Package: pointcoral (via r-universe)

June 20, 2026

Title Local Point-Count Processing for Coral Photoquadrats

Version 0.1.0

Description Imports Coral Point Count with Excel extensions (CPCe) point-count annotations and related exported tables, standardizes labels with a user-supplied crosswalk, creates ecological cover summaries, writes quality-control overlays, and exports machine-learning-ready point labels, image patches, sparse masks, and train/validation/test splits. The package is fully local and does not depend on third-party web platforms, user accounts, or other closed services. CPCe methods are described by Kohler and Gill (2006) ``Coral Point Count with Excel extensions (CPCe): A Visual Basic program for the determination of coral and substrate coverage using random point count methodology" <doi:10.1016/j.cageo.2005.11.009>.

License MIT + file LICENSE

URL https://github.com/el-cordero/pointcoral

BugReports https://github.com/el-cordero/pointcoral/issues

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Depends R (>= 4.1)

Imports cli, dplyr, fs, janitor, magick, png, purrr, readr, readxl, rlang, stringr, tibble

Suggests devtools, glue, knitr, rmarkdown, roxygen2, styler, testthat (>= 3.0.0), tidyr, writexl

VignetteBuilder knitr

Config/testthat/edition 3

Config/pak/sysreqs cmake make libmagick++-dev gsfonts libicu-dev libpng-dev libuv1-dev libssl-dev libx11-dev

Repository https://el-cordero.r-universe.dev

Date/Publication 2026-06-12 10:34:33 UTC

RemoteUrl https://github.com/el-cordero/pointcoral

RemoteRef HEAD

RemoteSha f0b8d7e5cd8ca8de90de9631f04fc0d07d00ba22

Contents

check_crosswalk *Check a label crosswalk against point data*

Description

Reports labels in CPCe data missing from the crosswalk, crosswalk labels not present in current data, duplicate mappings, missing class IDs, and excluded classes.

Usage

```
check_crosswalk(points, crosswalk, by = NULL)
```

Arguments

points	A tidy point table.
crosswalk	A crosswalk data frame.
by	Optional join columns. See standardize_labels().

Value

A tibble report.

Examples

```
cpc <- system.file("extdata", "HIW_158_W_U-1.cpc", package = "pointcoral")
xwalk <- system.file(
  "extdata", "pointcoral_example_crosswalk.csv",
  package = "pointcoral"
)
pts <- read_cpce_file(cpc)
check_crosswalk(pts, read_label_crosswalk(xwalk))
```

convert_cpce_coords *Convert CPCe coordinates to image pixel coordinates*

Description

Converts point coordinates from the CPCe coordinate space to actual image pixels by proportional scaling. Original CPCe coordinates are preserved.

Usage

```
convert_cpce_coords(
  points,
  cpce_width = NULL,
  cpce_height = NULL,
  image_width = NULL,
  image_height = NULL
)
```

Arguments

points	A data frame with cpce_x and cpce_y columns.
cpce_width, cpce_height	CPCe coordinate-space width and height. If NULL, the function uses points$cpce_width and points$cpce_height.
image_width, image_height	Image width and height in pixels. If NULL, the function uses points$image_width and points$image_height.

Value

A tibble with updated x_px and y_px columns.

Examples

```
pts <- tibble::tibble(cpce_x = c(0, 50, 100), cpce_y = c(0, 25, 50))
convert_cpce_coords(pts, 100, 50, 1000, 500)
```

---

export_coralnet_points

Export point labels in a simple CoralNet-style CSV

Description

This helper writes local point labels only. It does not connect to CoralNet.

Usage

```
export_coralnet_points(points, out_dir)
```

Arguments

points	A tidy point table.
out_dir	Output directory.

Value

Path to the written CSV.

Examples

```
cpc <- system.file("extdata", "HIW_158_W_U-1.cpc", package = "pointcoral")
export_coralnet_points(
  read_cpce_file(cpc),
  file.path(tempdir(), "pointcoral-coralnet-example")
)
```

`export_segformer_sparse` *Export SegFormer-style sparse masks*

Description

Writes sparse weak-label masks and a manifest. These are not dense segmentation annotations.

Usage

```
export_segformer_sparse(points, image_root, out_dir, radius = 3)
```

Arguments

`points`	A tidy point table.
`image_root`	Image root.
`out_dir`	Output directory.
`radius`	Point disk radius.

Value

A mask manifest tibble.

Examples

```
example_dir <- system.file("extdata", package = "pointcoral")
pts <- read_cpce_file(file.path(example_dir, "HIW_158_W_U-1.cpc"))
export_segformer_sparse(
  pts[1:3, ],
  image_root = example_dir,
  out_dir = file.path(tempdir(), "pointcoral-segformer-example"),
  radius = 2
)
```

export_yolo_classification *Export YOLO-style classification patches*

Description

Extracts point-centered patches into split/class folders and writes a patch manifest.

Usage

```
export_yolo_classification(points, image_root, out_dir, patch_size = 224)
```

Arguments

points	A tidy point table.
image_root	Image root.
out_dir	Output directory.
patch_size	Patch size in pixels.

Value

A patch manifest tibble.

Examples

```
example_dir <- system.file("extdata", package = "pointcoral")
pts <- read_cpce_file(file.path(example_dir, "HIW_158_W_U-1.cpc"))
export_yolo_classification(
  pts[20:25, ],
  image_root = example_dir,
  out_dir = file.path(tempdir(), "pointcoral-yolo-example"),
  patch_size = 64
)
```

extract_point_patches *Extract point-centered image patches*

Description

Extracts square image patches centered on CPCe points and writes a manifest.

Usage

```
extract_point_patches(
  points,
  image_root,
  out_dir,
  patch_size = 224,
  class_col = "ml_class",
  edge = c("skip", "pad")
)
```

Arguments

points	A tidy point table.
image_root	Image root used to match images when image_path is absent.
out_dir	Output patch directory.
patch_size	Patch width/height in pixels.
class_col	Class column used for folder names and labels.
edge	"skip" skips points too close to an edge; "pad" pads images with black pixels before cropping.

Value

A patch manifest tibble.

Examples

```
example_dir <- system.file("extdata", package = "pointcoral")
pts <- read_cpce_file(file.path(example_dir, "HIW_158_W_U-1.cpc"))
out_dir <- file.path(tempdir(), "pointcoral-patches-example")
extract_point_patches(
  pts[1:3, ],
  image_root = example_dir,
  out_dir = out_dir,
  patch_size = 64,
  class_col = "raw_label",
  edge = "pad"
)
```

---

make_class_lookup *Make a class lookup table*

Description

Produces a distinct class lookup table from point data. If no usable class ID column exists, IDs are assigned in sorted class order starting at 0.

Usage

```
make_class_lookup(points, class_col = "ml_class", id_col = "class_id")
```

Arguments

points	A tidy point table.
class_col	Column containing class labels.
id_col	Column containing integer class IDs.

Value

A tibble with class labels and IDs.

Examples

```
cpc <- system.file("extdata", "HIW_158_W_U-1.cpc", package = "pointcoral")
xwalk <- system.file(
  "extdata", "pointcoral_example_crosswalk.csv",
  package = "pointcoral"
)
pts <- standardize_labels(read_cpce_file(cpc), read_label_crosswalk(xwalk))
make_class_lookup(pts, class_col = "ml_class")
```

---

make_ml_points *Make an ML-ready point-label table*

Description

Creates a compact table suitable for patch classification or weakly supervised segmentation workflows.

Usage

```
make_ml_points(points, image_root = NULL, class_col = "ml_class")
```

Arguments

points	A tidy point table.
image_root	Optional image root used to match image paths.
class_col	Label column to use as the ML label.

Value

A tibble with `image_path`, `image_id`, `x_px`, `y_px`, `label`, `class_id`, `split`, and available metadata.

Examples

```
cpc <- system.file("extdata", "HIW_158_W_U-1.cpc", package = "pointcoral")
make_ml_points(read_cpce_file(cpc), class_col = "raw_label")
```

`make_sparse_masks` *Create sparse semantic segmentation masks from point labels*

Description

Writes sparse masks where point neighborhoods contain `class_id` values and all unlabeled pixels are `ignore_index`. These are weak labels, not dense human-annotated segmentation masks.

Usage

```
make_sparse_masks(
  points,
  image_root,
  out_dir,
  radius = 3,
  ignore_index = 255,
  background_index = 0,
  class_col = "ml_class"
)
```

Arguments

`points`	A tidy point table.
`image_root`	Image root used to match images when needed.
`out_dir`	Output directory.
`radius`	Disk radius in pixels around each point.
`ignore_index`	Pixel value for unlabeled pixels.
`background_index`	Reserved background value. Included for downstream schemas; unlabeled pixels still default to `ignore_index`.
`class_col`	Label column used to assign `class_id` values when they are missing. Defaults to `ml_class`, with automatic fallback to raw CPCe labels for bare workflows without a crosswalk.

Value

A mask manifest tibble.

Examples

```
example_dir <- system.file("extdata", package = "pointcoral")
pts <- read_cpce_file(file.path(example_dir, "HIW_158_W_U-1.cpc"))
out_dir <- file.path(tempdir(), "pointcoral-masks-example")
make_sparse_masks(
  pts[1:3, ],
  image_root = example_dir,
  out_dir = out_dir,
  radius = 2,
  class_col = "raw_label"
)
```

`match_images` *Match CPCe point rows to image files*

Description

Matches `image_file` values in a point table to files under `image_root`. Image dimensions are filled in, and pixel coordinates are calculated when CPCe and image dimensions are available.

Usage

```
match_images(points, image_root, image_col = "image_file")
```

Arguments

`points`	A pointcoral point table.
`image_root`	Root folder containing images.
`image_col`	Column in `points` containing image file names.

Value

A point table with `image_path`, `image_width`, and `image_height`.

Examples

```
example_dir <- system.file("extdata", package = "pointcoral")
pts <- read_cpce_file(file.path(example_dir, "HIW_158_W_U-1.cpc"))
pts$image_path <- NA_character_
match_images(pts, image_root = example_dir)
```

plot_points_on_image *Plot CPCe points on an image*

Description

Draws point halos and labels on an image and returns a magick-image object.

Usage

```
plot_points_on_image(
  image_path,
  points,
  label_col = "ml_class",
  point_size = 8
)
```

Arguments

image_path	Path to an image.
points	Point rows for that image.
label_col	Column to use for text labels.
point_size	Point radius in pixels.

Value

A magick-image overlay.

Examples

```
example_dir <- system.file("extdata", package = "pointcoral")
pts <- read_cpce_file(file.path(example_dir, "HIW_158_W_U-1.cpc"))
plot_points_on_image(pts$image_path[1], pts[1:5, ], label_col = "raw_label")
```

qc_label_summary *Summarize point-label QC issues*

Description

Reports unmapped labels, rare labels, duplicate points, and class balance in a compact tibble.

Usage

```
qc_label_summary(points, label_col = "ml_class", rare_threshold = 1L)
```

Arguments

points	A tidy point table.
label_col	Label column to summarize.
rare_threshold	Count at or below which labels are flagged as rare.

Value

A QC summary tibble.

Examples

```
cpc <- system.file("extdata", "HIW_158_W_U-1.cpc", package = "pointcoral")
qc_label_summary(read_cpce_file(cpc), label_col = "raw_label")
```

---

read_cpce_export *Read a CPCe CSV or Excel export*

Description

Reads generic CPCe-like point exports from CSV, TSV, XLS, or XLSX files, cleans column names, and maps common coordinate/label columns to the pointcoral tidy point schema. Project-specific ecological "Data Summary" workbooks are not fully translated yet because representative workbook fixtures are not bundled with the package.

Usage

```
read_cpce_export(path)
```

Arguments

path	Path to a CSV, TSV, XLS, or XLSX export.

Value

A tidy point table when point-like columns are present.

Examples

```
cpc <- system.file("extdata", "HIW_158_W_U-1.cpc", package = "pointcoral")
pts <- read_cpce_file(cpc)
tmp <- tempfile(fileext = ".csv")
readr::write_csv(pts[, c("image_file", "point_id", "x_px", "y_px", "raw_label")], tmp)
read_cpce_export(tmp)
```

read_cpce_file *Read one CPCe .cpc file*

Description

Reads the tested text .cpc format used by the bundled examples: header row, four ROI vertices, point count, point coordinate rows, and point label rows. If an image with the same basename is present next to the .cpc file, image dimensions are read and x_px/y_px are calculated.

Usage

```
read_cpce_file(path)
```

Arguments

path	Path to a .cpc file.

Value

A tidy point table.

Examples

```
cpc <- system.file("extdata", "HIW_158_W_U-1.cpc", package = "pointcoral")
pts <- read_cpce_file(cpc)
dplyr::glimpse(pts)
```

read_cpce_folder *Read CPCe files and exports from a folder*

Description

Recursively reads supported CPCe-related files from a folder. .cpc files are parsed with read_cpce_file(). CSV/TSV/XLS/XLSX files are attempted with read_cpce_export().

Usage

```
read_cpce_folder(path, image_root = NULL, recursive = TRUE)
```

Arguments

path	Folder containing CPCe files/exports.
image_root	Optional image root used to match image paths after import.
recursive	Whether to search recursively.

Value

A combined tidy point table.

Examples

```
example_dir <- system.file("extdata", package = "pointcoral")
read_cpce_folder(example_dir, image_root = example_dir, recursive = FALSE)
```

`read_cpce_output_raw_tabs` *Read* `_raw` *sheets from a CPCe output workbook*

Description

CPCe output workbooks often contain one worksheet per image, with raw point annotation worksheets named like `image_name_raw`. This function reads only those raw worksheets, skips `deep_cres_..._raw` worksheets by default, keeps the raw worksheet columns, adds `image_name` and `point_index` as the first columns, and adds a full `major_category` column using a label crosswalk. `point_index` is a 1-based row index within each raw worksheet, matching the CPCe point order.

Usage

```
read_cpce_output_raw_tabs(
  path,
  crosswalk = NULL,
  skip_deep_cres = TRUE,
  sheet_pattern = "_raw$"
)
```

Arguments

`path`	Path to a CPCe output `.xls` or `.xlsx` workbook.
`crosswalk`	Optional label crosswalk data frame or path. When `NULL`, the bundled example crosswalk is used. The raw CPCe label in the workbook is joined to `raw_code` or `raw_label`.
`skip_deep_cres`	Whether to skip worksheets whose names start with `deep_cres_`. Those sheets use a different layout.
`sheet_pattern`	Regular expression used to identify raw worksheets. Defaults to sheets ending in `_raw`.

Value

A tibble combining all selected raw worksheets. The first columns are `image_name` and `point_index`. The original CPCe raw major/group column is preserved as `cpce_major_category` when present.

Examples

```
xlsx <- system.file(
  "extdata", "pointcoral_example_cpce_output_raw_tabs.xlsx",
  package = "pointcoral"
)
read_cpce_output_raw_tabs(xlsx)
```

`read_label_crosswalk` *Read a label crosswalk*

Description

Reads a CSV, TSV, XLS, or XLSX crosswalk table, cleans column names, recognizes common synonyms, and validates that the table contains at least one raw label/code key and at least one output class field.

Usage

```
read_label_crosswalk(path)
```

Arguments

`path`	Path to a crosswalk file.

Value

A tidy crosswalk tibble.

Examples

```
xwalk <- system.file(
  "extdata", "pointcoral_example_crosswalk.csv",
  package = "pointcoral"
)
read_label_crosswalk(xwalk)
```

`run_pointcoral` *Run the full pointcoral workflow from folders*

Description

Reads CPCe files/exports from a folder, matches images, validates points, and writes analysis/ML-ready outputs. A crosswalk is optional. Without one, `pointcoral` uses the raw labels already stored in the CPCe files. With one, it standardizes those raw labels to full labels, ecological classes, and custom ML classes.

Usage

```
run_pointcoral(
  cpce_dir,
  image_root,
  out_dir,
  crosswalk_path = NULL,
  recursive = TRUE,
  class_col = "ml_class",
  patch_size = 224,
  make_patches = TRUE,
  make_masks = FALSE,
  make_qc = TRUE
)
```

Arguments

`cpce_dir`	Folder containing CPCe files or exports.
`image_root`	Folder containing source images.
`out_dir`	Output directory.
`crosswalk_path`	Optional path to a user-supplied label crosswalk.
`recursive`	Whether to search `cpce_dir` recursively.
`class_col`	Class column to use for ML labels.
`patch_size`	Patch size for point-centered patches.
`make_patches`, `make_masks`, `make_qc`	Whether to write optional outputs.

Value

A list from `write_pointcoral_dataset()`.

Examples

```
example_dir <- system.file("extdata", package = "pointcoral")
run_pointcoral(
  cpce_dir = example_dir,
  image_root = example_dir,
  out_dir = file.path(tempdir(), "pointcoral-run-example"),
  recursive = FALSE,
  make_patches = FALSE,
  make_masks = FALSE,
  make_qc = FALSE,
  class_col = "raw_label"
)
```

split_ml_points *Split ML points into train/validation/test sets*

Description

Assigns split labels by image, transect, or site to avoid leakage. Splits are reproducible with seed.

Usage

```
split_ml_points(
  points,
  split_by = c("image", "transect", "site"),
  train = 0.7,
  val = 0.15,
  test = 0.15,
  seed = 1
)
```

Arguments

points	A tidy point table.
split_by	One of "image", "transect", or "site".
train, val, test	Split proportions.
seed	Random seed.

Value

The input table with split and split_unit columns.

Examples

```
example_dir <- system.file("extdata", package = "pointcoral")
pts <- read_cpce_folder(example_dir, image_root = example_dir, recursive = FALSE)
split_ml_points(pts, split_by = "image", train = 0.5, val = 0, test = 0.5)
```

standardize_labels *Standardize CPCe labels with a crosswalk*

Description

Joins raw CPCe labels/codes to user-defined clean labels, ecological categories, ML classes, and class IDs. Raw labels and raw codes are always preserved.

Usage

```
standardize_labels(
  points,
  crosswalk,
  by = NULL,
  unknown_action = c("warn", "keep", "drop", "error")
)
```

Arguments

points	A tidy point table.
crosswalk	A crosswalk data frame, usually from read_label_crosswalk().
by	Optional join columns. Use a character vector for same-name joins or a named vector where names are point-table columns and values are crosswalk columns.
unknown_action	How to handle labels not found in the crosswalk: "warn" keeps rows and warns, "keep" keeps rows silently, "drop" drops unmapped rows with a warning, and "error" stops.

Value

A tidy point table with standardized label columns.

Examples

```
cpc <- system.file("extdata", "HIW_158_W_U-1.cpc", package = "pointcoral")
xwalk <- system.file(
  "extdata", "pointcoral_example_crosswalk.csv",
  package = "pointcoral"
)
pts <- read_cpce_file(cpc)
standardize_labels(pts, read_label_crosswalk(xwalk))
```

---

summarize_images *Summarize points at image level*

Description

Summarize points at image level

Usage

```
summarize_images(points, class_col = "major_category")
```

Arguments

points	A tidy point table.
class_col	Class/label column to summarize.

Value

An image-level summary tibble.

Examples

```
cpc <- system.file("extdata", "HIW_158_W_U-1.cpc", package = "pointcoral")
summarize_images(read_cpce_file(cpc), class_col = "raw_label")
```

`summarize_points` *Summarize point counts and percent cover*

Description

Counts point labels by grouping variables and returns percent cover within each group.

Usage

```
summarize_points(
  points,
  by = c("site", "transect", "image_id"),
  class_col = "major_category"
)
```

Arguments

`points`	A tidy point table.
`by`	Grouping columns.
`class_col`	Class/label column to summarize.

Value

A summary tibble with `n`, `n_points`, and `percent`.

Examples

```
cpc <- system.file("extdata", "HIW_158_W_U-1.cpc", package = "pointcoral")
pts <- read_cpce_file(cpc)
summarize_points(pts, by = "image_id", class_col = "raw_label")
```

summarize_sites *Summarize points at site level*

Description

Summarize points at site level

Usage

```
summarize_sites(points, class_col = "major_category")
```

Arguments

points	A tidy point table.
class_col	Class/label column to summarize.

Value

A site-level summary tibble.

Examples

```
cpc <- system.file("extdata", "HIW_158_W_U-1.cpc", package = "pointcoral")
summarize_sites(read_cpce_file(cpc), class_col = "raw_label")
```

summarize_transects *Summarize points at transect level*

Description

Summarize points at transect level

Usage

```
summarize_transects(points, class_col = "major_category")
```

Arguments

points	A tidy point table.
class_col	Class/label column to summarize.

Value

A transect-level summary tibble.

Examples

```
cpc <- system.file("extdata", "HIW_158_W_U-1.cpc", package = "pointcoral")
summarize_transects(read_cpce_file(cpc), class_col = "raw_label")
```

---

validate_points *Validate a point table*

Description

Checks required fields, missing coordinates, duplicated point IDs within images, coordinates outside image bounds, missing image files, and missing image dimensions.

Usage

```
validate_points(points)
```

Arguments

points	A tidy point table.

Value

A validation report tibble.

Examples

```
cpc <- system.file("extdata", "HIW_158_W_U-1.cpc", package = "pointcoral")
validate_points(read_cpce_file(cpc))
```

---

write_ml_points_csv *Write ML point CSV files*

Description

Writes `labels.csv`, `class_lookup.csv`, and split-specific label CSVs when a `split` column is present.

Usage

```
write_ml_points_csv(points, out_dir)
```

Arguments

points	An ML-ready point table or tidy point table.
out_dir	Output directory.

Value

A named list of written file paths.

Examples

```
cpc <- system.file("extdata", "HIW_158_W_U-1.cpc", package = "pointcoral")
ml <- make_ml_points(read_cpce_file(cpc), class_col = "raw_label")
write_ml_points_csv(ml, file.path(tempdir(), "pointcoral-ml-csv-example"))
```

---

write_pointcoral_dataset

Write a complete pointcoral dataset from imported points

Description

Runs the common local workflow: validation, ecological summaries, train/validation/test splits, ML label CSVs, optional point patches, optional sparse masks, and optional QC overlays. If `crosswalk` is `NULL`, the workflow uses the raw CPCe labels already stored in the point table. A crosswalk is an optional standardization layer for full labels, ecological major classes, and custom ML classes.

Usage

```
write_pointcoral_dataset(
  points,
  image_root,
  out_dir,
  crosswalk = NULL,
  patch_size = 224,
  make_patches = TRUE,
  make_masks = FALSE,
  make_qc = TRUE,
  class_col = "ml_class"
)
```

Arguments

`points`	Imported point data.
`image_root`	Image root for matching images.
`out_dir`	Output directory.
`crosswalk`	Optional crosswalk data frame or path.
`patch_size`	Patch size for point-centered patches.
`make_patches, make_masks, make_qc`	Whether to write optional outputs.
`class_col`	Class column to use for ML labels.

Value

A list containing paths, tibbles, and manifests.

Examples

```
example_dir <- system.file("extdata", package = "pointcoral")
pts <- read_cpce_folder(example_dir, image_root = example_dir, recursive = FALSE)
write_pointcoral_dataset(
  points = pts,
  image_root = example_dir,
  out_dir = file.path(tempdir(), "pointcoral-dataset-example"),
  make_patches = FALSE,
  make_masks = FALSE,
  make_qc = FALSE,
  class_col = "raw_label"
)
```

---

write_qc_overlays *Write QC overlays for point annotations*

Description

Writes one overlay image per source image.

Usage

```
write_qc_overlays(points, image_root, out_dir, label_col = "ml_class")
```

Arguments

points	A tidy point table.
image_root	Image root used to match images when needed.
out_dir	Output directory.
label_col	Label column to draw.

Value

A manifest tibble of written overlays.

Examples

```
example_dir <- system.file("extdata", package = "pointcoral")
pts <- read_cpce_file(file.path(example_dir, "HIW_158_W_U-1.cpc"))
write_qc_overlays(
  pts[1:5, ],
  image_root = example_dir,
  out_dir = file.path(tempdir(), "pointcoral-qc-example"),
  label_col = "raw_label"
)
```

`write_summary_tables` *Write ecological summary tables*

Description

Writes image-, transect-, and site-level summaries for one or more class columns, plus a class lookup table when possible.

Usage

```
write_summary_tables(
  points,
  out_dir,
  class_cols = c("major_category", "clean_label", "ml_class")
)
```

Arguments

`points`	A tidy point table.
`out_dir`	Output directory for summary CSV files.
`class_cols`	Class columns to summarize.

Value

A named list of written file paths.

Examples

```
cpc <- system.file("extdata", "HIW_158_W_U-1.cpc", package = "pointcoral")
pts <- read_cpce_file(cpc)
out_dir <- file.path(tempdir(), "pointcoral-summary-example")
write_summary_tables(pts, out_dir, class_cols = "raw_label")
```

Index