FLIRjpg_processing

Rosalee Elting

3/4/2022

IMAGE PROCESSING CODE *This code is designed to allow users to process radiometric jpeg images produced from a FLIR camera into raw data where each pixel is assigned a temperature. This workflow was designed to then analyze these raw data files in ImageJ for temperature analysis. The code is run in two chunks, with details oulined below. ThermImage package was used for these analyses and adapted to a loop. This package was created by Glenn Tattersall and can be found here*

CHUNK 1 *This chunk will create a function that pulls jpgs from an input folder and places the raw product into an output folder. In Chunk 2, you will adapt the directories to your loacl machine where you'd like these input and output folders. Before running this code you will need to install Exiftools and follow the instructions below*

MAC USERS:*In chunk 1 below, change any statement that says "exiftoolspath" to read "exiftoolpath="installed"* **WINDOWS USERS**:*Save the downloaded ExifTools into your machines C:/ folder. In chunk 1 below, change any statement that says"exiftools path" to read "exiftoolspath"'C:/". If you need to save it in another location, that's okay, just reflect that in this directory.* **You can do either of the code replacements above with the Ctrl/Cmd + F function in R and replace with your needed path. Code will not run if R cannot find your local Exiftools** *Things to change in this chunk: Exiftools location*

```
image_processing <- function(input,output) {
  require(Thermimage);require(ggplot2);require(tidyr);require(Thermimage);require(fields)
  images <- list.files(path=input, pattern= "*.jpg")
  #images <- list.files(path="/Users/kevinl.epperly/Dropbox/R_scripts/bill_thermal-main/input", pattern
  names <- unique(images)
  nimages <- length(images)
  for (i in 1:nimages){
    #set working directory to makes sure the code below is using the images in the input folder
    setwd(input)
    #read Flir jpg into img
    img<-readflirJPG(names[i], exiftoolpath="C:/")
    #get img dimensions
    dim <- dim(img)
    #get camera settings (cams) with flir settings
    cams <-flirsettings(names[i], exiftoolpath="C:/", camvals="")
    #save heading of camsinfo
    head <- head(cbind(cams$Info), 20)
    #save plancks from settings
    plancks<-flirsettings(names[i], exiftoolpath="C:/", camvals="-*Planck*")
    unlist(plancks$Info)
    cbind(unlist(cams$Dates))
    #assign variables information in camsinfo

    ObjectEmissivity<-  cams$Info$Emissivity              # Image Saved Emissivity - should be ~0.95 or
    dateOriginal<-cams$Dates$DateTimeOriginal             # Original date/time extracted from file
    dateModif<-   cams$Dates$FileModificationDateTime     # Modification date/time extracted from file
    PlanckR1<-    cams$Info$PlanckR1                      # Planck R1 constant for camera
    PlanckB<-     cams$Info$PlanckB                       # Planck B constant for camera
    PlanckF<-     cams$Info$PlanckF                       # Planck F constant for camera
    PlanckO<-     cams$Info$PlanckO                       # Planck O constant for camera
    PlanckR2<-    cams$Info$PlanckR2                      # Planck R2 constant for camera
    ATA1<-        cams$Info$AtmosphericTransAlpha1        # Atmospheric Transmittance Alpha 1
    ATA2<-        cams$Info$AtmosphericTransAlpha2        # Atmospheric Transmittance Alpha 2
    ATB1<-        cams$Info$AtmosphericTransBeta1         # Atmospheric Transmittance Beta 1
    ATB2<-        cams$Info$AtmosphericTransBeta2         # Atmospheric Transmittance Beta 2
    ATX<-         cams$Info$AtmosphericTransX             # Atmospheric Transmittance X
    OD<-          cams$Info$ObjectDistance                # object distance in metres
    FD<-          cams$Info$FocusDistance                 # focus distance in metres
    ReflT<-       cams$Info$ReflectedApparentTemperature  # Reflected apparent temperature
    AtmosT<-      cams$Info$AtmosphericTemperature        # Atmospheric temperature
    IRWinT<-      cams$Info$IRWindowTemperature           # IR Window Temperature
    IRWinTran<-   cams$Info$IRWindowTransmission          # IR Window transparency
    RH<-          cams$Info$RelativeHumidity              # Relative Humidity
    h<-           cams$Info$RawThermalImageHeight         # sensor height (i.e. image height)
    w<-           cams$Info$RawThermalImageWidth          # sensor width (i.e. image width)

    #create a sting from image
    str(img)
    #make data frame of temperature from raw data and show resulting string
    temperature<-raw2temp(img, ObjectEmissivity, OD, ReflT, AtmosT, IRWinT, IRWinTran, RH,
                          PlanckR1, PlanckB, PlanckF, PlanckO, PlanckR2,
                          ATA1, ATA2, ATB1, ATB2, ATX)
    str(temperature)
    #plot temperature data, can add rotation note here (ie. trans="rotate270.matrix") or the color pale

    plotTherm(temperature, h=h, w=w, minrangeset=21, maxrangeset=32)
    #set new WD for the file, since I'd like it save in outputs
    setwd(output)
    #setwd("/Users/kevinl.epperly/Dropbox/R_scripts/bill_thermal-main/output")
    #Write the raw folder into the outpur folder.
    writeFlirBin(as.vector(t(temperature)), templookup=NULL, w=w, h=h, I="", rootname=cams$Info$FileName
  }
}
```

CHUNK 2 *This Chunk will run the function we made in chunk 1 called* **Image processing** *This will pull the input from images that you have saved and place them in an output folder. Therfore for this chunk you will need to put your machine's directory into the argument below. They can be identical, however i recommend making subfolers within your project titled "input" and "output". You will use their location in the path below for your machine. Thinks to change in this chunk: the paths of your input and output on your local machine*

```
#The format for this code is image_processing("input location", "output location")
#Sample: image_processing("C:/Users/Mellisuga/Documents/R/bill_thermal/input","C:/Users/Mellisuga/Docum
image_processing("C:/Users/Mellisuga/Documents/R/bill_thermal/input","C:/Users/Mellisuga/Documents/R/bil
```

```
## Loading required package: Thermimage
## Warning: package 'Thermimage' was built under R version 4.1.2
```

```
## Loading required package: ggplot2
```

```
## Loading required package: tidyr
```

```
## Loading required package: fields
```

```
## Warning: package 'fields' was built under R version 4.1.2
```

```
## Loading required package: spam
```

```
## Warning: package 'spam' was built under R version 4.1.2
```

```
## Spam version 2.8-0 (2022-01-05) is loaded.
## Type 'help( Spam)' or 'demo( spam)' for a short introduction
## and overview of this package.
## Help for individual functions is also obtained by adding the
## suffix '.spam' to the function name, e.g. 'help( chol.spam)'.
```

```
##
## Attaching package: 'spam'
```

```
## The following objects are masked from 'package:base':
##
##     backsolve, forwardsolve
```

```
## Loading required package: viridis
```

```
## Loading required package: viridisLite
```

```
##
## Try help(fields) to get started.
```

```
##  int [1:480, 1:640] 11703 11701 11717 11702 11709 11713 11722 11723 11730 11734 ...
##  num [1:480, 1:640] 9.01 8.99 9.12 9 9.05 ...
```

FLIRjpg_processing_files/figure-latex/unnamed-chunk-2-1.pdf

```
##  int [1:480, 1:640] 13632 13672 13746 13837 13931 13982 14093 14157 14165 14259 ...
##  num [1:480, 1:640] 23.4 23.7 24.2 24.8 25.5 ...
```

FLIRjpg_processing_files/figure-latex/unnamed-chunk-2-2.pdf

```
##  int [1:480, 1:640] 13716 13700 13701 13694 13695 13696 13690 13713 13693 13692 ...
##  num [1:480, 1:640] 24 23.9 23.9 23.8 23.8 ...
```

FLIRjpg_processing_files/figure-latex/unnamed-chunk-2-3.pdf

Your raw files should now be available to view in your output folder!

This Rmd Can be knit into a HTML or PDF and will create a report on your photos for future reference, should you choose