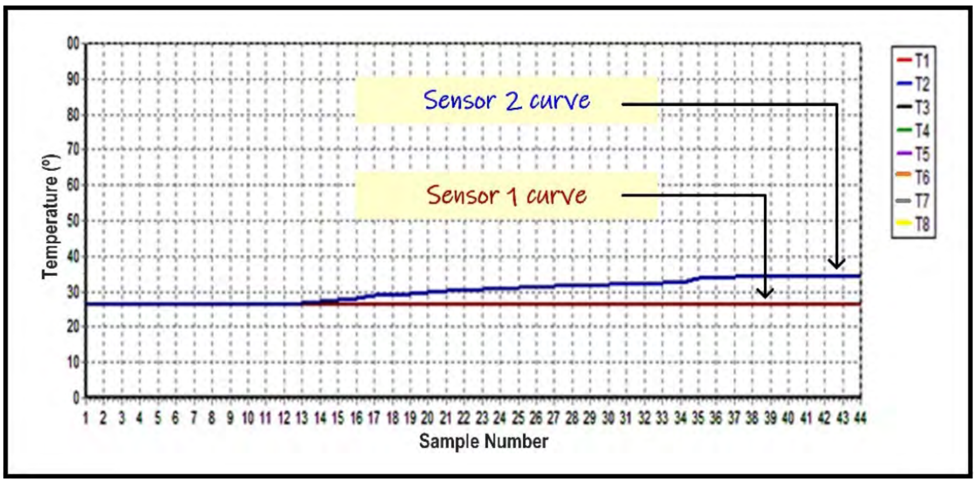

Fig. 10 Temperatures measured with two sensors.

On the right side, vertically in the upper half of the screen, there are three small displays referred as “Displays of configurations“, which are:

1. *Serial Port Connected*: It shows the serial port name that connects the Arduino board and was defined through the Serial Port procedureof the main menu.
2. *Sampling Interval*: It show the time interval between two consecutive temperature measurements, whose value is set by user through the Sampling Interval procedureof the main menu.
3. *Data Acquisition*: It has two buttons named “Start” and “Pause”, that, respectively, start and pause the measurements. After stopping, the measurements will continue again when the Start button will be clicked.

In short, the software operation is very simple. After connecting the sensor hardware, the user only needs to:

1. Set the communication serial associated with the Arduino.
2. Set the sampling interval.
3. Click the “Start”button.

Fig. 11 shows the form designer used in this software design, where the **form**is a visual box that accommodates the user interface (UI) of an application and shows exactly what the user will see later, when the program is run. Therefore, the form represents the final software screen with the UI elements, such as buttons, labels, text boxes, and menus. These elements compose visual components that are associated with specific software procedures designed by the programmer, but the form can also accommodate non-visual components that run without a visual interaction with the final user.

Fig. 11 shows all the software components and software procedure name associated with each component. For example, the button component “Start” (number 10) in the form is associated with the software procedure referred as "procedure TForm1.Button1Click". The source code of this program is shown below, in the appendix after the references, and it requires only two components that may not be native to Delphi, which are Tcomport and Tchart, which, respectively performs the serial data communication and the plotting graphs. The software is modular and was coded as simply as possible to allow its change or improvements, if necessary. Fig. 11 is the Delphi form that follows the exactly the same layout showed in Fig. 6 for the final executable program.

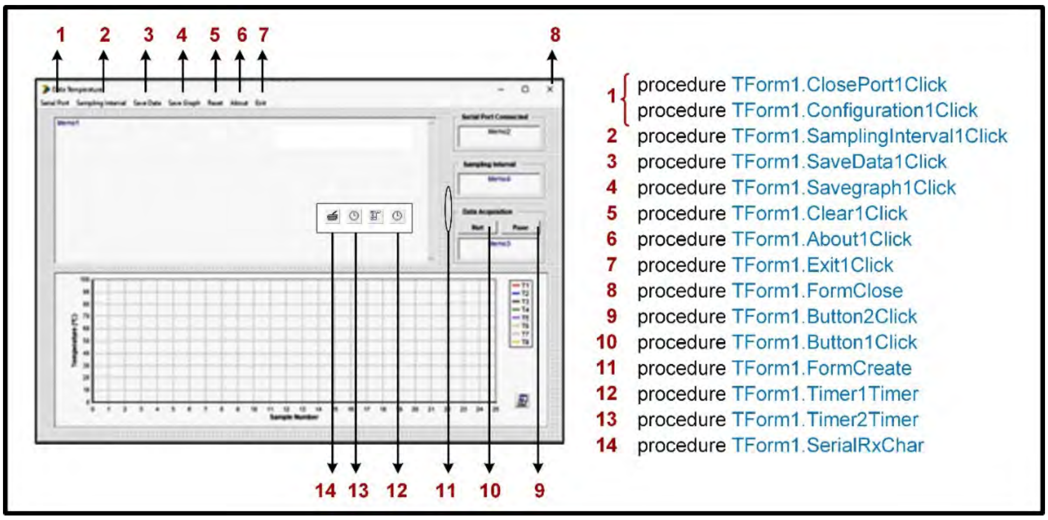

Fig. 11 The software form and procedures.

5 Conclusions

This work showed the design of a complete open-source data acquisition system for monitoring water temperature, called WTLog, including all the hardware and software components. The sensor used, model DS 18B20, proved to be very efficient in the tests, presenting excellent levels of resolution and precision whose statistical analysis showed a very small coefficient of variation, less than 0.1%. The software achieved the initial proposal of having an intuitive human-machine interface and also allows data storage at any time, without interrupting the measurement process. In addition, the user can define the number of sensors, ranging from 1 up to 8, without having to make any software changes or configurations.

Appendix

This appendix shows the source code of the data acquisition progrqm, written in Delphi programming language.

```
//*****************************************************************
// TEMPERATURE DATA LOGGER PROGREAM SOURCE CODE
//*****************************************************************
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, CPort, TeEngine, Series, ExtCtrls, TeeProcs, Chart,
  Menus, ComCtrls, ActnList;
type
TForm1 = class(TForm)
    Memo1: TMemo;    Memo2: TMemo;    Memo3: TMemo; Memo4: TMemo;
Serial: TComPort;
    Chart1: TChart;
    Series1: TLineSeries;
    Timer1: TTimer;
    GroupBox1: TGroupBox; GroupBox2: TGroupBox; GroupBox3: TGroupBox;
    MainMenu1: TMainMenu;
    About1: TMenuItem;
    SerialPort1: TMenuItem;
    Configuration1: TMenuItem;
    ClosePort1: TMenuItem;
    SamplingInterval1: TMenuItem;
    SaveData1: TMenuItem;
    SaveGraph1: TMenuItem;
    Clear1: TMenuItem;
    Series2: TLineSeries; Series3: TLineSeries; Series4: TLineSeries;
    Series5: TLineSeries; Series6: TLineSeries; Series7: TLineSeries;
    Series8: TLineSeries;
    Exit1: TMenuItem;
    Button1: TButton; Button2: TButton;
```

```
        Timer2: TTimer;
        ActionList1: TActionList;

procedureSerialRxChar(Sender: TObject; Count: Integer);
procedureFormCreate(Sender: TObject);
procedure Timer1Timer(Sender: TObject);
procedure SaveData1Click(Sender: TObject);
procedure Savegraph1Click(Sender: TObject);
procedure About1Click(Sender: TObject);
procedure Configuration1Click(Sender: TObject);
procedure ClosePort1Click(Sender: TObject);
procedure SamplingInterval1Click(Sender: TObject);
procedure Clear1Click(Sender: TObject);
procedureFormClose(Sender: TObject; var Action: TCloseAction);
procedure Exit1Click(Sender: TObject);
procedure Button1Click(Sender: TObject);
procedure Button2Click(Sender: TObject);
procedure Timer2Timer(Sender: TObject);
    private
{ Private declarations }
    public
{ Public declarations }
end;

var
    Form1: TForm1;
varamostra: integer;
timesampling: word;
            start, SerialFlag: boolean;
lf, stg, sg, Sdata, myFile: string;
vDate :TDateTime; vHour: TDateTime;
implementation
{$R *.dfm}

//*********************************************************
```

procedure TForm1.SerialRxChar(Sender: TObject; Count: Integer);

```
var s, SX, dados:string;
ID, p, ind, t: integer;
begin
t:=Serial.InputCount;
if t>0 thenbegin
Serial.ReadStr(dados,t);
sg:=sg+dados;
p:=pos(#10,sg);
```

```
if p>0 thenbegin
dados:=copy(sg,1,p);
        delete(sg,1,p);
        delete(dados,p-1,2);      // Delete #13 and #10
sx:=copy(dados,1,1);
ID:=strtoint(sx);
s:='';
amostra:=amostra+1;
vDate:=Now;    vHour := Time;
memo1.Lines.Add (inttostr(amostra)+' '+DateToStr(vDate)+' '+
TimeToStr(vHour)+' '+dados);
ind:=1;           s:='';
        while ind<=(length(dados)) do begin
if dados[ind]=' ' thenbegin
if length(s)=1 thenID:=strtoint(s) // ID isolated
elsebegin
DecimalSeparator:='.';
                                        Chart1.Series[ID].AddXY(amostra, strtofloat(s));
end;
s:=''
end   //if dados[ind]=' '
elses:=s+dados[ind];
ind:=ind+1;
end    // While ind
end        // p>0
end         // t>0
end;            // Procedure
```

//*

procedure TForm1.FormCreate(Sender: TObject);

```
varic:integer;
begin
ifFileExists('DataBackup.txt') thenCopyFile
          ('DataBackup.txt', 'DataBackupOlder.txt', true);
myFile:='DataBackup.txt';
lf:=#10;
  for ic:= 0 to (Chart1.SeriesCount - 1) do Chart1.Series[ic].Clear;
sg:='';
amostra:=0;
SerialFlag:=false;
Start:=False;
  Memo1.Clear; Memo2.Clear; Memo3.Clear; Memo4.Clear;
Memo2.Font.Color:=clRed;      memo2.Lines.Add('No Port Connected');
Timer1.Enabled:=False;
```

```
timesampling:=1;
Memo4.Text:='1 Second';
memo3.Font.Color:=clRed;       memo3.Lines.Add('Paused');
end;

//********************************************************
procedure TForm1.Timer1Timer(Sender: TObject);
begin
Serial.WriteStr('1');
end;

//********************************************************
procedure TForm1.SaveData1Click(Sender: TObject);
varvLDate: TDateTime;    vLHour: TDateTime;
begin
if (Memo1.GetTextLen = 0) thenshowMessage
          (lf+'              There are not data to be saved              '+lf)
elsebegin
vLDate:=Now; vLHour := Time;
stg:=DateToStr(vLDate)+'-'+ TimeToStr(vLHour);
stg    := StringReplace(stg, ':', '',[rfReplaceAll, rfIgnoreCase]);
stg    := StringReplace(stg, '/', '',[rfReplaceAll, rfIgnoreCase]);
Memo1.Lines.SaveToFile('FT-'+stg+'.txt');
showMessage(#10+
                    'Temperature Data Saved in the Current Directory With Name: '
                    +#10+'FT-'+stg+'.txt'+#10);
end    // else
end;     // Procedure

//********************************************************
procedure TForm1.Savegraph1Click(Sender: TObject);
varvLDate:     TDateTime; vLHour: TDateTime;
begin
if (Memo1.GetTextLen = 0) thenshowMessage
           (lf+'              There are not data to be saved              '+lf)
elsebegin
vLDate:=Now;
vLHour := Time;
stg:=DateToStr(vLDate)+'-'+ TimeToStr(vLHour);
stg    := StringReplace(stg, ':', '',[rfReplaceAll, rfIgnoreCase]);
stg    := StringReplace(stg, '/', '',[rfReplaceAll, rfIgnoreCase]);
   Chart1.SaveToBitmapFile('GT-'+stg+'.bmp');
showMessage(#10+'Temperature Graph Saved innthe Current Directory With Name: '
                    +#10+'GT-'+stg+'.bmp'+#10);
```

```
end
end;

//*******************************************************
procedure TForm1.About1Click(Sender: TObject);
begin
showmessage (lf+'          WATER TEMPERATURE MEASUREMENT SYSTEM          '+lf+lf+
    '        For comments, contact the system designer:'+lf+lf+
'        João E. M. Perea Martins, Ph.D. '+lf+
'        Computer Science Department, School of Sciences (FC)         '+lf+
    '        São Paulo State University (UNESP), Brazil'+lf+lf+
'        E-mail:    joao.perea@unesp.br'+lf+lf+lf);
end;

//*******************************************************
procedure TForm1.Configuration1Click(Sender: TObject);
begin
ifSerial.Connectedthenbegin
showMessage(lf+lf+
                              'There is already an connected serial port.'+lf+
                              'First close it, and next connect a new port.'+lf+lf);
                           exit;
end
elsebegin
Serial.ShowSetupDialog;                    // Open options box
Serial.Open;                                // Open serial port
Serial.ClearBuffer(True, True);       //Serial.Buffer
      Memo2.Clear;
ifSerial.Connectedthenbegin      // Verify the openning operation
Memo2.Font.Color := clNavy;
memo2.Lines.Add(''+Serial.Port);
SerialFlag:=true;
showMessage(lf+lf+' The serial has been connected '+lf+
             ' Click on the Button START for the data aquisitionbeginning '+lf);
end
elsebegin
Memo2.Font.Color := clRed;
memo2.Lines.Add('Connection Error');
SerialFlag:=false;
end
end
end;

//*******************************************************
```

```
procedure TForm1.ClosePort1Click(Sender: TObject);
begin
Serial.Close;
Timer1.Enabled:=False;
SerialFlag:=false;
Memo2.Clear; Memo2.Font.Color := clRed;
memo2.Lines.Add('No Port Connected');
end;

//*******************************************************
procedure TForm1.SamplingInterval1Click(Sender: TObject);
varClickedOK: Boolean;
NewString, range:string;
begin
IfMessageDlg(lf+'The sampling interval can be in Seconds or Minutes'+lf+
  'Do you want to set it in seconds?'+lf, mtConfirmation,[mbyes,mbno],0)=mryes
thenrange:='second' else range:='minute';
ClickedOK := InputQuery('Sampling Interval',
                    'Type the measurments interval in '+range+'s', NewString);
ifClickedOKthenbegin
                    try
timesampling:=strtoint(NewString)
                    except on E: econverterror do begin
showMessage(lf+'         Invalid Parameter         '+lf+
                                                '         Type only integer numbers'+lf);
                        exit;
end;         // Except
end          // try
end                              // ClickedOKthenbegin
elseexit;                    // ifClickedOK
if range='second' thenTimer1.Interval:=strtoint(Newstring)*1000
elseTimer1.Interval:=strtoint(Newstring)*60000;
ifstrtoint(NewString)>1 then range:=range+'s';
Memo4.Clear; Memo4.Lines.Add(''+Newstring+' '+range);
end;

//*******************************************************
procedure TForm1.Clear1Click(Sender: TObject);
varic: integer;
begin
ifMessageDlg(lf+lf+'         Attention: It resets the data acquistion system.'+lf+
      '         All the numerical and graph data will be deleted. Continue ?'+lf+lf,
mtwarning,[mbYes,mbNo],    0)= mrNothen exit;
Memo1.Clear;
```

```
Timer1.Enabled:=False;
amostra:=0;
Serial.Close;
Memo2.Clear;
Memo2.Font.Color := clRed;
memo2.Lines.Add('No Port Connected');
timesampling:=1; Memo4.Text:='1 Second' ;
memo3.Lines.Add('Paused');
ifFileExists('DataBackup.txt') then
CopyFile('DataBackup.txt', 'DataBackupOlder.txt', true);
myFile:='DataBackup.txt';
for ic:= 0 to (Chart1.SeriesCount - 1) do Chart1.Series[ic].Clear;
end;

//*****************************************************************
procedure TForm1.FormClose(Sender: TObject; var Action: TCloseAction);
begin
if (Memo1.GetTextLen > 0) thenbegin
ifDialogs.MessageDlg(lf+'There are data on the system screen. '+lf+
        'If you exit they will be lost. However, you can save them first.'+lf+
        'Do you really want to exit?'+lf,
mtConfirmation, [mbYes, mbNo], 0) = mrYesthenApplication.Terminate
else Abort;
end;
Application.Terminate;
end;

//*****************************************************************
procedure TForm1.Exit1Click(Sender: TObject);
begin
if (Memo1.GetTextLen > 0) thenbegin
ifDialogs.MessageDlg(lf+'There are data on the system screen. '+lf+
        'If you exit they will be lost. However, you can save them first.'+lf+
        'Do you really want to exit?'+lf,
mtConfirmation, [mbYes, mbNo], 0) = mrYesthenApplication.Terminate
else Abort;
end;
Application.Terminate;
end;

//*****************************************************************
procedure TForm1.Button1Click(Sender: TObject);
begin
ifSerial.Connected=False thenbegin
```

```
showMessage(lf+'      First Open the Serial Communication Interface      '+lf);
SerialFlag:=false;
   Memo3.Clear; Memo3.Font.Color:=clRed; Memo3.Lines.Add('Paused');
end
elsebegin
Timer1.Enabled:=True; SerialFlag:=true;
   Memo3.Clear;    memo3.Font.Color:=clNavy; memo3.Lines.Add('In Progress');
end
end;

//*******************************************************************
procedure TForm1.Button2Click(Sender: TObject);
begin
Timer1.Enabled:=False;
SerialFlag:=false;
Memo3.Clear;       memo3.Font.Color:=clRed;
memo3.Lines.Add('Paused');
end;

//*******************************************************************
procedure TForm1.Timer2Timer(Sender: TObject);
begin
if (Memo1.GetTextLen > 0) thenMemo1.Lines.SaveToFile(myFile);
end;

//*******************************************************************
end.
```

References

Al Mamun MR, et al. 2024. IoT-based real-time biofloc monitoring and controlling system. Smart Agricultural Technology, 9: 100598. Doi: 10.1016/j.atech.2024.100598

Barros JRA. 2022. Temperature: A major climatic determinant of cowpea production. Acta Scientiarum. Agronomy, 45: e56812. Doi: 10.4025/actasciagron.v45i1.56812

Bogdan R, et al. 2023. Low-cost internet-of-things water-quality monitoring system for rural areas. Sensors, 23(8): 3919. Doi: 10.3390/s23083919

Cornelius D. 2021. Fearless Cross-Platform Development with Delphi: Expand your Delphi skills to build a new generation of Windows, web, mobile, and IoT applications (1st Edition). Packt Publishing Ltd

FAO (Food and Agriculture Organization of the United Nations), 2020. The State of World Fisheries and Aquaculture. Rome, Italy. Doi: 10.4060/ca9229en

Feigl M, Lebiedzinski K, Herrnegger M, Schulz K. 2021. Machine-learning methods for stream water temperature prediction. Hydrology and Earth System Sciences, 25(5): 2951-2977. Doi: 10.5194/hess-25-2951-2021

Indumathy M, et al. 2022. Modelling and control of plate heat exchanger with continuous high-temperature short time milk pasteurization process: A review. Chemical Engineering Journal Advances, 11: 100305. Doi: 10.1016/j.ceja.2022.100305

Islam, MD et al. 2020. Extreme ambient temperature effects in European seabass, Dicentrarchuslabrax: Growth performance and hemato-biochemical parameters. Aquaculture, 522: 735093. Doi: 10.1016/j.aquaculture.2020.735093

JGCM (Joint Committee for Guides in Metrology). 2012. International Vocabulary of Metrology – Basic and General Concepts and Associated Terms (VIM). International Bureau of Weights and Measures (BIPM) Ed, Sevres, France

Johnson MF, et al. 2024. Rising water temperature in rivers: Ecological impacts and future resilience. Wiley Interdisciplinary Reviews: Water, 11(4): e1724. Doi: 10.1002/wat2.1724

Kaushal SS, et al. 2010. Rising stream and river temperatures in the United States. Frontiers in Ecology and the Environment, 8(9): 461-466. Doi: 10.1890/090037

Lan X, Wu S, Du Q, Min L. 2024. The investigation of changes in bacterial community of pasteurized milk during cold storage. Foods, 13(3): 451. Doi: 10.3390/foods13030451

Li H. 2021. Research on big data analysis data acquisition and data analysis. IEEE International Conference on Artificial Intelligence, Big Data and Algorithms (CAIBDA), 162-165. Doi: 10.1109/CAIBDA53561.2021.00041

Liu S, et al. 2020. Global river water warming due to climate change and anthropogenic heat emission. Global and Planetary Change, 193, 103289. Doi: 10.1016/j.gloplacha.2020.103289

Loubna B, et al. 2007. A review of statistical water temperature models. Canadian Water Resources Journal, 32(3): 179-192. Doi: 10.4296/cwrj3203179

Mihai V, Rusu L. 2022. Improving the ventilation of machinery spaces with direct adiabatic cooling system. Inventions, 7(3): 78. Doi: 10.3390/inventions7030078

Padrón RS, et al. 2024. Extended range forecasting of stream water temperature with deep learning models, EGUsphere, 2024: 1-27. Doi: 10.5194/egusphere-2024-2591

Perea Martins JEM. 2022. Physical analysis of a waterproof temperature sensor responsiveness for agricultural applications. Journal of Agricultural Informatics, 13(2). Doi: 10.17700/jai.2022.13.2.623

Poloczanska E, et al. 2013. Global imprint of climate change on marine life. Nature Climate Change, 3: 919-925. Doi: 10.1038/nclimate1958

Rebaudo F, et al. 2023. A low-cost IoT network to monitor microclimate variables in ecosystems. Methods in Ecology and Evolution, 14(4): 1025-1034. Doi: 10.1111/2041-210X.14062

Sghaier A, et al. The Effects of Temperature and water on the seed germination and seedling development of rapeseed (*Brassica napus* L.). Plants (Basel), 11(21): 2819. Doi: 10.3390/plants11212819.

Simanjuntak D, Hakim L. 2022. Design and control system of temperature and water level in hydroponic plants. In Journal of Physics: Conference Series, 2193(1): 012018. IOP Publishing. Doi: 10.1088/1742-6596/2193/1/012018

Song Q. 2024. Multi-device universal automation data acquisition and integration system. IEEE Access, 12: 104503-104517. Doi: 10.1109/ACCESS.2024.3435386

Vehanen T, Tapio S, Ari H. 2023. Potential impact of climate change on salmonid smolt ecology. Fishes, 8(7): 382. Doi: 10.3390/fishes8070382

Venegas RM, Acevedo J, Treml EA. 2023. Three decades of ocean warming impacts on marine ecosystems: A review and perspective. Deep Sea Research Part II: Topical Studies in Oceanography, 212: 105318. Doi: 10.1016/j.dsr2.2023.105318

Wen Y, et al. 2023. Effects of drip irrigation timing and water temperature on soil conditions, cotton phenological period, and fiber quality under plastic film mulching. Agricultural Water Management, 287: 108435. Doi: 10.1016/j.agwat.2023.108435

Xu F, et al. 2021. Prediction of fish migration caused by ocean warming based on SARIMA model. Complexity, 2021(10): 1-9. Doi: 10.1155/2021/5553935

Zhang WJ. 2023. A desktop calculator for effect sizes: Towards the new statistics. Computational Ecology and Software, 13(4): 136-181. http://www.iaees.org/publications/journals/ces/articles/2023-13(4)/4-Zhang-Abstract.asp

Zhang WJ, Qi YH. 2024. ANOVA-nSTAT: ANOVA methodology and computational tools in the paradigm of new statistics. Computational Ecology and Software, 14(1): 48-67. http://www.iaees.org/publications/journals/ces/articles/2024-14(1)/4-Zhang-Abstract.asp

Zhu S, Piotrowski A.P. 2020. River/stream water temperature forecasting using artificial intelligence models: a systematic review. Acta Geophysics, 68: 1433-1442. Doi: https://doi.org/10.1007/s11600-020-00480-7