Package: vprr (via r-universe)

October 27, 2024

Type Package

Title Processing and Visualization of Video Plankton Recorder Data

Version 0.3.0

Maintainer Emily O'Grady <vprrcontact@gmail.com>

Description An oceanographic data processing package for analyzing and visualizing Video Plankton Recorder data. This package was developed at 'Bedford Institute of Oceanography'. Functions are designed to process automated image classification output and create organized and easily portable data products.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

Depends R (>= 2.10)

Imports ggplot2, oce, dplyr, interp, magick, gsw, tidyr, stringr, metR, gridExtra, lubridate, cmocean, withr, usethis, data.table, fs, rjson

Suggests knitr, rmarkdown, testthat, markdown

VignetteBuilder knitr

BuildVignettes true

Roxygen list(markdown = TRUE)

URL https://eogrady21.github.io/vprr/

Repository https://eogrady21.r-universe.dev

RemoteUrl https://github.com/eogrady21/vprr

RemoteRef HEAD

RemoteSha f70ddde7f7007606bb2c23a5c61f6c9071744f9a

Contents

`bin_calculate` *Get bin averages for VPR and CTD data*

Description

Bins CTD data for an individual cast to avoid depth averaging across tow-yo's

Usage

```
bin_calculate(data, binSize = 1, imageVolume, rev = FALSE)
```

Arguments

`data`	ctd data frame object including scan, salinity, temperature, depth, conductivity, time, fluor_ref, turbidity_ref, turbidity_mv, altitude, cast_id, n_roi
`binSize`	the height of bins over which to average, default is 1 metre
`imageVolume`	the volume of VPR images used for calculating concentrations (mm^3)
`rev`	logical value, if TRUE, binning will begin at bottom of each cast, this controls data loss due to uneven binning over depth. If bins begin at bottom, small amounts of data may be lost at the surface of each cast, if binning begins at surface (rev = FALSE), small amounts of data may be lost at bottom of each cast

Details

Image volume calculations can change based on optical setting of VPR as well as autodeck setting used to process images For IML2018051 (S2) image volume was calculated as 108155 mm^3 by seascan (6.6 cubic inches) For COR2019002 S2 image volume was calculated as 83663 mm^3 and S3 image volume was calculated as 366082 mm^3. Used internally (`bin_cast`) after `ctd_cast` on a single ascending or descending section of VPR cast

Note

binSize should be carefully considered for best results

Depth is used for calculations! Please ensure depth is included in data frame using swDepth

Author(s)

E. Chisholm, K. Sorochan

bin_cast *Bin vpr data*

Description

Formats oce style VPR data into depth averaged bins using `ctd_cast` and `bin_calculate` This function is used inside `concentration_category`

Usage

```
bin_cast(
  ctd_roi_oce,
  imageVolume,
  binSize,
  rev = FALSE,
  breaks = NULL,
  cutoff = 0.1
)
```

Arguments

`ctd_roi_oce`	oce ctd format VPR data from `vpr_oce_create`
`imageVolume`	the volume of VPR images used for calculating concentrations (mm^3)
`binSize`	passed to `bin_calculate`, determines size of depth bins over which data is averaged
`rev`	logical value,passed to `bin_calculate` if TRUE, binning will begin at bottom of each cast, this controls data loss due to uneven binning over depth. If bins begin at bottom, small amounts of data may be lost at the surface of each cast, if binning begins at surface (rev = FALSE), small amounts of data may be lost at bottom of each cast
`breaks`	Argument passed to ctdFindProfiles
`cutoff`	Argument passed to ctdFindProfiles

Details

Image volume calculations can change based on optical setting of VPR as well as autodeck setting used to process images For IML2018051 (S2) image volume was calculated as 108155 mm^3 by seascan (6.6 cubic inches) For COR2019002 S2 image volume was calculated as 83663 mm^3 and S3 image volume was calculated as 366082 mm^3

Value

A dataframe of depth averaged bins of VPR data over an entire cast with calculated concentration values

`category_conc_n` *A binned data frame of concentration data per category*

Description

A 'binned' dataframe from sample VPR data, including concentrations of each category, where each data point represents a 5 metre bin of averaged VPR data. Produced using `vpr_roi_concentration`

Usage

```
category_conc_n
```

Format

A dataframe with 21 variables

depth Depth calculated from pressure in metres

min_depth The minimum depth of the bin in metres

max_depth The maximum depth of the bin in metres

depth_diff The difference between minimum and maximum bin depth in metres

min_time_s The minimum time in seconds of the bin

max_time_s The maximum time in seconds of the bin

time_diff_s The difference between minimum and maximum time in a bin, in seconds

n_roi_bin The number of ROI observations in a bin

conc_m3 The concentration of ROIs in a bin, calculated based on image volume and number of frames per bin

temperature Temperature measured from the VPR CTD in celsius (averaged within the bin)

salinity Salinity measured from the VPR CTD (averaged within the bin)

density sigma T density calculated from temperature, salinity and pressure (averaged within the bin)

fluorescence Fluorescence measured by the VPR CTD in millivolts (uncalibrated) (averaged within the bin)

turbidity Turbidity measured by the VPR CTD in millivolts (uncalibrated) (averaged within the bin)

avg_hr The mean time in which bin data was collected, in hours

n_frames The number of frames captured within a bin

vol_sampled_bin_m3 The volume of the bin sampled in metres cubed

toyo Identifier of the tow-yo section which bin is a part of, either ascending or descending, appended by a number

max_cast_depth The maximum depth of the entire VPR cast

category The category in which ROIs in bin have been classified by Visual Plankton

station Station identifier provided during processing

concentration_category *Binned concentrations*

Description

This function produces depth binned concentrations for a specified category. Similar to `bin_cast` but calculates concentrations for only one category. Used inside `vpr_roi_concentration`

Usage

```
concentration_category(
  data,
  category,
  binSize,
  imageVolume,
  rev = FALSE,
  breaks = NULL,
  cutoff = 0.1
)
```

Arguments

`data`	dataframe produced by processing internal to vpr_roi_concentration
`category`	name of category isolated
`binSize`	passed to `bin_calculate`, determines size of depth bins over which data is averaged
`imageVolume`	the volume of VPR images used for calculating concentrations (mm^3)
`rev`	Logical value defining direction of binning, FALSE - bins will be calculated from surface to bottom, TRUE- bins will be calculated bottom to surface
`breaks`	Argument passed to ctdFindProfiles
`cutoff`	Argument passed to ctdFindProfiles

Details

Image volume calculations can change based on optical setting of VPR as well as autodeck setting used to process images For IML2018051 (S2) image volume was calculated as 108155 mm^3 by seascan (6.6 cubic inches) For COR2019002 S2 image volume was calculated as 83663 mm^3 and S3 image volume was calculated as 366082 mm^3

Author(s)

E. Chisholm

ctd_cast *Isolate ascending or descending section of ctd cast*

Description

This is an internal step required to bin data

Usage

```
ctd_cast(
  data,
  cast_direction = "ascending",
  data_type,
  cutoff = 0.1,
  breaks = NULL
)
```

Arguments

`data`	an oce ctd object
`cast_direction`	’ascending’ or ’descending’ depending on desired section
`data_type`	specify ’oce’ or ’df’ depending on class of desired output
`cutoff`	Argument passed to ctdFindProfiles
`breaks`	Argument passed to ctdFindProfiles

Value

Outputs either data frame or oce ctd object

Note

`ctdFindProfiles` arguments for `minLength` and `cutOff` were updated to prevent losing data (EC 2019/07/23)

Author(s)

K Sorochan, E Chisholm

ctd_dat_combine *VPR CTD data*

Description

A dataframe including all CTD parameters from the VPR CTD, produced by `vpr_ctd_read`

Usage

```
ctd_dat_combine
```

Format

A dataframe with 15 variables

time_ms Time stamp when ROI was collected (milliseconds)

conductivity Conductivity collected by the VPR CTD

pressure Pressure measured from the VPR CTD in decibars

temperature Temperature measured from the VPR CTD in celsius

salinity Salinity measured from the VPR CTD

fluor_ref A reference fluorescence baseline provided in millivolts by the VPR CTD for calibrating fluorescence_mv data

fluorescence_mv Fluorescence in millivolts from the VPR CTD (uncalibrated)

turbidity_ref A reference turbidity baseline provided in millivolts for calibrating turbidity_mv

turbidity_mv Turbidity in millivolts from the VPR CTD (uncalibrated)

altitude_NA Altitude data from the VPR CTD

day Day on which VPR data was collected (from AutoDeck)

hour Hour during which VPR data was collected (from AutoDeck)

station Station idnetifier provided during processing

sigmaT Density caluclated from temperature, pressure and salinity data

depth Depth in metres caluclated form pressure

`ctd_df_cols` *Read CTD data (SBE49) from CTD- VPR package*

Description

Internal use `vpr_ctd_read`

Usage

```
ctd_df_cols(x, col_list)
```

Arguments

`x`	full filename (ctd .dat file)
`col_list`	list of CTD data column names

Details

WARNING This is hard coded to accept a specific order of CTD data columns. The names and values in these columns can change based on the specific instrument and should be updated before processing data from a new VPR.

Text file format .dat file Outputs ctd dataframe with variables time_ms, conductivity, temperature, pressure, salinity

Author(s)

K. Sorochan, E. Chisholm

`ctd_roi_merge` *VPR CTD data combined with tabulated ROIs*

Description

A dataframe representing CTD data which has been merged with tabulated ROIs in each category, produced by `vpr_ctdroi_merge`

Usage

```
ctd_roi_merge
```

Format

A dataframe with 28 variables

time_ms Time stamp when ROI was collected (milliseconds)

conductivity Conductivity collected by the VPR CTD

pressure Pressure measured from the VPR CTD in decibars

temperature Temperature measured from the VPR CTD in celsius

salinity Salinity measured from the VPR CTD

fluor_ref A reference fluorescence baseline provided in millivolts by the VPR CTD for calibrating fluorescence_mv data

fluorescence_mv Fluorescence in millivolts from the VPR CTD (uncalibrated)

turbidity_ref A reference turbidity baseline provided in millivolts for calibrating turbidity_mv

turbidity_mv Turbidity in millivolts from the VPR CTD (uncalibrated)

altitude_NA Altitude data from the VPR CTD

day Day on which VPR data was collected (from AutoDeck)

hour Hour during which VPR data was collected (from AutoDeck)

station Station identifier provided during processing

sigmaT Density caluclated from temperature, pressure and salinity data

depth Depth in metres caluclated form pressure

roi ROI identification number

categories For each category name (eg. bad_image_blurry, Calanus, krill), there is a line in the dataframe representing the number of ROIs identified in this category

n_roi_total Total number of ROIs in all categories for each CTD data point

`ctd_roi_oce` *VPR data including CTD and ROI information*

Description

An oce formatted CTD object with VPR CTD and ROI data from package example data set.

Usage

```
ctd_roi_oce
```

Format

An oce package format, a 'CTD' object with VPR CTD and ROI data (1000 data rows)

insertRow — *INTERNAL USE ONLY quick data frame function from github to insert row inside dat frame*

Description

INTERNAL USE ONLY quick data frame function from github to insert row inside dat frame

Usage

```
insertRow(existingDF, newrow, r)
```

Arguments

`existingDF`	data frame
`newrow`	new row of data
`r`	index of new row

isopycnal_calculate — *Get vector to draw isopycnal lines on TS plot Used internally to create TS plots*

Description

Get vector to draw isopycnal lines on TS plot Used internally to create TS plots

Usage

```
isopycnal_calculate(sal, pot.temp, reference.p = 0)
```

Arguments

`sal`	salinity vector
`pot.temp`	temperature vector in deg C
`reference.p`	reference pressure for calculation, set to 0

Note

: modified from source:`https://github.com/Davidatlarge/ggTS/blob/master/ggTS_DK.R`

Author(s)

E. Chisholm

normalize_matrix *Normalize a matrix*

Description

take each element of matrix dived by column total

Usage

```
normalize_matrix(mat)
```

Arguments

mat	a matrix to normalize

Details

Make sure to remove total rows before using with VP data

Note

used internally for visualization of confusion matrices

package-imports *Packages*

Description

Packages

px_to_mm *Get conversion factor for pixels to mm for roi measurements*

Description

Used internally

Usage

```
px_to_mm(x, opticalSetting)
```

Arguments

`x` an aidmea data frame (standard) to be converted into mm from pixels

`opticalSetting` the VPR setting determining the field of view and conversion factor between mm and pixels

Details

converts pixels to mm using conversion factor specific to optical setting

Options for opticalSetting are ’S0’, ’S1’, ’S2’, or ’S3’

`read_aid_cnn` *Read aid files produced by automated classification*

Description

Read aid files produced by automated classification

Usage

```
read_aid_cnn(aid_file)
```

Arguments

`aid_file` a file path to an aid file produced by automated classification (with ROI path and probability value)

Value

ROI path and probability values in a table

`roimeas_dat_combine` *VPR measurement data calculated by Visual Plankton*

Description

A data frame of measurement information for each ROI in the sample data set including long axis length, perimeter and area, produced by `vpr_autoid_read`

Usage

```
roimeas_dat_combine
```

Format

A data frame with 12 variables

roi Unique ROI identifier - 10 digit

category Category in which ROI has been classified

day_hour day and hour in which data was collected (from Autodeck)

Perimeter The perimeter of the ROI in millimeters

Area The area of the ROI in millimeters

width1 Width at a first point of the ROI in millimetres (defined in more detail in VPR manual)

width2 Width at a second point of the ROI in millimetres (defined in more detail in VPR manual)

width3 Width at a third point of the ROI in millimetres (defined in more detail in VPR manual)

short_axis_length The length in millimeters of the ROI along the shorter axis

long_axis_length The length in millimeters of the ROI along the longer axis

station Station identifier provided in processing

time_ms Time stamp when ROI was collected in milliseconds

`roi_dat_combine` *VPR ROI data*

Description

A dataframe including VPR ROI data from the sample dataset, produced by `vpr_autoid_read`

Usage

```
roi_dat_combine
```

Format

A dataframe with 13 variables

roi Unique ROI identifier - 8 digit

categories For each category name (eg. bad_image_blurry, Calanus, krill), there is a line in the dataframe representing the number of ROIs identified in this category

time_ms Time stamp when ROI was collected (milliseconds)

size_df_f *VPR size information dataframe*

Description

A sample data frame of size information from Visual Plankton outputs, processed using `vpr_ctdroisize_merge`

Usage

```
size_df_f
```

Format

An object of class `data.frame` with 14 rows and 14 columns.

Details

@format A dataframe with 14 variables including

frame_ID Unique identifier for each VPR frame

pressure Pressure measured from the VPR CTD in decibars

temperature Temperature measured from the VPR CTD in celsius

salinity Salinity measured from the VPR CTD

sigmaT Density calculated from temperature, salinity and pressure

fluorescence_mv Fluorescence measured by the VPR CTD in millivolts (uncalibrated)

turbidity_mv Turbidity measured by the VPR CTD in millivolts (uncalibrated)

roi Unique ROI identification number - 10 digits, 8 digit millisecond time stamp and two unique digits to denote multiple ROIs within a millisecond

category Category in which ROI has been classified by Visual Plankton

day_hour Day and hour in which data was collected, from AutoDeck processing

long_axis_length The length of the longest axis of the ROI image, measured by Visual Plankton

station Station identifier provided during processing

time_ms Time stamp when ROI was collected (milliseconds)

roi_ID ROI identification number- 8 digit time stamp, without unique 2 digit ending

vpr_autoid_check *Checks manually created aid files for errors*

Description

Checks for empty files, with an option to delete them. Then checks all the data for duplicated or missing ROIs which would indicate a problem with vpr_autoid_create()

Usage

```
vpr_autoid_check(new_autoid, original_autoid, cruise, dayhours)
```

Arguments

new_autoid	file path to autoid folder eg. C:/data/CRUISENAME/autoid/ (produced by vpr_autoid_create())
original_autoid	file path to original autoid folder (produced by automated classification)
cruise	name of cruise which is being checked
dayhours	chr vector, of unique day and hour values to check through (format d123.h12)

Value

text file (saved in working directory) named CRUISENAME_aid_file_check.txt

Author(s)

E Chisholm

vpr_autoid_copy *Copy VPR images into folders*

Description

Organize VPR images into folders based on classifications provided by visual plankton

Usage

```
vpr_autoid_copy(
  new_autoid,
  roi_path,
  day,
  hour,
  cast,
  station,
  threshold = NULL,
  org = "dayhour"
)
```

Arguments

`new_autoid` A file path to your autoid folder where data is stored eg. "C:/data/cruise_X/autoid/"

`roi_path` (optional) provide if ROI data has been moved since autoid files were created (if path strings in aid files do not match where data currently exists), a file path where ROI data is stored (up to "rois" folder)

`day` character string representing numeric day of interest (3 chr)

`hour` character string representing hour of interest (2 chr)

`cast` character string, VPR cast number of interest (3 chr)

`station` character string, station name of interest (eg. "Shediac")

`threshold` (optional) a numeric value, supplied only if you are copying images based on automated classifications, only images below this threshold of confidence will be copied for manual classification. Default is set to NULL.

`org` chr value, if 'station', images will be output in folders labelled by station, if 'dayhour', images will be output in folders labelled by day and hour

Value

organized file directory where VPR images are contained with folders, organized by day, hour and classification, inside your autoid folder

Note

this function uses tidy paths, see fs::path_tidy() for more info

---

`vpr_autoid_create` *Modifies aid and aid mea files based on manual reclassification*

Description

Modifies aid and aid mea files based on manual reclassification

Usage

```
vpr_autoid_create(
  reclassify,
  misclassified,
  basepath,
  day,
  hour,
  mea = TRUE,
  categories
)
```

Arguments

`reclassify` list of reclassify files (output from vpr_manual_classification())

`misclassified` list misclassify files (output from vpr_manual_classification())

`basepath` path to folder containing autoid files (e.g., ’extdata/COR2019002/autoid’)

`day` day identifier for relevant aid & aidmeas files

`hour` hour identifier for relevant aid & aidmeas files

`mea` logical indicating whether or not there are accompanying measurement files to be created

`categories` A list object with all the potential classification categories

Author(s)

E. Chisholm

Examples

```
## Not run:
basepath <- 'E:/autoID_EC_07032019/'
day <- '289'
hr <- '08'
categories <-
c(”bad_image_blurry”, ”bad_image_malfunction”, ”bad_image_strobe”, ”Calanus”, ”chaetognaths”,
”ctenophores”, ”krill”, ”marine_snow”, ”Other”, ”small_copepod”, ”stick”)
day_hour_files <-  paste0('d', day, '.h', hr)
misclassified <- list.files(day_hour_files, pattern = 'misclassified_', full.names = TRUE)
reclassify <- list.files(day_hour_files, pattern = 'reclassify_', full.names = TRUE)
vpr_autoid_create(reclassify, misclassified, basepath, categories)

## End(Not run)
```

vpr_autoid_read *Read VPR aid files*

Description

Read aid text files containing ROI string information or measurement data and output as a dataframe

Usage

```
vpr_autoid_read(
  file_list_aid,
  file_list_aidmeas,
  export,
  station_of_interest,
  opticalSetting,
  warn = TRUE,
  categories
)
```

Arguments

`file_list_aid` a list object of aid text files, containing ROI strings.

`file_list_aidmeas`
a list object of aidmea text files, containing ROI measurements.

`export` a character string specifying which type of data to output, either 'aid' (roi strings) or 'aidmeas' (measurement data)

`station_of_interest`
Station information to be added to ROI data output, use NA if irrelevant

`opticalSetting` Optional argument specifying VPR optical setting. If provided will be used to convert size data into mm from pixels, if missing size data will be output in pixels

`warn` Logical, FALSE silences size data unit warnings

`categories` A list object (of chr strings) with all the potential classification categories

Details

Only outputs either ROI string information OR measurement data

Note

Full paths to each file should be specified

Author(s)

E. Chisholm & K. Sorochan

Examples

```
station_of_interest <- 'test'
dayhour <- c('d222.h03', 'd222.h04')
categories <- c("bad_image_blurry", "bad_image_malfunction",
"bad_image_strobe", "Calanus", "chaetognaths","ctenophores","krill",
"marine_snow","Other","small_copepod", "stick")

#' #VPR OPTICAL SETTING (S0, S1, S2 OR S3)
opticalSetting <- "S2"
imageVolume <- 83663 #mm^3

auto_id_folder <- system.file('extdata/COR2019002/autoid/', package = 'vprr', mustWork = TRUE)
auto_id_path <- list.files(paste0(auto_id_folder, "/"), full.names = TRUE)

#'   # Path to aid for each category
aid_path <- paste0(auto_id_path, '/aid/')
# Path to mea for each category
aidmea_path <- paste0(auto_id_path, '/aidmea/')

# AUTO ID FILES
aid_file_list <- list()
aidmea_file_list <- list()
```

```
for (i in 1:length(dayhour)) {
  aid_file_list[[i]] <-
    list.files(aid_path, pattern = dayhour[[i]], full.names = TRUE)
  # SIZE DATA FILES
  aidmea_file_list[[i]] <-
    list.files(aidmea_path, pattern = dayhour[[i]], full.names = TRUE)
}

aid_file_list_all <- unlist(aid_file_list)
aidmea_file_list_all <- unlist(aidmea_file_list)

 # ROIs
roi_dat_combine <-
  vpr_autoid_read(
    file_list_aid = aid_file_list_all,
    file_list_aidmeas = aidmea_file_list_all,
    export = 'aid',
    station_of_interest = station_of_interest,
    opticalSetting = opticalSetting,
    warn = FALSE,
    categories = categories
  )

# MEASUREMENTS
roimeas_dat_combine <-
  vpr_autoid_read(
    file_list_aid = aid_file_list_all,
    file_list_aidmeas = aidmea_file_list_all,
    export = 'aidmeas',
    station_of_interest = station_of_interest,
    opticalSetting = opticalSetting,
    warn = FALSE,
    categories = categories
 )
```

vpr_category *Get category ids from string*

Description

Get category ids from string

Usage

```
vpr_category(x, categories)
```

Arguments

x	A chr string which represents file paths from which category should be extracted
categories	A list object with all the potential classification categories

Value

A chr string of only the category id

Note

This function searches for exact matches to categories within '/' file separators. You may encounter errors if

Author(s)

K Sorochan

See Also

vpr_hour, vpr_day, vpr_roi

Examples

```
category_string <- 'C:/data/cruise/autoid/Calanus/d000/h00'
categories <- list("Calanus", "marine_snow", "blurry", "other_copepod")
vpr_category(category_string, categories)
```

---

vpr_category_create *Create a new category to be considered for classification after processing with VP*

Description

creates empty directory structure to allow consideration of new category during vpr_manual_classification()

Usage

```
vpr_category_create(category, basepath)
```

Arguments

category	new category name to be added (can be a list of multiple category names)
basepath	path to folder containing autoid files (e.g., 'extdata/COR2019002/autoid')

Value

empty directory structure using new category name inside basepath

vpr_ctdroisize_merge *Format CTD and Size data from VPR*

Description

Format CTD and Meas data frames into combined data frame for analysis and plotting of size data

Usage

```
vpr_ctdroisize_merge(data, data_mea, category_of_interest)
```

Arguments

`data`	VPR dataframe from `vpr_ctdroi_merge`, with calculated variable sigmaT
`data_mea`	VPR size data frame from `vpr_autoid_read`
`category_of_interest`	a list of category of interest to be included in output dataframe

Value

A dataframe containing VPR CTD and size data

Examples

```
## Not run:
data("ctd_roi_merge")
data("roimeas_dat_combine")
category_of_interest = 'Calanus'

ctd_roi_merge$time_hr <- ctd_roi_merge$time_ms /3.6e+06

size_df_f <- vpr_ctdroisize_merge(ctd_roi_merge, data_mea = roimeas_dat_combine,
 category_of_interest = category_of_interest)

## End(Not run)
```

vpr_ctdroi_merge *Merge CTD and ROI data from VPR*

Description

Combines CTD data (time, hydrographic parameters), with ROI information (identification number) into single dataframe, aligning ROI identification numbers and category classifications with time and hydrographic parameters

Usage

```
vpr_ctdroi_merge(ctd_dat_combine, roi_dat_combine)
```

Arguments

`ctd_dat_combine`
: a CTD dataframe from VPR processing from `vpr_ctd_read`

`roi_dat_combine`
: a data frame of roi aid data from `vpr_autoid_read`

Author(s)

E. Chisholm & K. Sorochan

Examples

```
data('ctd_dat_combine')
data('roi_dat_combine')

ctd_roi_merge <- vpr_ctdroi_merge(ctd_dat_combine, roi_dat_combine)
```

---

vpr_ctd_files *Create a list of ctd files to be read*

Description

Searches through typical VP directory structure

Usage

```
vpr_ctd_files(castdir, cruise, day_hour)
```

Arguments

castdir	root directory for ctd cast files
cruise	cruise name (exactly as in directory structure)
day_hour	vector of day-hour combinations (e.g, dXXX.hXX)

Details

Use with caution

Value

vector of ctd file paths matching days-hour combinations provided

Author(s)

E. Chisholm and K. Sorochan

vpr_ctd_read *Read and format CTD VPR data*

Description

Acts as a wrapper for `ctd_df_cols`

Usage

```
vpr_ctd_read(ctd_files, station_of_interest, day, hour, col_list)
```

Arguments

`ctd_files`	full file paths to vpr ctd `.dat` files
`station_of_interest`	VPR station name
`day`	Day of interest, if not provided will be pulled from file path
`hour`	Hour of interest, if not provided will be pulled from file path
`col_list`	Optional chr vector of CTD data column names

Details

Reads CTD data and adds day, hour, and station information. Calculates sigma T and depth variables from existing CTD data to supplement raw data. If there are multiple hours of CTD data, combines them into single dataframe.

WARNING `ctd_df_cols` is hard coded to accept a specific order of CTD data columns. The names and values in these columns can change based on the specific instrument and should be updated/confirmed before processing data from a new VPR.

Author(s)

E. Chisholm & K. Sorochan

Examples

```
station_of_interest <- 'test'

ctd_files <- system.file("extdata/COR2019002/rois/vpr5/d222", "h03ctd.dat.gz",
package = "vprr", mustWork = TRUE)

ctd_dat_combine <- vpr_ctd_read(ctd_files, station_of_interest)
```

vpr_ctd_ymd *Add Year/ month/ day hour:minute:second information*

Description

Obtain columns for date and time (i.e., column "ymdhms") and time in hours (i.e., column time_hr) for each row in VPR data frame by utilizing day-of-year, hour, and millisecond outputs from VPR data output.

Usage

```
vpr_ctd_ymd(data, year, offset)
```

Arguments

data	VPR data frame from vpr_ctdroi_merge
year	Year of data collection
offset	time offset in hours between VPR CPU and processed data times (optional)

Value

A VPR data frame with columns for date and time (i.e., column 'ymdhms') and hour (i.e., column time_hr)

Examples

```
year <- 2019
data('ctd_roi_merge')
dat <- vpr_ctd_ymd(ctd_roi_merge, year)
```

vpr_day *Get day identifier*

Description

Get day identifier

Usage

```
vpr_day(x)
```

Arguments

x	A string specifying the directory and file name of the size file

Value

A string of only the day identifier (i.e., "dXXX")

Author(s)

K Sorochan

See Also

vpr_hour, vpr_roi, vpr_category

Examples

```
day_string <- 'C:/data/cruise/autoid/Calanus/d000/h00'
vpr_day(day_string)
```

vpr_dayhour *Find day & hour info to match each station of interest for processing*

Description

@author E. Chisholm and K. Sorochan

Usage

```
vpr_dayhour(stations, file)
```

Arguments

stations	a vector of character values naming stations of interest
file	CSV file containing 'day', 'hour', 'station', and 'day_hour' columns

Value

Vector of day-hour combinations corresponding to stations of interest

vpr_export *Format and export VPR data for publication (IN DEVELOPMENT) Exports a csv file with standard column names based on British Oceanographic Data Centre, BODC::P01 and DarwinCore (DwC) naming conventions, and a JSON metadata file for station level metadata*

Description

Format and export VPR data for publication (IN DEVELOPMENT) Exports a csv file with standard column names based on British Oceanographic Data Centre, BODC::P01 and DarwinCore (DwC) naming conventions, and a JSON metadata file for station level metadata

Usage

```
vpr_export(data, metadata, columnNames, file)
```

Arguments

`data` a VPR data frame

`metadata` (optional) a named list of character values giving metadata to be included in JSON file

`columnNames` (optional) a named list of character values giving relationships between existing names of data columns and standard names

`file` a file name for the data.csv

Examples

```
## Not run:
data(category_conc_n)
metadata <- list(
  "station_level" = list(
    "title" = list("en" = "VPR data from the Scotian Shelf",
                   "fr" = "Données VPR de l'étagère néo-écossaise"),
    "dataset_ID" = 1,
    "decimalLatitudeStart" = 44.5,
    "decimalLongitudeStart" = -64.5,
    "decimalLatitudeEnd" = 45.5,
    "decimalLongitudeEnd" = -65.5,
    "maximumDepthInMeters" = 1000,
    "eventDate" = "2019-08-11",
    "eventTime" = "00:00:00",
    "basisOfRecord" = "MachineObservation",
   "associatedMedia" = "https://ecotaxa.obs-vlfr.fr/ipt/archive.do?r=iml2018051",
   "identificationReferences" = "Iv3 model v3.3",
   "instrument" = list("opticalSetting" = "S2",
                       "imageVolume" = 83663),
   "resources" = list(
      "data" = list("name" = "vpr123_station25.csv",
                    "creationDate" = "2023-01-01"),
      "metadata" = list("name" = "vpr123_station25-metadata.json",
                        "creationDate" = "2023-01-01")
    ),
    "dataAttributes" = list(
      "eventID" = list(
        "dataType" = "chr",
        "definition" = "An identifier for the set of information associated
        with a dwc:Event (something that occurs at a place and time). May be
        a global unique identifier or an identifier specific to the data set.",
        "vocabulary" = "dwc"
      ),
      "minimumDepthInMeters" = list(
        "dataType" = "float",
        "definition" = "The lesser depth of a range of depth below the local",
        "vocabulary" = "dwc"
      ),
      "maximumDepthInMeters" = list(
        "dataType" = "float",
        "definition" = "The greater depth of a range of depth below the local",
        "vocabulary" = "dwc"
      ),
      "DEPHPRST" = list(
        "dataType" = "float",
        "definition" = "Depth (spatial coordinate) of sampling event start
        relative to water surface in the water body by profiling pressure
         sensor and conversion to depth using unspecified algorithm",
        "vocabulary" = "BODC::P01"
      ),
      "individualCount" = list(
        "dataType" = "float",
        "definition" = "The number of individuals present at the time of the
         dwc:Occurrence.",
        "vocabulary" = "dwc"
      ),
      "verbatimIdentification" = list(
        "dataType" = "chr",
       "definition" = "A string representing the taxonomic identification as
       it appeared in the original record.",
        "vocabulary" = "dwc"
      ),
      "SDBIOL01" = list(
        "dataType" = "float",
        "definition" = "Abundance of biological entity specified elsewhere
        per unit volume of the water body",
        "vocabulary" = "BODC::P01"
      ),
      "TEMPST01" = list(
        "dataType" = "float",
        "definition" = "Temperature of the water body by CTD or STD",
        "vocabulary" = "BODC::P01"
      ),
      "PSALST01" = list(
        "dataType" = "float",
        "definition" = "Practical salinity of the water body by CTD and
        computation using UNESCO 1983 algorithm",
        "vocabulary" = "BODC::P01"
      ),
      "POTDENS0" = list(
        "dataType" = "float",
        "definition" = "Density (potential) of the water body by computation
         from salinity and potential temperature using UNESCO algorithm with
          0 decibar reference pressure",
        "vocabulary" = "BODC::P01"
      ),
      "FLUOZZZZ" = list(
        "dataType" = "float",
        "definition" = "Fluorescence of the water body",
        "vocabulary" = "BODC::P01"
      ),
      "TURBXXXX" = list(
        "dataType" = "float",
        "definition" = "Turbidity of water in the water body",
       "vocabulary" = "BODC::P01"
     ),
      "sampleSizeValue" = list(
        "dataType" = "float",
        "definition" = "A numeric value for a measurement of the size (time
        duration, length, area, or volume) of a sample in a sampling
        dwc:Event.",
        "vocabulary" = "dwc"
      ),
      "sampleSizeUnit" = list(
        "dataType" = "chr",
        "definition" = "The unit of measurement of the size (time duration,
        length, area, or volume) of a sample in a sampling dwc:Event.",
       "vocabulary" = "dwc"
      ),
      "scientificName" = list(
        "dataType" = "chr",
        "definition" = "The full scientific name, with authorship and date
        information if known. When forming part of a dwc:Identification, this
         should be the name in lowest level taxonomic rank that can be
         determined. This term should not contain identification
         qualifications, which should instead be supplied in the
         dwc:identificationQualifier term.",
        "vocabulary" = "dwc"
      ),
      "identifiedBy" = list(
        "dataType" = "chr",
        "definition" = "A list (concatenated and separated) of names of
        people, groups, or organisations who assigned the Taxon to the subject.",
        "vocabulary" = "dwc"
      ),
      "identificationVerificationStatus" = list(
        "dataType" = "chr",
        "definition" = "A categorical indicator of the extent to which the
        taxonomic identification has been verified to be correct.",
        "vocabulary" = "dwc"
      ),
      "depthDifferenceMeters" = list(
       "dataType" = "float",
       "definition" = "Difference between maximumDepthInMeters and
       minimumDepthInMeters of an individual data bin, in meters",
        "vocabulary" = "BIO"
      ),
      "minimumTimeSeconds" = list(
        "dataType" = "float",
        "definition" = "minimum time value in a data bin, measured in seconds
         from the start of the day of sampling",
        "vocabulary" = "BIO"
      ),
      "maximumTimeSeconds" = list(
        "dataType" = "float",
        "definition" = "maximum time value in a data bin, measured in seconds
         from the start of the day of sampling",
        "vocabulary" = "BIO"
      ),
      "timeDifferenceSeconds" = list(
        "dataType" = "float",
        "definition" = "Difference between maximumTimeSeconds and
        minimumTimeSeconds of an individual data bin, in seconds",
        "vocabulary" = "BIO"
      ),
      "numberOfFrames" = list(
        "dataType" = "float",
        "definition" = "number of VPR frames captured within an individual data bin",
        "vocabulary" = "BIO"
      ),
      "timeMilliseconds" = list(
        "dataType" = "float",
       "definition" = "Time measured in milliseconds since the start of the sampling day",
        "vocabulary" = "BIO"
      ),
      "towyoID" = list(
        "dataType" = "chr",
        "definition" = "A string identifying the section of the cast to which
         the data point belongs",
        "vocabulary" = "BIO"
      ),
      "maximumCastDepthInMeters" = list(
        "dataType" = "float",
        "definition" = "Maximum depth in Meters of the cast dataset",
        "vocabulary" = "BIO"
      )
    )
  )
```

```
)

# new_name = old_name
columnNames = list( "DEPHPRST" = "depth" ,
                    "verbatimIdentification" = "category",
                    "eventID" = "station",
                   "minimumDepthInMeters" = "min_depth",
                    "maximumDepthInMeters" = "max_depth",
                    "individualCount" = "n_roi_bin",
                    "SDBIOL01" = "conc_m3",
                    "TEMPST01" = "temperature",
                    "PSALST01" = "salinity",
                    "POTDENS0" = "density",
                    "FLUOZZZZ" = "fluorescence",
                    "TURBXXXX" = "turbidity",
                    "sampleSizeValue" = "vol_sampled_bin_m3",
                    "depthDifferenceMeters" = "depth_diff",
                    "minimumTimeSeconds" = "min_time_s",
                    "maximumTimeSeconds" = "max_time_s",
                    "timeDifferenceSeconds" = "time_diff_s",
                    "numberOfFrames" = "n_frames",
                    "timeMilliseconds" = "time_ms",
                    "towyoID" = "towyo",
                    "maximumCastDepthInMeters" = "max_cast_depth"
)

# add any new data columns required
# (eg. sampleSizeUnit, scientificName, identifiedBy, identificationVerificationStatus)
sampleSizeUnit <- "cubic metre"
identifiedBy <- "K. Sorochan"
identificationVerificationStatus <- "ValidatedByHuman"

data <- category_conc_n %>%
  mutate(., identifiedBy = identifiedBy,
         sampleSizeUnit = sampleSizeUnit,
         identificationVerificationStatus = identificationVerificationStatus)

# Define the mapping between category and scientific name
# scientific names based ecotaxa taxonomic system
scientificName <- list("blurry" = "bad_image_blurry",
                      "artefact" = c("bad_image_malfunction", "bad_image_strobe"),
                      "Calanus" = "Calanus")

# Create a new column of data called scientificName based on matches to category
data <- data %>%
  dplyr::mutate(., scientificName = case_when(
    category %in% scientificName[["blurry"]] ~ "blurry",
    category %in% scientificName[["artefact"]] ~ "artefact",
    category == scientificName[["Calanus"]] ~ "Calanus",
    TRUE ~ NA
  ))

vpr_export(data, metadata, columnNames, file = "vpr123_station25")
```

```
## End(Not run)
```

vpr_hour *Get hour identifier*

Description

Get hour identifier

Usage

```
vpr_hour(x)
```

Arguments

x	A string specifying the directory and file name of the size file

Value

A string of only the hour identifier (i.e., "hXX")

Author(s)

K Sorochan

See Also

vpr_day, vpr_roi, vpr_category

Examples

```
hour_string <- 'C:/data/cruise/autoid/Calanus/d000/h00'
vpr_hour(hour_string)
```

vpr_img_category *Explore images by depth and classification*

Description

Pulls images from specific depth ranges in specific classification group

Usage

```
vpr_img_category(
  data,
  min.depth,
  max.depth,
  roiFolder,
  format = "list",
  category_of_interest
)
```

Arguments

`data` data frame containing CTD and ROI data from `vpr_ctdroi_merge`, which also contains calculated variables sigmaT and time_hr

`min.depth` minimum depth of ROIs you are interested in looking at

`max.depth` maximum depth of ROIs you are interested in exploring

`roiFolder` directory that ROIs are within (can be very general eg. C:/data, but will be quicker to process with more specific file path)

`format` option of how images will be output, either as 'list' a list of file names or 'image' where images will be displayed

`category_of_interest` character string of classification group from which to pull images

vpr_img_check *Remove ROI strings from aid and aidmeas files based on a manually organized folder of images*

Description

Should be used after `vpr_img_copy`, and manual image removal from created folders

Usage

```
vpr_img_check(folder_dir, basepath)
```

Arguments

`folder_dir` directory path to day hour folders containing manually reorganized images of a specific category eg. 'C:/data/cruise_IML2018051/krill/images/' where that folder contains '......d123.h01/' which contains manually sorted images of krill

`basepath` directory path to original Visual Plankton files, specified down to the classification group. eg. 'C:/data/cruise_IML2018051/autoid/krill'

`vpr_img_copy` *Image copying function for specific category of interest*

Description

This function can be used to copy images from a particular category, day and hour into distinct folders within the auto id directory This is useful for visualizing the ROIs of a particular classification group or for performing manual tertiary checks to remove images not matching classification group descriptions.

Usage

```
vpr_img_copy(auto_id_folder, categories.of.interest, day, hour)
```

Arguments

`auto_id_folder` eg "D:/VP_data/IML2018051/autoid"

`categories.of.interest`
eg. categories.of.interest <- c('Calanus')

`day` character, day of interest

`hour` character, hour of interest

`vpr_img_depth` *Explore VPR images by depth bin*

Description

Allows user to pull VPR images from specific depth ranges, to investigate trends before classification of images into category groups

Usage

```
vpr_img_depth(data, min.depth, max.depth, roiFolder, format = "list")
```

Arguments

`data`	data frame containing CTD and ROI data from `vpr_ctdroi_merge`, which also contains calculated variables sigmaT and time_hr
`min.depth`	minimum depth of ROIs you are interested in looking at
`max.depth`	maximum depth of ROIs you are interested in exploring
`roiFolder`	directory that ROIs are within (can be very general eg. C:/data, but will be quicker to process with more specific file path)
`format`	option of how images will be output, either as 'list' a list of file names or 'image' where images will be displayed

`vpr_img_reclassified` *Explore reclassified images*

Description

Pull image from reclassified or misclassified files produced during `vpr_manual_classification`

Usage

```
vpr_img_reclassified(day, hour, base_dir, category_of_interest, image_dir)
```

Arguments

`day`	Character string, 3 digit day of interest of VPR data
`hour`	Character string, 2 digit hour of interest of VPR data
`base_dir`	directory path to folder containing day/hour folders in which misclassified and reclassified files are organized (eg.'C:/VPR_PROJECT/r_project_data_vis/classification files/') which would contain 'd123.h01/reclassified_krill.txt')
`category_of_interest`	Classification group from which to pull images
`image_dir`	directory path to ROI images, eg. "E:\\data\\cruise_IML2018051\\", file separator MUST BE "\\" in order to be recognized

Value

folders of misclassified or reclassified images inside image_dir

vpr_manual_classification *Function to check results of classification manually*

Description

Displays each image in day hour specified, prompts user to confirm or deny classification. If classification is denied, asks for a reclassification value based on available category

Usage

```
vpr_manual_classification(
  day,
  hour,
  basepath,
  category_of_interest,
  gr = TRUE,
  scale = "x300",
  opticalSetting = "S2",
  img_bright = TRUE,
  threshold_score,
  path_score
)
```

Arguments

`day` day of interest in autoid (3 chr)

`hour` hour of interest in autoid (2 chr)

`basepath` path to folder containing autoid files (e.g., 'extdata/COR2019002/autoid')

`category_of_interest` list of category folders you wish you sort through

`gr` logical indicating whether pop up graphic menus are used (user preference - defaults to TRUE)

`scale` argument passed to `image_scale`, default = 'x300'

`opticalSetting` specifies optical setting of VPR, defining image frame size, current options are 'S0', 'S1', 'S2' (default), 'S3', see further info in details

`img_bright` logical value indicating whether or not to include a blown out high brightness version of image (can be helpful for viewing dark field fine appendages)

`threshold_score` (optional) a numeric value defining the minimum confidence value, under which automatic classifications will be passed through manual reclassification. This argument should match the threshold provided in `vpr_autoid_copy()`

`path_score` (optional) file path to the autoid_cnn_scr folder (autoid files with confidence values produced by automated classification)

Details

Optical Setting frame sizes: S0 = 7x7 mm, S1 = 14x14mm, S2 = 24x24mm, S3 = 48x48 mm. These settings define the conversion factor from pixels to millimetres and calculate image size for classification reference

Development

- Add "undo" functionality to go back on a typing mistake
- Fix scaling/ size issue so images are consistently sized

`vpr_oce_create` *Create ctd oce object with vpr data*

Description

Formats VPR data frame into oce format CTD object

Usage

```
vpr_oce_create(data)
```

Arguments

`data`	data frame of vpr data

Author(s)

E. Chisholm

Examples

```
data('ctd_roi_merge')
oce_dat <- vpr_oce_create(ctd_roi_merge)
```

vpr_plot_contour *Interpolated contour plot of particular variable*

Description

Creates interpolated contour plot, can be used as a background for ROI or tow yo information

Usage

```
vpr_plot_contour(
  data,
  var,
  dup = "mean",
  method = "interp",
  labels = TRUE,
  bw = 1,
  cmo
)
```

Arguments

data	data frame needs to include time_hr, depth, and variable of choice (var)
var	variable in dataframe which will be interpolated and plotted
dup	if method == 'interp'. Method of handling duplicates in interpolation, passed to interp function (options: 'mean', 'strip', 'error')
method	Specifies interpolation method, options are 'interp' or 'oce', oce uses slightly different method (oce is least error prone)
labels	logical value indicating whether or not to plot contour labels
bw	bin width defining interval at which contours are labelled
cmo	name of a cmocean plotting theme, see ?cmocean for more information

Author(s)

E. Chisholm & Kevin Sorochan

vpr_plot_profile *Plots VPR profiles of temperature, salinity, density, fluorescence and concentration (by classification group)*

Description

This plot allows a good overview of vertical distribution of individual classification groups along with reference to hydrographic parameters. Facet wrap is used to create distinct panels for each category provided

Usage

```
vpr_plot_profile(category_conc_n, category_to_plot, plot_conc)
```

Arguments

`category_conc_n`
: A VPR data frame with hydrographic and concentration data separated by category (from `vpr_roi_concentration`)

`category_to_plot`
: The specific classification groups which will be plotted, if NULL, will plot all category combined

`plot_conc`
: Logical value whether or not to include a concentration plot (FALSE just shows CTD data)

Value

A gridded object of at least 3 ggplot objects

`vpr_plot_TS` *Make a balloon plot against a TS plot*

Description

TS balloon plot with ROI concentration, sorted by category includes isopycnal line calculations

Usage

```
vpr_plot_TS(x, reference.p = 0, var)
```

Arguments

`x`	dataframe with temperature, salinity, number of rois (n_roi_bin)
`reference.p`	reference pressure (default at 0 for surface)- used to calculate isopycnals
`var`	variable on which size of points will be based, eg conc_m3 or n_roi_bin

Note

modified from source: `https://github.com/Davidatlarge/ggTS/blob/master/ggTS_DK.R`

Author(s)

E. Chisholm

vpr_plot_TScat *Make a balloon plot*

Description

Balloon plot against a TS plot with ROI concentration and sorted by category includes isopycnal line calculations. Version of `vpr_plot_TS`, with only relevant* category specified. *to current analysis and research objectives (See note).

Usage

```
vpr_plot_TScat(x, reference.p = 0)
```

Arguments

`x`	dataframe with temperature, salinity, number of rois named by category
`reference.p`	reference pressure (default at 0 for surface)- used to calculate isopycnals

Note

WARNING HARD CODED FOR 5 category, CALANUS, KRILL, ECHINODERM LARVAE, SMALL COPEPOD, CHAETOGNATHS !! Uses isopycnal labelling method which does not label every contour

modified from source: `https://github.com/Davidatlarge/ggTS/blob/master/ggTS_DK.R`

vpr_pred_read *Read prediction output from a CNN model*

Description

Read prediction output from a CNN model

Usage

```
vpr_pred_read(filename)
```

Arguments

`filename`	model prediction output file (.txt) from `vpr_transferlearn::save_output()`

Value

a dataframe

vpr_roi *Get roi ids from string*

Description

Get roi ids from string

Usage

```
vpr_roi(x)
```

Arguments

x	A string specifying directory and file name of roi

Value

A string of only the 10 digit roi identifier

Author(s)

K Sorochan

See Also

vpr_hour, vpr_day, vpr_category

Examples

```
roi_string <- 'roi.0100000000.tif'
vpr_roi(roi_string)
```

vpr_roi_concentration *Calculate VPR concentrations*

Description

Calculates concentrations for each named category in dataframe

Usage

```
vpr_roi_concentration(
  data,
  category_list,
  station_of_interest,
  binSize,
  imageVolume,
  rev = FALSE
)
```

Arguments

`data` a VPR dataframe as produced by `vpr_ctdroi_merge`

`category_list` a vector of character strings representing category present in the station being processed

`station_of_interest` The station being processed

`binSize` passed to `bin_calculate`, determines size of depth bins over which data is averaged

`imageVolume` the volume of VPR images used for calculating concentrations (mm^3)

`rev` Logical value defining direction of binning, FALSE (default) - bins will be calculated from surface to bottom, TRUE- bins will be calculated bottom to surface

Examples

```
data('ctd_roi_merge')
ctd_roi_merge$time_hr <- ctd_roi_merge$time_ms /3.6e+06

category_list <- c('Calanus', 'krill')
binSize <- 5
station_of_interest <- 'test'
imageVolume <- 83663

category_conc_n <- vpr_roi_concentration(ctd_roi_merge, category_list,
station_of_interest, binSize, imageVolume)
```

vpr_save *Save VPR data as an as.oce object*

Description

Save VPR data as an as.oce object

Usage

```
vpr_save(data, metadata)
```

Arguments

`data` a VPR data frame

`metadata` (optional) a named list of character values giving metadata values. If this argument is not provided user will be prompted for a few generic metadata requirements.

Details

This function will pass a VPR data frame to an `oce` object. Using an `oce` object as the default export format for VPR data allows for metadata and data to be kept in the same, space efficient file, and avoid redundancy in the data frame. The function checks for data parameters that may actually be metadata parameters (rows which have the same value repeated for every observation). These parameters will automatically be copied into the metadata slot of the `oce` object. The function will also prompt for a variety of required metadata fields. Depending on specific research / archiving requirements, these metadata parameters could be updated by providing the argument `metadata`.

Default metadata parameters include ’deploymentType’, ’waterDepth’, ’serialNumber’, ’latitudeStart’, ’longitudeStart’, ’castDate’, ’castStartTime’, ’castEndTime’, ’processedBy’, ’opticalSetting’, ’imageVolume’, ’comment’.

Value

an oce CTD object with all VPR data as well as metadata

Examples

```
data(”category_conc_n”)
metadata <- list('deploymentType' = 'towyo', 'waterDepth' =
max(ctd_roi_merge$pressure), 'serialNumber' = NA, 'latitudeStart' = 47,
'longitudeStart' = -65, 'castDate' = '2019-08-11', 'castStartTime'= '00:00',
'castEndTime' = '01:00', 'processedBy' = 'E. Chisholm', 'opticalSetting' =
'S2', 'imageVolume' = 83663, 'comment' = 'test data')

oce_dat <- vpr_save(category_conc_n, metadata)
# save(oce_dat, file = vpr_save.RData') # save data
```

`vpr_size_bin` *Bin VPR size data*

Description

Calculates statistics for VPR measurement data in depth averaged bins for analysis and visualization

Usage

```
vpr_size_bin(data_all, bin_mea)
```

Arguments

data_all a VPR CTD and measurement dataframe from vpr_ctdroisize_merge

bin_mea Numerical value representing size of depth bins over which data will be combined, unit is metres, typical values range from 1 - 5

Value

a dataframe of binned VPR size data statistics including number of observations, median, interquartile ranges, salinity and pressure, useful for making boxplots

Examples

```
## Not run:
data('size_df_f')
vpr_size_bin(size_df_f, bin_mea = 5)

## End(Not run)
```

---

vpr_trrois_size *Get size data from idsize files*

Description

useful for getting size distribution of known rois from each category. gathers size information from idsize text files produced when training a new classifier in VP (Visual Plankton)

Usage

```
vpr_trrois_size(directory, category, opticalSetting)
```

Arguments

directory cruise directory eg. ’C:/data/IML2018051/’

category list of character elements containing category of interest

opticalSetting VPR optical setting determining conversion between pixels and millimetres (options are ’S0’, ’S1’, ’S2’, or ’S3’)

Index