Package 'BioThermR'

January 21, 2026

Type Package

Title Standardized Processing and Analysis of Thermal Imaging Data in Animal Studies

Version 0.1.0

Description A modular framework for standardized analysis of thermal imaging data in animal experimentation. The package integrates thermographic data import (FLIR, raw, CSV), automated region of interest (ROI) segmentation based on 'EBImage' (Pau et al., 2010 <doi:10.1093/bioinformatics/btq046>), interactive ROI refinement, and high-throughput batch processing.

URL https://github.com/RightSZ/BioThermR,

https://rightsz.github.io/BioThermR/

BugReports https://github.com/RightSZ/BioThermR/issues

Depends R (>= 4.1.0)

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.3

Imports EBImage, ggplot2, ggrepel, ggsci, plotly, shiny, stats, Thermimage

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

NeedsCompilation no

Author BeiHao Li [aut, cre]

Maintainer BeiHao Li <szright2000@gmail.com>

Repository CRAN

Date/Publication 2026-01-21 19:40:02 UTC

Contents

`aggregate_replicates` *Aggregate Technical Replicates for Statistical Rigor*

Description

Collapses technical replicates (e.g., multiple thermal images taken of the same animal) into a single biological data point per subject. This step is crucial for avoiding pseudoreplication in downstream statistical analyses (e.g., t-tests, ANOVA).

Usage

```
aggregate_replicates(data, id_col, method = "mean", keep_cols = NULL)
```

Arguments

`data`	A data frame. Typically the output from `compile_batch_stats` or `merge_clinical_data`.
`id_col`	String. The column name representing the unique Biological Subject ID (e.g., "MouseID", "Subject_No"). Rows sharing this ID will be condensed into one.

method	String. The mathematical function used for aggregation: either "mean" (default) or "median". Median is often more robust to outliers (e.g., one blurry image).
keep_cols	Vector of strings. Names of non-numeric metadata columns to preserve in the final output (e.g., "Group", "Genotype", "Sex", "Treatment").

Value

A data frame with exactly one row per unique ID. The column order is reorganized to place ID and metadata first, followed by the aggregated thermal statistics and the n_replicates count.

Examples

```
# Create a toy dataset with repeated measurements
df_raw <- data.frame(
  SampleID = rep(paste0("M", 1:3), each = 3),
  Group = rep(c("ND", "HFD", "ND"), each = 3),
  Sex = rep("M", 9),
  Median = runif(9, 33, 36),
  IQR = runif(9, 0.5, 1.5)
)

df <- aggregate_replicates(
  data = df_raw,
  id_col = "SampleID",
  method = "median",
  keep_cols = c("Group", "Sex")
)
```

---

analyze_thermal_stats *Calculate Comprehensive Thermal Statistics*

Description

Computes a detailed set of summary statistics from the thermal matrix. Metrics include central tendency (Mean, Median, Peak Density), dispersion (SD, IQR, CV), and range (Min, Max, Quantiles). NA values (background) are automatically excluded.

Usage

```
analyze_thermal_stats(img_obj, use_processed = TRUE)
```

Arguments

img_obj	A 'BioThermR' object.
use_processed	Logical. If TRUE (default), calculates statistics on the 'processed' matrix (where background is likely masked as NA). If FALSE, uses the 'raw' matrix.

Details

The function calculates the following metrics:

- **Min/Max:** Extremities of the temperature distribution.
- **Mean/Median:** Measures of central tendency.
- **SD (Standard Deviation):** Absolute measure of spread.
- **IQR (Interquartile Range):** Robust measure of spread (Q75 - Q25).
- **CV (Coefficient of Variation):** Relative measure of spread (SD / Mean), useful for assessing thermal heterogeneity.
- **Peak_Density:** The temperature value corresponding to the peak of the kernel density estimate (Mode).

Value

A 'BioThermR' object with the `stats` slot updated containing a data frame of results.

Examples

```
img_obj <- system.file("extdata", "C05.raw", package = "BioThermR")
img <- read_thermal_raw(img_obj)
img <- analyze_thermal_stats(img)
```

---

as_EBImage *Convert BioThermR Object to EBImage Object*

Description

Extracts the thermal matrix (raw or processed) from a 'BioThermR' object and converts it into an 'EBImage' class object. This conversion is essential for applying advanced morphological operations (e.g., thresholding, watershed, labeling) provided by the 'EBImage' package.

Usage

```
as_EBImage(img_obj, use_processed = TRUE, replace_na = 0, normalize = FALSE)
```

Arguments

`img_obj`	A 'BioThermR' object.
`use_processed`	Logical. If `TRUE` (default), uses the 'processed' matrix (which might already have masks applied). If `FALSE`, uses the 'raw' temperature matrix.
`replace_na`	Numeric. The value to replace `NA`s with, as 'EBImage' does not support missing values. Default is 0 (typically treated as background).
`normalize`	Logical. If `TRUE`, scales the matrix values linearly to the range [0, 1]. This is highly recommended for visualization or standard thresholding algorithms (like Otsu) in 'EBImage'. Default is `FALSE` (preserves actual temperature values).

Value

An Image object (defined in the 'EBImage' package).

Examples

```
# Load data
mat <- matrix(runif(160*120, 20, 40), nrow = 120, ncol = 160)
obj <- create_BioThermR(mat, name = "Simulation_01")

# Convert to EBImage format with normalization (for thresholding)
eb_norm <- as_EBImage(obj, normalize = TRUE)

# Convert preserving temperature values (for calculation)
eb_temp <- as_EBImage(obj, normalize = FALSE)
```

---

compile_batch_stats *Compile Batch Statistics into a Tidy Data Frame*

Description

Iterates through a list of 'BioThermR' objects, extracts the pre-calculated statistics (from analyze_thermal_stats), and aggregates them into a single summary data frame. This function transforms the nested list structure into a flat, tabular format (Tidy Data) suitable for downstream statistical analysis (ANOVA, t-test) or visualization.

Usage

```
compile_batch_stats(img_list)
```

Arguments

img_list	A list of 'BioThermR' objects (typically the output of a batch processing workflow). Note: analyze_thermal_stats must be run on these objects first to populate the 'stats' slot.

Value

A data.frame where:

- **Rows** represent individual images.
- **Columns** include 'Filename' and all metrics computed by analyze_thermal_stats (e.g., Min, Max, Mean, Median, SD, IQR, CV, Peak_Density).

Examples

```
# 1. Import and Process
img_obj_list <- system.file("extdata",package = "BioThermR")
img_list <- read_thermal_batch(img_obj_list)
img_list <- lapply(img_list, analyze_thermal_stats)

# 2. Compile Results
df_results <- compile_batch_stats(img_list)
```

`create_BioThermR` *Create BioThermR Object from Data*

Description

Manually creates a BioThermR object from a numeric matrix or data frame. Useful for converting data loaded from other formats (Excel, CSV, txt) or simulation data.

Usage

```
create_BioThermR(data, name = "Sample")
```

Arguments

`data`	A numeric matrix or data frame representing the thermal image grid (rows x cols).
`name`	String. An identifier for this sample (e.g., filename or sample ID). Default is "Sample".

Value

A 'BioThermR' object.

Examples

```
mat <- matrix(runif(160*120, 20, 40), nrow = 120, ncol = 160)
obj <- create_BioThermR(mat, name = "Simulation_01")
plot_thermal_heatmap(obj)
```

from_EBImage *Import EBImage Object into BioThermR*

Description

Converts an 'EBImage' class object back into the 'BioThermR' framework. This function is designed to integrate results from external morphological operations (e.g., watershed segmentation, filtering) back into the standard BioThermR analytical workflow.

Usage

```
from_EBImage(
  eb_img,
  template_obj = NULL,
  name = "Imported_EBImage",
  mask_zero = FALSE
)
```

Arguments

eb_img	An Image object (from the 'EBImage' package).
template_obj	Optional. An existing 'BioThermR' object to update. Providing this ensures that original metadata (like filenames and original raw data) is retained. Default is NULL.
name	String. The name assigned to the new object (used only if template_obj is NULL). Default is "Imported_EBImage".
mask_zero	Logical. If TRUE, converts values of 0 in the imported matrix to NA. This is particularly useful when importing binary masks where 0 represents the background, as BioThermR uses NA to exclude background pixels from statistical calculations. Default is FALSE.

Details

This function supports two modes:

- **Update Mode:** If template_obj is provided, the input image replaces the processed matrix of the template. Metadata (filename, path) is preserved. Statistics are reset to NULL as the data has changed.
- **Create Mode:** If template_obj is NULL, a new 'BioThermR' object is created from scratch using the input matrix.

If the input eb_img has more than 2 dimensions (e.g., color/RGB), only the first channel is utilized, and a warning is issued.

Value

A 'BioThermR' object with the imported matrix stored in the processed slot.

Examples

```
# Load data
mat <- matrix(runif(160*120, 20, 40), nrow = 120, ncol = 160)
obj <- create_BioThermR(mat, name = "Simulation_01")

# Convert to EBImage format with normalization
eb_obj <- as_EBImage(obj, normalize = TRUE)

# Convert to BioThermR from EBImage
new_obj <- from_EBImage(eb_obj)
```

---

load_biothermr *Load BioThermR Data from Disk*

Description

Restores a previously saved 'BioThermR' object or a list of objects from a .rds file. This function is the counterpart to save_biothermr and allows you to resume analysis from a saved checkpoint.

Usage

```
load_biothermr(file_path)
```

Arguments

file_path	String. The full path to the .rds file (e.g., "results/experiment_data.rds").

Details

Upon loading, the function performs an automatic validation check to ensure the file contains a valid 'BioThermR' class instance (or a list of them). It provides feedback to the console regarding the type and quantity of objects loaded.

Value

A single 'BioThermR' object or a list of 'BioThermR' objects, depending on the structure of the saved data.

See Also

save_biothermr

merge_clinical_data *Merge Thermal Stats with Clinical Data*

Description

Merges the compiled thermal statistics with an external clinical dataset based on filenames/IDs. Automatically handles column name conflicts (removes .x/.y suffixes).

Usage

```
merge_clinical_data(
  thermal_df,
  clinical_df,
  thermal_id = "Filename",
  clinical_id = "Filename",
  clean_ids = TRUE
)
```

Arguments

thermal_df	Data frame. The output from compile_batch_stats().
clinical_df	Data frame. Your external clinical data.
thermal_id	String. Column name in thermal_df to merge on. Default is "Filename".
clinical_id	String. Column name in clinical_df to merge on. Default is "Filename".
clean_ids	Logical. If TRUE, automatically removes file paths and extensions for matching. Default is TRUE.

Value

A merged data frame.

plot_roi_overlap *Visualize ROI Overlap and Dice Coefficient*

Description

Visually compares two segmentation masks (ROIs) by overlaying them on the original thermal image. This function is primarily used for validation purposes: to compare an automated segmentation (filled layer) against a manual ground truth (contour line). It automatically calculates and displays the **Dice Similarity Coefficient (DSC)** in the title.

Usage

```
plot_roi_overlap(
  img_obj1,
  img_obj2,
  title = NULL,
  color = "green",
  alpha = 0.5,
  line_color = "white",
  palette = "inferno"
)
```

Arguments

`img_obj1`	A 'BioThermR' object. Typically the **Automated/Predicted** segmentation. This object must contain the raw thermal matrix. Its mask will be plotted as a filled area.
`img_obj2`	A 'BioThermR' object. Typically the **Manual/Ground Truth** segmentation. Its mask will be plotted as a contour outline. Dimensions must match `img_obj1`.
`title`	String. Custom title for the plot. If `NULL` (default), the title shows "ROI overlap (DICE: X.XXX)".
`color`	String. Fill color for the `img_obj1` mask. Default is "green".
`alpha`	Numeric. Transparency level for the `img_obj1` mask (0 to 1). Default is 0.5.
`line_color`	String. Line color for the `img_obj2` contour. Default is "white".
`palette`	String. Color palette for the background thermal image (passed to `scale_fill_viridis_c`). Default is "inferno".

Details

The visualization consists of three layers:

1. **Background:** The raw thermal image from `img_obj1`.
2. **Prediction (img_obj1):** The processed mask from the first object, rendered as a semi-transparent filled raster (default green).
3. **Ground Truth (img_obj2):** The processed mask from the second object, rendered as a contour outline (default white).

The function calculates the Dice Similarity Coefficient (DSC) using the formula:

$$DSC = \frac{2 \times |X \cap Y|}{|X| + |Y|}$$

where X and Y are the set of pixels in the two masks. A DSC of 1 indicates perfect overlap.

Value

A `ggplot` object showing the overlay.

Examples

```
#' # Load raw data
img_obj <- system.file("extdata", "C05.raw", package = "BioThermR")
img <- read_thermal_raw(img_obj)
# Apply automated segmentation
img1 <- roi_segment_ebimage(img, keep_largest = TRUE)

# Simple background removal: Keep everything above 24 degrees
img2 <- roi_filter_threshold(img, threshold = c(33, Inf))

# Compare them
plot_roi_overlap(img_obj1 = img1,
                 img_obj2 = img2)
```

---

plot_thermal_3d *Interactive 3D Thermal Surface Plot*

Description

Generates a rotatable, interactive 3D surface plot using the 'plotly' engine. This visualization maps temperature to the Z-axis, allowing users to intuitively explore the thermal topology, gradients, and intensity of hotspots.

Usage

```
plot_thermal_3d(img_obj, use_processed = TRUE)
```

Arguments

`img_obj`	A 'BioThermR' object.
`use_processed`	Logical. If `TRUE` (default), uses the 'processed' matrix (where background is likely `NA`). If `FALSE`, uses the 'raw' sensor data.

Details

3D visualization is particularly powerful for:

- **Quality Control:** Quickly identifying noise spikes or "cold" artifacts that flat heatmaps might hide.
- **Gradient Analysis:** Visualizing how heat dissipates from a central source (e.g., a tumor or inflammation site).
- **Presentation:** Creating engaging, interactive figures for HTML reports or Shiny dashboards.

The output is an HTML widget that allows zooming, panning, and hovering to see specific pixel values.

Value

A `plotly` object (HTML widget).

Examples

```
# Load raw data
img_obj <- system.file("extdata", "C05.raw", package = "BioThermR")
img <- read_thermal_raw(img_obj)

# Apply automated segmentation
img <- roi_segment_ebimage(img, keep_largest = TRUE)

# 3d plot
plot_thermal_3d(img)
```

---

plot_thermal_cloud *Generate a "Thermal Cloud" Visualization (Phyllotaxis Layout)*

Description

Arranges a collection of segmented thermal objects into an organic, spiral "cloud" formation. Unlike a rigid grid, this layout uses a golden-angle spiral (phyllotaxis) algorithm to cluster subjects efficiently. This is particularly effective for visualizing the diversity of thermal phenotypes in large datasets or creating artistic figures for presentations and covers.

Usage

```
plot_thermal_cloud(
  img_list,
  spread_factor = 1.1,
  jitter_factor = 0.5,
  palette = "inferno",
  text_color = "black",
  text_size = 3,
  show_labels = TRUE
)
```

Arguments

`img_list`	A list of 'BioThermR' objects. For best results, these should be pre-processed (e.g., background removed via `roi_filter_threshold`).
`spread_factor`	Numeric. Multiplier for the distance between objects. Values > 1.0 increase spacing (airier cloud), values < 1.0 pack objects tighter. Default is 1.1.
`jitter_factor`	Numeric. Introduces random noise to the placement coordinates to break perfect symmetry and create a more natural look. Default is 0.5.
`palette`	String. The color palette from the 'viridis' package. Default is "inferno".

text_color	String. Color of the filename labels. Default is "black".
text_size	Integer. Font size for the labels. Default is 3.
show_labels	Logical. If TRUE, displays the filename below each object. Default is TRUE.

Details

The function performs object-centric rendering:

1. **Extraction:** For each image, it extracts only the valid foreground pixels (non-NA), ignoring the original frame dimensions.
2. **Re-centering:** Each object is mathematically centered at (0,0) relative to its own coordinate system.
3. **Placement:** Objects are placed along a spiral path defined by the Golden Angle (~137.5 degrees).

The spacing and randomness of the spiral can be tuned using spread_factor and jitter_factor.

Value

A ggplot object with a white background and void theme.

Examples

```
# Load a batch of images
img_obj_list <- system.file("extdata",package = "BioThermR")
batch <- read_thermal_batch(img_obj_list)
batch <- lapply(batch, roi_segment_ebimage)

# Create an artistic thermal cloud
p_cloud <- plot_thermal_cloud(batch, spread_factor = 1.5, jitter_factor = 2.0)
```

---

plot_thermal_density *Visualize Temperature Distribution (Density Plot)*

Description

Generates a probability density plot of the temperature values within the image. This visualization is critical for assessing the homogeneity of the subject's temperature and identifying potential artifacts (e.g., bimodal distributions often indicate poor background removal).

Usage

```
plot_thermal_density(
  img_obj,
  use_processed = TRUE,
  show_peak = TRUE,
  show_max = TRUE,
  show_min = TRUE,
  digits = 2,
  color = "skyblue",
  point_size = 2,
  point_color = "red",
  point_label_color = "black",
  point_label_size = 2
)
```

Arguments

img_obj	A 'BioThermR' object.
use_processed	Logical. If TRUE (default), uses the 'processed' matrix (masked data). If FALSE, uses the 'raw' matrix.
show_peak	Logical. If TRUE, highlights and labels the peak density value (Mode). Default is TRUE.
show_max	Logical. If TRUE, highlights and labels the maximum temperature value. Default is TRUE.
show_min	Logical. If TRUE, highlights and labels the minimum temperature value. Default is TRUE.
digits	Integer. Number of decimal places to round the labels to. Default is 2.
color	String. Fill color for the density area curve. Default is "skyblue".
point_size	Numeric. Size of the points marking Peak/Min/Max. Default is 2.
point_color	String. Color of the points marking Peak/Min/Max. Default is "red".
point_label_color	String. Color of the text labels. Default is "black".
point_label_size	Numeric. Size of the text labels. Default is 2.

Details

The function computes the kernel density estimate of the valid pixels (ignoring NAs). It can optionally annotate key statistical landmarks:

- **Peak:** The mode of the distribution (most frequent temperature).
- **Max/Min:** The hottest and coldest points in the ROI.

Text labels are automatically repelled using 'ggrepel' to ensure they do not overlap.

Value

A ggplot object. Layers can be added subsequently.

Examples

```
# Load raw data
img_obj <- system.file("extdata", "C05.raw", package = "BioThermR")
img <- read_thermal_raw(img_obj)

# Apply automated segmentation
img <- roi_segment_ebimage(img, keep_largest = TRUE)

# Density plot
plot_thermal_density(img)
```

---

plot_thermal_heatmap *Visualize Thermal Matrix as 2D Heatmap*

Description

Generates a high-quality raster plot of the thermal data using the 'ggplot2' framework. This function allows for quick visualization of raw or processed matrices with customizable perceptually uniform color scales (viridis).

Usage

```
plot_thermal_heatmap(img_obj, use_processed = TRUE, palette = "inferno")
```

Arguments

img_obj	A 'BioThermR' object.
use_processed	Logical. If TRUE (default), plots the 'processed' matrix (showing masking effects). If FALSE, plots the original 'raw' data.
palette	String. The color map option from the 'viridis' package. Options include: "magma", "inferno", "plasma", "viridis", "cividis". Default is "inferno".

Details

The function performs the following steps:

- Converts the thermal matrix into a long-format data frame suitable for ggplot.
- Renders the image using geom_raster.
- Maps temperature to color using the specified 'viridis' palette.
- Ensures the aspect ratio is preserved (coord_fixed) so the image does not appear distorted.
- Sets NA values (masked background) to transparent.

Since the output is a standard ggplot object, layers can be added subsequently (e.g., new titles or annotations).

Value

A ggplot object.

Examples

```
# Load raw data
img_obj <- system.file("extdata", "C05.raw", package = "BioThermR")
img <- read_thermal_raw(img_obj)

# Apply automated segmentation
img <- roi_segment_ebimage(img, keep_largest = TRUE)

# Standard plot
plot_thermal_heatmap(img)
```

---

plot_thermal_montage *Create a Gap-Free Thermal Image Montage*

Description

Combines a list of 'BioThermR' objects into a single, high-resolution grid plot (montage). Unlike standard faceting, this function uses a custom integer-coordinate system to center each subject within a uniform grid cell, ensuring pixel-perfect alignment without the visual artifacts (white gaps) often seen in R raster plots.

Usage

```
plot_thermal_montage(
  img_list,
  ncol = NULL,
  padding = 10,
  palette = "inferno",
  text_color = "black",
  text_size = 4
)
```

Arguments

img_list	A list of 'BioThermR' objects (e.g., output from read_thermal_batch or roi_filter_interactive).
ncol	Integer. The number of columns in the grid. If NULL (default), it is automatically calculated based on the square root of the number of images to create a roughly square layout.
padding	Integer. The size of the whitespace gap (in pixels) between grid cells. Default is 10.
palette	String. The color palette to use (from 'viridis' package). Default is "inferno".
text_color	String. Color of the filename labels. Default is "black".
text_size	Integer. Font size for the filename labels. Default is 4.

Details

The function executes a two-pass algorithm:

1. **Scan Pass:** Iterates through all objects to calculate the bounding box dimensions of the largest subject. This defines the uniform cell size for the grid.
2. **Layout Pass:** Calculates the integer offsets required to center each smaller subject within the master cell. It merges all pixel data into a single master data frame.

The result is rendered using `geom_tile(width=1, height=1)` to guarantee continuous, gap-free visualization.

Value

A `ggplot` object representing the combined montage.

Examples

```
# Load a batch of images
img_obj_list <- system.file("extdata",package = "BioThermR")
batch <- read_thermal_batch(img_obj_list)

# Create a montage with 4 columns
p <- plot_thermal_montage(batch, ncol = 4, padding = 20)
```

`read_thermal_batch` *Batch Read .raw Files*

Description

Scans a folder and imports all matching .raw files into a list.

Usage

```
read_thermal_batch(folder_path, pattern = "\\.raw$", recursive = FALSE, ...)
```

Arguments

`folder_path`	String. Path to the folder.
`pattern`	String. Regex pattern. Default is "\.raw$".
`recursive`	Logical. Default is FALSE.
`...`	Additional arguments passed to `read_thermal_raw`.

Value

A named list of "BioThermR" objects.

Examples

```
# Example using raw thermal files
img_obj_list <- system.file("extdata",package = "BioThermR")
img_list <- read_thermal_batch(img_obj_list)
```

`read_thermal_flir` *Read FLIR Radiometric JPG File*

Description

Reads a FLIR radiometric JPG file, parses the embedded metadata (emissivity, distance, Planck constants, atmospheric parameters), converts the raw sensor data to temperature (Celsius), and constructs a 'BioThermR' object.

Usage

```
read_thermal_flir(file_path, exiftoolpath = "installed")
```

Arguments

`file_path`	String. The full path to the FLIR radiometric .jpg file.
`exiftoolpath`	String. Path to the ExifTool executable. Default is "installed" (assumes it is available in the system PATH).

Details

This function relies on the 'Thermimage' package and the external tool 'ExifTool'. It automatically extracts calibration constants (Planck R1, B, F, O, R2) and environmental parameters to ensure accurate physical temperature conversion. Please ensure 'ExifTool' is installed and added to your system PATH.

Value

A list object of class "BioThermR" containing:

`raw`	The calculated temperature matrix in degrees Celsius.
`processed`	A copy of the raw matrix, intended for subsequent ROI filtering or masking.
`meta`	A list containing metadata: `filename`, `fullpath`, and `dims`.

Examples

```
# Example using a flir thermal file
img_obj <- system.file("extdata", "IR_2412.jpg", package = "Thermimage")
img <- read_thermal_flir(img_obj)
```

`read_thermal_raw` *Read a Single .raw Thermal Image File*

Description

Reads a binary .raw file (typically 32-bit floating point), converts it into a matrix, and constructs a 'BioThermR' object containing raw data and metadata.

Usage

```
read_thermal_raw(file_path, width = 160, height = 120, rotate = TRUE)
```

Arguments

`file_path`	String. The full path to the .raw file.
`width`	Integer. The width of the thermal sensor (number of columns). Default is 160.
`height`	Integer. The height of the thermal sensor (number of rows). Default is 120.
`rotate`	Logical. Whether to rotate the image 90 degrees counter-clockwise. Default is TRUE (corrects orientation for many standard sensor exports).

Value

A list object of class "BioThermR" containing:

`raw`	The original temperature matrix (numeric).
`processed`	A copy of the raw matrix, intended for subsequent ROI filtering or masking.
`meta`	A list containing metadata: `filename`, `fullpath`, and `dims`.

Examples

```
img_obj <- system.file("extdata", "C05.raw", package = "BioThermR")
img <- read_thermal_raw(img_obj)
```

`roi_filter_interactive` *Interactive ROI Selector with Denoising (Shiny App)*

Description

Launches a Shiny GUI application that allows users to interactively refine regions of interest (ROI) for a batch of thermal images. Key features include dynamic temperature thresholding and an automated "Clean Noise" tool that removes artifacts by retaining only the largest connected object (e.g., the mouse).

Usage

```
roi_filter_interactive(img_input, start_index = 1, use_processed = TRUE)
```

Arguments

img_input	A single 'BioThermR' object OR a list of 'BioThermR' objects (e.g., from read_thermal_batch).
start_index	Integer. The index of the image to start viewing. Default is 1. Useful for resuming work on a large batch.
use_processed	Logical. If TRUE (default), the interactive filter is applied to the already 'processed' matrix (e.g., refining an auto-segmented result). If FALSE, it starts from the raw data.

Details

This function opens a local web server (Shiny App). The workflow is as follows:

1. **Navigate:** Use "Prev/Next" buttons to browse the image batch.
2. **Threshold:** Adjust the slider to set the min/max temperature range. Real-time feedback is shown on the plot.
3. **Denoise:** Toggle the "Remove Noise" button. This applies a topological filter (Connected Component Analysis) that identifies the largest contiguous heat source and removes all smaller isolated islands (e.g., urine spots, reflections).
4. **Confirm:** Click "Confirm" to lock in the settings for the current image and auto-advance.
5. **Export:** Click "Finish & Export Data" to close the app and return the processed object list to the R console.

Value

The modified 'BioThermR' object (or list of objects). **Note:** You must assign the result of this function to a variable (e.g., `data <- roi_filter_interactive(data)`) to save the changes.

---

roi_filter_threshold *Filter Thermal Image by Temperature Thresholds*

Description

Masks the thermal image by retaining only pixels that fall within a specified temperature window. Pixels outside this range are set to NA (background).

Usage

```
roi_filter_threshold(img_obj, threshold, use_processed = FALSE)
```

Arguments

`img_obj`	A 'BioThermR' object.
`threshold`	Numeric vector of length 2, e.g., `c(22, 38)`. Defines the inclusive temperature range `[min, max]` to keep. Use `Inf` for open upper bounds (e.g., `c(25, Inf)` keeps everything above 25 degrees Celsius).
`use_processed`	Logical. • If `FALSE` (default): The filter is applied to the **raw** matrix, discarding any previous masks. • If `TRUE`: The filter is applied to the **processed** matrix, allowing for cumulative masking (e.g., refining an Otsu segmentation).

Details

This is the most fundamental segmentation method for thermal images. Since animals are typically warmer than their environment, a simple low-pass filter (e.g., keep > 22 degrees Celsius) is often sufficient to separate the subject from the cage. Open boundaries can be defined using `Inf` or `-Inf`.

Value

A 'BioThermR' object with the `processed` matrix updated.

Examples

```
# Load raw data
img_obj <- system.file("extdata", "C05.raw", package = "BioThermR")
img <- read_thermal_raw(img_obj)

# Simple background removal: Keep everything above 24 degrees
img <- roi_filter_threshold(img, threshold = c(24, Inf))
```

`roi_segment_ebimage` *Automated ROI Segmentation via EBImage*

Description

Performs automated background removal using a hybrid pipeline of global thresholding (Otsu's method), morphological operations, and connected component analysis.

Usage

```
roi_segment_ebimage(
  img_obj,
  method = "otsu",
  keep_largest = TRUE,
  morphology = TRUE
)
```

Arguments

img_obj	A 'BioThermR' object.
method	String. The thresholding algorithm. Currently, only "otsu" is supported.
keep_largest	Logical. If TRUE (default), keeps only the largest connected object and converts all other clusters to background (NA). This is a powerful denoising step for animal experiments.
morphology	Logical. If TRUE (default), applies morphological opening and closing operations to smooth edges and reduce noise.

Details

The segmentation pipeline consists of four steps:

1. **Normalization:** The temperature matrix is scaled to [0, 1] to be compatible with 'EBImage'.
2. **Thresholding:** Otsu's method is used to calculate an optimal global threshold that separates the foreground (subject) from the background based on histogram bimodality.
3. **Morphology:** A disc-shaped brush (size=5) is used for 'Opening' (to remove salt noise) and 'Closing' (to fill small holes inside the subject).
4. **Component Filter:** If keep_largest is TRUE, the function labels all connected regions and retains only the largest one (assuming the animal is the largest heat source), effectively removing smaller artifacts like bedding noise or reflections.

Value

A 'BioThermR' object where the processed matrix has been masked (background pixels are set to NA).

Examples

```
# Load raw data
img_obj <- system.file("extdata", "C05.raw", package = "BioThermR")
img <- read_thermal_raw(img_obj)

# Apply automated segmentation
img <- roi_segment_ebimage(img, keep_largest = TRUE)
```

---

save_biothermr *Save BioThermR Data Objects to Disk*

Description

Serializes and saves a single 'BioThermR' object or a list of objects to a compressed .rds file. This ensures that all components—raw temperature matrices, processed masks, metadata, and calculation stats—are preserved accurately.

Usage

```
save_biothermr(img_input, file_path)
```

Arguments

img_input	A single 'BioThermR' class object or a list of 'BioThermR' objects (e.g., the output from read_thermal_batch).
file_path	String. The destination path (e.g., "results/obj.rds"). If the directory structure does not exist, it will be created automatically.

Value

None (invisible NULL). Prints a success message to the console upon completion.

Examples

```
# Load data
mat <- matrix(runif(160*120, 20, 40), nrow = 120, ncol = 160)
obj <- create_BioThermR(mat, name = "Simulation_01")

# Save a single object to a temporary directory
out_file <- file.path(tempdir(), "mouse_01.rds")
save_biothermr(obj, out_file)
```

---

viz_thermal_barplot *Generate Publication-Ready Comparative Barplots*

Description

Creates a high-quality bar plot to compare thermal metrics across experimental groups. The visualization includes bars representing the mean, customizable error bars (SD or SE), and overlaid individual data points to show biological variation.

Usage

```
viz_thermal_barplot(
  data,
  y_var,
  x_var,
  fill_var = NULL,
  error_bar = "mean_sd",
  add_points = TRUE,
  point_size = 1.5,
  point_alpha = 0.6,
  palette = "npg"
)
```

Arguments

`data`	Data frame. The merged dataset (e.g., output from `aggregate_replicates`).
`y_var`	String. The name of the numeric column to plot (e.g., "Max_Temp", "Mean_Temp").
`x_var`	String. The name of the categorical column for the X-axis groupings (e.g., "Treatment", "Genotype").
`fill_var`	String. The name of the variable used for fill colors. Default is `NULL`, which uses `x_var`.
`error_bar`	String. The type of error bar to display. Options: • `"mean_sd"`: Mean +/- Standard Deviation (shows spread of data). • `"mean_se"`: Mean +/- Standard Error of the Mean (shows precision of the mean).
`add_points`	Logical. If `TRUE` (default), overlays individual data points using `geom_jitter`. This is highly recommended for small sample sizes (n < 20) to maintain transparency.
`point_size`	Numeric. The size of the individual jitter points. Default is 1.5.
`point_alpha`	Numeric. The transparency of the jitter points (0 = transparent, 1 = opaque). Default is 0.6 to handle overlapping points.
`palette`	String or Vector. • If a string: Pre-defined scientific palettes (`"npg"` for Nature Publishing Group, `"jco"` for Journal of Clinical Oncology). • If a character vector: A custom list of hex codes (e.g., `c("#FF0000", "#0000FF")`).

Details

This function is designed to produce figures that are immediately suitable for scientific manuscripts. Key features include:

- **Automatic Statistics:** Calculates Mean and SD/SE internally using `stat_summary`.
- **Smart Coloring:** Supports scientific palettes ("npg", "jco"). If the number of groups exceeds the palette size, it automatically interpolates to generate distinct colors.
- **Layout:** Uses `theme_classic()` and automatically expands the Y-axis limit by 15% to ensure error bars and significance annotations fit comfortably.

Value

A `ggplot` object. Can be further customized with standard ggplot2 functions.

Examples

```
df_bio <- data.frame(
  Treatment = rep(c("ND", "HFD"), each = 5),
  Mean = c(runif(5, 33, 35), runif(5, 34, 36))
)
```

```
# Boxplot with individual points
p <- viz_thermal_barplot(df_bio,y_var="Mean",x_var="Treatment",error_bar = "mean_se")
p
```

`viz_thermal_boxplot` *Generate Publication-Ready Comparative Boxplots*

Description

Creates a high-quality box-and-whisker plot to visualize the distribution of thermal metrics across groups. This function is ideal for displaying median values, quartiles, and range, while optionally overlaying individual data points to reveal the underlying sample distribution.

Usage

```
viz_thermal_boxplot(
  data,
  y_var,
  x_var,
  fill_var = NULL,
  add_points = TRUE,
  point_size = 1.5,
  point_alpha = 0.6,
  palette = "npg"
)
```

Arguments

`data`	Data frame. The merged dataset (e.g., output from `aggregate_replicates`).
`y_var`	String. The name of the numeric column to plot (e.g., "Max", "Mean").
`x_var`	String. The name of the categorical column for the X-axis groupings.
`fill_var`	String. The name of the variable used for fill colors. Default is `NULL`, which uses `x_var`.
`add_points`	Logical. If `TRUE` (default), overlays individual data points using `geom_jitter`. Highly recommended to show sample size and distribution density.
`point_size`	Numeric. The size of the individual jitter points. Default is 1.5.
`point_alpha`	Numeric. The transparency of the jitter points (0 to 1). Default is 0.6.
`palette`	String or Vector. • If a string: Pre-defined scientific palettes (`"npg"`, `"jco"`). • If a character vector: A custom list of hex codes.

Details

This function includes several automated optimizations for scientific reporting:

- **Smart Outlier Handling:** If `add_points` is `TRUE`, the function automatically hides the standard boxplot outliers (`outlier.shape = NA`) to avoid plotting the same data point twice (once as an outlier, once as a jittered point).
- **Palette Expansion:** Like the barplot function, it automatically interpolates colors if the number of experimental groups exceeds the palette's limit.
- **Layout:** Uses `theme_classic()` for a clean, academic look.

Value

A `ggplot` object. Can be further customized with standard ggplot2 functions (e.g., `+ ylim(20, 40)`).

Examples

```
df_bio <- data.frame(
  Treatment = rep(c("ND", "HFD"), each = 5),
  Mean = c(runif(5, 33, 35), runif(5, 34, 36))
)

# Boxplot with individual points
p <- viz_thermal_boxplot(df_bio, y_var = "Mean", x_var = "Treatment")
p
```

`viz_thermal_montage2` *Create a Thermal Image Montage*

Description

Combines multiple BioThermR objects into a single plot (a montage/collage). Images are laid out in a grid, and filenames are labeled above each image. Uses the 'processed' matrix (NA values are transparent).

Usage

```
viz_thermal_montage2(
  img_list,
  ncol = NULL,
  padding = 10,
  palette = "inferno",
  text_color = "white",
  text_size = 4
)
```

Arguments

img_list	A list of 'BioThermR' objects.
ncol	Integer. Number of columns in the grid. If NULL, tries to create a roughly square grid.
padding	Integer. Pixel gap between images. Default is 10.
palette	String. Color palette for temperature. Default is "inferno".
text_color	String. Color of the filename labels. Default is "white" (good contrast on dark backgrounds).
text_size	Integer. Font size of the labels. Default is 4.

Value

A ggplot object.

Examples

```
# Load a batch of images
img_obj_list <- system.file("extdata",package = "BioThermR")
batch <- read_thermal_batch(img_obj_list)

# Create a montage2 with 4 columns
p <- viz_thermal_montage2(batch, ncol = 4, padding = 20)
```

Index