

Categories All Posts My Posts

Login / Sign up

cristian.pinto
Jun 06, 2021



How to extract temperature data from any Audiomoth's file?

in Device Support

Hi Audiomoth Fans,

I'm searching how to extract the temperature data from any audiomoth recording. I tried using seewave & tuneR from R environment. Does any way to extract this data?

**20210507_233000.WAV 5,8 MB**

Modified: Friday, 7 May 2021, 23:31

Add Tags...

▼ General:

Kind: Waveform audio

Size: 5.760.488 bytes (5,8 MB on disk)

Where: Macintosh HD ▸ Users ▸ crispintof
▸ Desktop ▸ 1 mes - 730PM Chile
UTC -4

Created: Friday, 7 May 2021, 23:29

Modified: Friday, 7 May 2021, 23:31

☐ Stationery pad☐ Locked

▼ More Info:

Duration: 01:00

Authors: AudioMoth

2423C2045F256546

Audio channels: Mono

Sample rate: 48 kHz

Bits per sample: 16

Comment: Recorded at 23:30:00

07/05/2021 (UTC) by

AudioMoth

2423C2045F256546 at

medium-high gain setting

while battery state was 3.8V

and temperature was 18.0C.

> Name & Extension:

7 Comments

Write a comment...

Sort by: Newest ▾



Alex Rogers
Jun 07, 2021



👍 Best Answer

Hi Cristian, There's a Python script in the Applications Notes folder - https://github.com/OpenAcousticDevices/Application-Notes/blob/master/AudioMoth_Temperature_Measurements.pdf - which will extract the temperature and battery data from all the files. The header comment is always in the same place in the files, and this script just parses it as text and extracts the relevant data. Alex

↑ 1 ↓



Unknown member
Mar 17, 2022



Replying to [Alex Rogers](#)

Thanks Alex.

♡ Like



Alex Rogers
Aug 31, 2022



This looks great. I don't think it will read negative temperatures so you should update that if it's an issue. The header is always less than 512 bytes long so you can just read the first 512 characters from each file. That might speed it up.

↑ 1 ↓



Kevin Tolan
Aug 31, 2022



Hello all, my coworker helped me out and created this R script. Very straight forward. It takes a while to run on a large number of files.

The export is a data.frame of metadata:

AudioMothID	File	Recorded	Battery_V	Temperature_C
24F319055F5775E5	file.wav	2021-04-14 23:53:39	4.2	23.1

```
# this lists all the *.wav files in a folder.
# if you have many files this will list them all with the full
# directory path
```

```
wavfiles <- list.files("C:/User/",
```

© 2023 Open Acoustic Devices

```
# extract the data you want, and put it into a data frame #run
function below first

allmetadata <- do.call(rbind,lapply(wavfiles, getWavMetaData))

# BELOW IS A FUNCTION TO EXTRACT ALL THE METADATA #

# Params
# @wavfiles = path to *.wav files of interest to extract data
# @filenames = optional vector of file names for the *.wav files
# - should be same length
#           as wavfiles and in same order

getWavMetaData <- function(wavfiles, filenames = NULL){

# parse out the file name - this assumes the full path is given
#
if(is.null(filenames)){
  if(sum(grep(pattern = "\\/", wavfiles))!=0L){
    fileID <- gsub(".*\\/","",wavfiles)}else{
    fileID <- wavfiles
  }
}else{
  fileID <- filenames
}

# how many characters at in the file #

size <- readBin(wavfiles, integer(), size = 4, endian =
"little")

# read in the data #

binDat <- suppressWarnings(readBin(wavfiles, "character", n =
size))

# determine which element of the list has what you want #

metadat <- binDat[which(grepl("Recorded",binDat))]

# grab the different metadata components #

date_time <- regexpr(pattern = "Recorded at
\\d*:\\d*:\\d*\\d*\\/\\d*\\/\\d*",
```

[Comment](#)[Follow Post](#) 433 views 7 comments

Similar Posts

[WAVESURFER and
AUDIOMOTH files](#)[API or interface to accessing
Data Files without SD card
removal](#)[Irregular file lengths on
Audiomoth 1.2.0](#)

```

{2}\\V[0-9]{2}\\V[0-9]{4}",
      text = metadat[1])

# Get voltage data location in string #
voltLoc <- regexpr("[0-9].[0-9]V", metadat[1])

# Get temp. data location in string #
tempLoc <- regexpr("[0-9]{1,2}.[0-9]C", metadat[1])

# Get Audiomoth ID location #
audiomothLoc <- regexpr(pattern = "AudioMoth (.*) at ",
  metadat[1])

# get the date in character form as is the file #
timechar <- substr(metadat[1],start = timeLoc, stop =
timeLoc+attr(timeLoc,"match.length")-1)

# make the date into a useful time format #
recordtime <- as.POSIXct(timechar, format = "%H:%M:%S %d/%m/%Y",
  tz = "UTC")

# Get state of battery in volts *note -2 to get rid of space and
V*
batteryState <- substr(metadat[1], start = voltLoc, stop =
voltLoc+attr(voltLoc,"match.length")-2)

# Get state of battery in C *note -2 to get rid of space and C*
tempRecord <- substr(metadat[1], start = tempLoc, stop =
tempLoc+attr(tempLoc,"match.length")-2)

# audioMoth id #
amID <- substr(metadat[1],
  start = audiomothLoc+10,
  stop =
(audiomothLoc)+attr(audiomothLoc,"match.length")-5)

return(data.frame(AudioMothID = amID,
  File = fileID,
  Recorded = recordtime,
  Battery_V = as.numeric(batteryState),
  Temperature_C = as.numeric(tempRecord)))
}

```

All thanks to Mike Hallworth

Bárbara Freitas

Jun 08, 2021



Hi Alex, Any chances of providing an R script also?



Alex Rogers

Jun 08, 2021

Replying to [Bárbara Freitas](#)

I don't use R but it should be very straightforward for someone who does to write the equivalent script.



p.holderried

Mar 22, 2022

Replying to [Bárbara Freitas](#)

The function `audiomoth_wave` from the R-package `sonicscrewdriver` is exactly what you are looking for. You can find the vignette [here](#). It didn't work properly the last time I tried to use it. I made a few changes to the code and it has worked fine for me so far, All the function does is extract the text from the wav header and split it before/after certain words or a certain number of characters.

```
library(stringr)
library(dplyr)

audiomoth_header <-
  function(filename) {

    f <- readBin(filename, "character", n = 25)

    for(i in 1:length(f)) {
      if(regexpr("Recorded", f[i]) == 1) {
        n = i
      }
    }

    raw <- f[n]
    if (regexpr("Recorded", raw) != 1) {
      print("No audiomoth comment field found.")
      return(FALSE)
    }

    r <- regexpr("Recorded at", raw) + 12
```

```

    day <- as.Date(substr(raw, r, r+10), format =
"%d/%m/%Y", tz = "UTC")

    r <- regexpr("AudioMoth", raw) + 10
    serial <- substr(raw, r, r+16)

    r <- regexpr("AudioMoth [0-9|A-Z]{16} at ", raw) + 30
    l <- regexpr("gain setting", raw) - r -2
    gain <- substr(raw, r, r+1)

    if ( regexpr("less than 2\\.5V", raw) != -1) {
      voltage <- "<2.5"
    } else if (regexpr("greater than 4\\.9V", raw) != -1)
    {
      voltage <- ">4.9"
    } else {
      r <- regexpr("[0-9].[0-9]V", raw)
      voltage <- substr(raw, r, r+2)
    }

    temp <- str_extract(raw, "(?<=temperature was )
[[:digit:]]{1,2}[[:punct:]]{1}[[:digit:]]{1,2}")

    filter <- FALSE
    filter_limit <- FALSE
    filter_limit <-
      if (regexpr("Low-pass filter applied", raw) != -1) {
        filter <- "Low-pass"
      }
    if (regexpr("Band-pass filter applied", raw) != -1) {
      filter <- "Band-pass"
    }
    if (regexpr("High-pass filter applied", raw) != -1) {
      filter <- "High-pass"
    }
    if (is.element(filter, c("Low-pass", "High-pass"))) {
      r <- regexpr("frequency of", raw) + 13
      l <- regexpr("kHz", raw) - 1
      filter_limit <- substr(raw,r,l)
    }
    if (filter == "Band-pass") {
      al <- regexpr("frequencies of", raw) + 15
      ar <- regexpr("kHz", raw) - 1
      bl <- regexpr("kHz and", raw) + 8
      br <- regexpr("kHz\\.\"", raw) - 1
    }
  }
}

```

```

    if (regexpr("microphone change.", raw) != -1) {
      cancelled <- "microphone change"
    } else if (regexpr("change of switch position.", raw)
!= -1) {
      cancelled <- "change of switch position"
    } else if (regexpr("low voltage.", raw) != -1) {
      cancelled <- "low voltage"
    } else if (regexpr("file size limit.", raw) != -1) {
      cancelled <- "file size limit"
    } else if (regexpr("change of switch position", raw)
!= -1) {
      cancelled <- "change of switch position"
    } else {cancelled <- FALSE}

header <- list(
  "raw" = raw,
  "date_time" = date_time,
  "day" = day,
  "serial" = serial,
  "gain" = gain,
  "voltage" = as.numeric(voltage),
  "temperature" = as.numeric(temp),
  "filter" = filter,
  "filter.limit" = filter_limit,
  "cancelled" = cancelled
)
return(header)

}

wav_paths <- c("path/to/files/file1.wav",
"path/to/files/file2.wav")
header <- lapply(wav_paths, function(w)
{audiomoth_header(w)})

```

 Like

 1 Like

Write a comment...