Supplementary Figure 1
Creepmeter and Rain Gauge Sketches

Arduino Sketch for LVDT Creepmeter

```
// Arduino Sketch for LVDT Creepmeter

#include <SD.h>
#define chipSelect 4
#define loopTime 10800000   // delay time in ms (1000ms = 1s) between loop iterations
File myFile;

void setup() {
  // put your setup code here, to run once:
  //Intialize the microcontroller's ports
Serial.begin(9600);
while (!Serial){
  Serial.print("Initializing SD card");
  if (!SD.begin(chipSelect)){
    Serial.println("Initialization failed!");
    while(1);
  }
  else{
    ;
    }
  ;
  }

}


void loop() {
  //Read the signal from the transducer
  int recSignal = analogRead(A0);
  //Convert voltage values to meters
  float convertedValue = recSignal * (13.07/1023);
  //Print converted value
  Serial.print("Distance = "); Serial.println(convertedValue); Serial.println("m");
  //Delay time in microseconds


  Serial.println("Creating data.txt. . ." );
  if (SD.exists("data.txt")) {
    Serial.println("data.txt exists");
    myFile = SD.open("data.txt", FILE_WRITE);
  if (myFile) {
    myFile.println(convertedValue);
    myFile.close();
    }
  }
  else{
    Serial.println("data.txt doesn't exist, creating file");
    myFile = SD.open("data.txt", FILE_WRITE);
    if (myFile) {
    myFile.println("Distance(mm),");
    myFile.println(convertedValue);
    myFile.close();
    }
  }
  myFile = SD.open("data.txt"); //Open the file
  if (myFile) {
    Serial.println("data.txt:");
    while (myFile.available()) { // If the file is present execute loop until done.
        Serial.write(myFile.read()); //Reading the text inside the file
    }
    myFile.close(); //Close the file after opening
    }
    else { //Display message if it was unsuccessfully opened.
    Serial.println("Error opening data.txt");
  }

  //For testing purposes only
  //SD.remove("data.txt");
  delay(loopTime);
}
```

Arduino Sketch for US-100 Creepmeter

```
// Arduino Sketch for Ultrasonic (US-100) Creepmeter

#include <SoftwareSerial.h>
#include <SD.h>
#include <SPI.h>
//#include <DFRobot_sim808.h> //library to access GPS data
//DFRobot_SIM808 sim808(&Serial);

#define txPin 8       // Connect pin 8 of Leonardo to US100 trigger/Tx
#define rxPin 9       // Connect pin 9 of Leonardo to US100 Echo/Rx
#define getDist  0x55  // send this to get distance in mm
#define getCels  0x50    // send this to get temp
#define loopTime 10800000   // delay time in ms (1000ms = 1s) between loop iterations
//#define loopTime 1000
#define maxWait  1000    // max time to wait for US100 response
int chipSelect = 4;      // Connect pin 4 of Leonardo to microSD card adapter

SoftwareSerial us100(rxPin,txPin);  // Set Leonardo for UART signaling
int mmDist;                 // measured distance in mm
uint8_t highDist,lowDist;       // bytes read in from US100
int cTemp;                  // temp in Celsius for reporting
float fTemp;                  // temp in Farenheit for reporting
File  myFile;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);           // talk to the controlling device
  while (!Serial){
    delay(1);
  }
  pinMode(rxPin, INPUT);        // Set pin directions for US100 communications
  pinMode(txPin, OUTPUT);
  us100.begin(9600);            // and start SoftSerial service

  Serial.print("Initializing SD card...");
  if (!SD.begin(chipSelect)) {
    Serial.println("Initialization failed!");
    while (1);

  }

}

void loop() {
  Serial.println("Start sampling loop for US-100 testing");

  us100.flush();
  us100.write(getDist);         // ask US-100 to get distance; wait for response
  for ( int i=0; (i<maxWait) && !us100.available(); i++ ) delay(1);
  if (us100.available()) {
    highDist = us100.read();    // get high byte
    for ( int i=0; (i<maxWait) && !us100.available(); i++ ) delay(1);
    if (us100.available()) {
      lowDist = us100.read();   // get low byte
      mmDist = highDist*256 + lowDist;
      }
    else {
      Serial.println("Incomplete response to distance request");
      mmDist = 0;
    }
  }
  else {
    Serial.println("No response to distance request");
    mmDist = 0;
    }

  delay(1);                     // let US100 catch its breath for the temperature sensor to work
  us100.flush();
  us100.write(getCels);
  for ( int j=0; (j<maxWait) && !us100.available(); j++ ) delay(1);
  if (us100.available()) {
    cTemp = us100.read() - 45;
    fTemp = cTemp * 1.8 + 32;
    }
    else {
      Serial.println("No response to temp request");
      cTemp = 0;
      fTemp = 0.0;
    }

  Serial.print("Distance = ");  Serial.print(mmDist); Serial.println(" m");
  Serial.print("Temp in C = ");  Serial.println(cTemp);
  Serial.print("Temp in F = "); Serial.println(fTemp);

  Serial.println("Creating data.txt. . ." );
  if (SD.exists("data.txt")) {
    Serial.println("data.txt exists");
    myFile = SD.open("data.txt", FILE_WRITE);
  if (myFile) {
    myFile.print(mmDist); myFile.print(",");myFile.println(cTemp);
    myFile.close();
    }
  }
  else{
    Serial.println("data.txt doesn't exist, creating file");
    myFile = SD.open("data.txt", FILE_WRITE);
    if (myFile) {
    myFile.print("Distance(mm),");  myFile.println("C");
    myFile.print(mmDist); myFile.print(",");myFile.println(cTemp);
    myFile.close();
    }
  }
  myFile = SD.open("data.txt"); //Open the file
  if (myFile) {
    Serial.println("data.txt:");
    while (myFile.available()) { // If the file is present execute loop until done.
       Serial.write(myFile.read()); //Reading the text inside the file
    }
    myFile.close(); //Close the file after opening
    }
    else { //Display message if it was unsuccessfully opened.
    Serial.println("Error opening data.txt");
  }
  //For testing purposes only
  //SD.remove("data.txt");
  delay(loopTime);
   }
```

Arduino Sketch for Zhafira Rain Gauge

```
// Arduino Sketch for Zhafira Rain Gauge
//Rainfall is the amount of water that falls on the ground surface during a certain period which is measured in millimeters (mm) above the horizontal surface.
//1 mm rainfall is the amount of rainwater that falls on the surface per unit area (m²) with a volume of 1 liter without any evaporating, seeping or flowing.

//Formula calculation
//Rainfall height (cm) = collected volume (mL) / collection area (cm²)
//Collector area (Funnel) 5.5cm x 3.5cm = 19.25 cm²
//Collection per tip we get by pouring 100ml of water into the collector is derived by counting how many times the water is wasted from the tip,
//In our calculations, water is wasted 70 times. 100ml / 70= 1.42mL per tip.
//So 1 tip is worth 1.42 / 19.25 = 0.07cm or 0.70 mm of precipitation.

// IMPORTANT
// Use pin D2 on Arduino, Voltage 5V, then upload this code

#include <SoftwareSerial.h>
#include <SD.h>
#include <SPI.h>

const int chipSelect = 4;
const int pin_interrupt = 2; // Menggunakan pin interrupt https://www.arduino.cc/reference/en/language/functions/external-interrupts/attachinterrupt/
long int count_tip = 0;
long int temp_count_tip = 0;
float cumulative_rainfall = 0.00;
float milimeter_per_tip = 0.70;
unsigned long interval = 1800000;  // 2 hours in milliseconds
unsigned long previousTime = 0;

File dataFile;

volatile boolean flag = false;

void calculate_cumulative_rainfall()
{
  flag = true;
}

void setup(){
  pinMode(pin_interrupt, INPUT);
  attachInterrupt(digitalPinToInterrupt(pin_interrupt), calculate_cumulative_rainfall, FALLING); // Will calculate the tip if the pin goes from HIGH to LOW logic
  printSerial();
  Serial.begin(9600);

  Serial.print("Initializing SD card...");
  if (!SD.begin(chipSelect)) {
    Serial.println("SD card Initialization failed!");
    return;
    }
  Serial.println("SD card initialized.");

  // Open file for writing
  dataFile = SD.open("rainfall.txt", FILE_WRITE);
  if (!dataFile) {
    Serial.println("Error opening file!");
  }
  dataFile.println("Rainfall data");
  dataFile.close();

  Serial.println("Rainfall data logging started.");
}

void loop()
{
  if (flag == true) // don't really need the == true but makes intent clear for new users
  {
    cumulative_rainfall += milimeter_per_tip; // Will increase in value when the tip is full
    count_tip++;
    delay(500);
    flag = false; // reset flag
  }
  cumulative_rainfall = count_tip * milimeter_per_tip;
  if ((count_tip != temp_count_tip)) // Print serial every 1 minute or when count_tip changes
  {
    printSerial();
  }
  unsigned long currentTime = millis();
  if (currentTime - previousTime >= interval) {
    previousTime = currentTime;
    printSerial();
    savetoSDcard();
  }

  temp_count_tip = count_tip;

}

void printSerial()
{
  Serial.print("Count tip = ");
  Serial.print(count_tip);
  Serial.println(" times ");
  Serial.print("Cumulative rainfall (mm): ");
  Serial.print(cumulative_rainfall, 1);
  Serial.println();
}
void savetoSDcard()
{
  File dataFile = SD.open("rainfall.txt", FILE_WRITE | O_APPEND);

  if (dataFile) {
    // Save rainfall data to file
  dataFile.print("Count tip = ");
  dataFile.print(count_tip);
  dataFile.print("; Cumulative Rainfall (mm): ");
  dataFile.println(cumulative_rainfall, 1);
  dataFile.close();
  } else {
    Serial.println("Error opening data file!");
  }

}
```