# Package 'spacey'

March 14, 2020

**Type** Package

**Title** Easily Obtain Spatial Data and Make Better Maps

**Version** 0.1.1

**Description** One of the remaining pain points in making beautiful maps via packages like 'rayshader' is both obtaining and processing spatial data to build from. 'spacey' aims to make it easier to obtain and use this data for locations within the United States, providing utilities to download 'USGS' and 'ESRI' geospatial data and quickly turn it into maps.

**License** MIT + file LICENSE

**URL** https://github.com/mikemahoney218/spacey, https://mikemahoney218.github.io/spacey/

**BugReports** https://github.com/mikemahoney218/spacey/issues

**Encoding** UTF-8

**LazyData** true

**Imports** httr, raster, rayshader, magrittr, png, jsonlite, rgdal

**RoxygenNote** 7.0.2

**Suggests** testthat (>= 2.1.0), covr, knitr, rmarkdown, future, pkgload

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Michael Mahoney [aut, cre] (<https://orcid.org/0000-0003-2402-304X>)

**Maintainer** Michael Mahoney <mike.mahoney.218@gmail.com>

**Repository** CRAN

**Date/Publication** 2020-03-14 18:50:02 UTC

## R topics documented:

---

`automap` *Automatically create 2D and 3D maps using USGS and ESRI map data*

---

## Description

This function takes a latitude/longitude coordinate pair alongside a desired distance to map and retrieves USGS (and optionally ESRI) map data, converts said data into matrices, and runs the data through `rayshader` using sensible defaults in order to return a 2D shaded map relief. It requires a functioning internet connection in order to retrieve data.

## Usage

```
automap(
  lat = NULL,
  lng = NULL,
  distance = 10,
  method = c("2d", "3d"),
  img.width = 600,
  img.height = 600,
  overlay = NULL,
  z = 9,
  overlay.alpha = 0.75,
  colorscale = "imhof4",
  color.intensity = 1,
  max.darken = 0.5,
  sun.angle = 315,
  sun.altitude = 45,
  water.cutoff = 0.999,
  water.min.area = length(heightmap)/400,
  water.max.height = NULL,
  solid = TRUE,
  shadow = TRUE,
  water = FALSE,
  waterdepth = 0,
  theta = 45,
  phi = 45,
  fov = 0,
  zoom = 1,
```

```
  save.file = c(FALSE, "tif", "png", TRUE),
  from.file = c(FALSE, "tif", "png", TRUE),
  tif.filename = NULL,
  png.filename = NULL,
  dist.unit = c("km", "miles", "m", "ft"),
  coord.unit = c("degrees", "radians"),
  sr_bbox = 4326,
  sr_image = 4326,
  print.map = TRUE
)
```

## Arguments

| | |
|---|---|
| lat | The latitude for the map's centroid, in decimal degrees or radians. |
| lng | The longitude for the map's centroid, in decimal degrees or radians. |
| distance | The distance between the centroid and any corner of the (square) output map to include. |
| method | Should a 2d or 3d plot be produced? |
| img.width | Image width, in pixels |
| img.height | Image height, in pixels |
| overlay | ESRI overlay map to include, if any – see get_image_overlay for list of options. If specifying a local file with from.file, any non NULL value will add your local file as an overlay. |
| z | zscale, passed to various rayshader functions. Defined as the ratio between the x and y spacing (which are assumed to be equal) and the z axis. For contiguous United States, USGS data is generally available at a ~9 meter spacing, with elevation provided in meter increments, resulting in a z value of 9 returning roughly representative maps; decrease this value to exaggerate elevation features. |
| overlay.alpha | Alpha value for the optional overlay layer. |
| colorscale | Color scale for land and water elements. If a vector of length 1, the same color scale will be applied for both land and water. If greater than length 1, values named "water" or "watercolor" will be used for water coloring, while "land" or "landcolor" will be used for land. |
| color.intensity | Intensity of color mapping – higher values result in more intense colors. |
| max.darken | Passed to add_shadow. The lower limit for how much the image will be darkened. 0 is completely black, 1 means the shadow map will have no effect. |
| sun.angle | Angle around the matrix from which lights originate. Values line up with compass directions – so 0 is directly North, while the default 315 places the sun in the Northwest. |
| sun.altitude | Angle in degrees from horizon from which light originates. Bounded [0, 90]. |
| water.cutoff | Passed to detect_water. Defined therein as the lower limit of the z-component of the unit normal vector to be classified as water. |
| water.min.area | Passed to detect_water. The minimum possible area to consider a body of water. |

`water.max.height`
: Passed to `detect_water`. If provided, the maximum height a point can be classified water.

`solid`
: Logical – should the output be rendered as a solid (`TRUE`) or just a surface (`FALSE`)?

`shadow`
: Logical – should shadows be rendered?

`water`
: Logical – should water be rendered?

`waterdepth`
: Water level.

`theta`
: Rotation around z axis.

`phi`
: Azimuth angle.

`fov`
: Field of view angle.

`zoom`
: Zoom factor.

`save.file`
: Should the heightmap (= `"tif"`), overlay (= `"png"`), or both (= `TRUE`) be saved? Default `FALSE` saves neither.

`from.file`
: Should the map be built from local `.tif` and `.png` files, rather than downloaded data? Accepts logical `FALSE` (no local files used) and `TRUE` (local files used for both height maps and textures, if requested), as well as strings `tif` (only use height map, redownload PNG) and `png` (only use texture, redownload tif). Overrides save arguments if local files are used.

`tif.filename`
: If `save.tif` is `TRUE`, filename to save the `.tif` height map to. If `from.file` is `TRUE`, filename to load the height map from.

`png.filename`
: If `save.png` is `TRUE`, filename to save the `.png` texture to. If `from.file` is `TRUE`, filename to load the texture from.

`dist.unit`
: Units for the distance argument. All units are converted to kilometers, with some errors in the transition due to floating point arithmetic – so if high accuracy is needed, convert units to kilometers beforehand.

`coord.unit`
: Units for latitude and longitude, in either decimal degrees or radians.

`sr_bbox`
: Spatial reference code (ISO 19111) for bounding box

`sr_image`
: Spatial reference code (ISO 19111) for image

`print.map`
: Logical – should the output map be printed?

## Value

A rayshader map object, by default printed and returned invisibly.

## References

Archuleta, C.M., Constance, E.W., Arundel, S.T., Lowe, A.J., Mantey, K.S., and Phillips, L.A., 2017, The National Map seamless digital elevation model specifications: U.S. Geological Survey Techniques and Methods, book 11, chap. B9, 39 p., https://doi.org/10.3133/tm11B9

## Examples

```
## Not run:
automap(44.121268, -73.903734)
automap(44.121268, -73.903734, overlay = "World_Imagery")
automap(44.121268, -73.903734, overlay = "World_Imagery", method = "3d")

## End(Not run)
```

---

`deg_to_rad` *Convert decimal degrees to radians*

---

## Description

Convert decimal degrees to radians

## Usage

```
deg_to_rad(deg)
```

## Arguments

| | |
|---|---|
| `deg` | A vector of values, in decimal degrees, to convert to radians |

## Value

A vector of the same length in radians

## Examples

```
deg_to_rad(360)
rad_to_deg(deg_to_rad(360))
```

---

`get_centroid` *Find central point for list of lat/long coordinates*

---

## Description

Find central point for list of lat/long coordinates

## Usage

```
get_centroid(lat, lng, coord.unit = c("degrees", "radians"))
```

### Arguments

| | |
|---|---|
| lat | A quoted string indicating what named value in the bounding box represents latitude. If NULL, will be inferred from bounding box names. |
| lng | A quoted string indicating what named value in the bounding box represents longitude. If NULL, will be inferred from bounding box names. |
| coord.unit | The unit latitude and longitude are stored in. |

### Examples

```
df <- data.frame(
  lat = c(44.05771, 44.18475),
  lng = c(-73.99212, -73.81515)
)
get_centroid(df$lat, df$lng)
```

---

## get_centroid_bounding_box

*Get bounding box for set of coordinate points*

---

### Description

Get bounding box for set of coordinate points

### Usage

```
get_centroid_bounding_box(
  centroid,
  distance,
  lat = NULL,
  lng = NULL,
  dist.unit = c("km", "miles", "m", "ft"),
  coord.unit = c("degrees", "radians")
)
```

### Arguments

| | |
|---|---|
| centroid | A vector of length 2 containing latitude and longitude values. |
| distance | The distance from the centroid to extend the bounding box. |
| lat | A quoted string indicating what named value in the centroid represents latitude. If NULL, will be inferred from centroid names. |
| lng | A quoted string indicating what named value in the centroid represents longitude. If NULL, will be inferred from centroid names. |
| dist.unit | A single value representing the units the distance value is in. |
| coord.unit | A single value representing the units the coordinates are in. |

## Value

A list of length 2, containing the bottom-left (named "bl") and top-right (named "tr") coordinates of the bounding box.

## Examples

```
get_centroid_bounding_box(c(
  "lat" = 44.121268,
  "lng" = -73.903734
),
distance = 10
)
```

---

`get_coord_bounding_box`

*Get bounding box for set of coordinate points*

---

## Description

Get bounding box for set of coordinate points

## Usage

```
get_coord_bounding_box(lat, lng)
```

## Arguments

| | |
|---|---|
| `lat` | A vector of latitudes |
| `lng` | A vector of longitudes |

## Value

A list of length 2, containing the bottom-left (named "bl") and top-right (named "tr") coordinates of the bounding box.

## Examples

```
df <- data.frame(
  lat = c(44.05771, 44.18475),
  lng = c(-73.99212, -73.81515)
)
get_coord_bounding_box(df$lat, df$lng)
```

get_heightmap *Retrieve height map from USGS national map API*

## Description

This function retrieves elevation data from the USGS national map API and converts it into a matrix appropriate for further usage with tools such as rayshader. It requires a functioning internet connection to retrieve data.

## Usage

```
get_heightmap(
  bbox,
  img.width = 600,
  img.height = 600,
  lat = NULL,
  lng = NULL,
  save.tif = FALSE,
  tif.filename = NULL,
  sr_bbox = 4326,
  sr_image = 4326,
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| bbox | Bounding box to download imagery for |
| img.width | Image width, in pixels |
| img.height | Image size, in pixels |
| lat | A quoted string indicating what named value in the bounding box represents latitude. If NULL, will be inferred from bounding box names. |
| lng | A quoted string indicating what named value in the bounding box represents longitude. If NULL, will be inferred from bounding box names. |
| save.tif | Logical: should the downloaded imagery be saved as a file? |
| tif.filename | If save.tif is TRUE, the filepath to save the resulting .tif to. |
| sr_bbox | Spatial reference code (ISO 19111) for bounding box |
| sr_image | Spatial reference code (ISO 19111) for image |
| verbose | Logical: print out debug information while trying to query the elevation map server? |

## Value

A matrix object containing elevation data suitable for use with mapping functions. Returned invisibly.

## Examples

```
## Not run:
bbox <- get_centroid_bounding_box(c(
  "lat" = 44.121268,
  "lng" = -73.903734
),
distance = 10
)

heightmap <- get_heightmap(bbox)

## End(Not run)
```

---

`get_image_overlay` *Get an ESRI image overlay for a rayshader map*

---

## Description

This function calls the ESRI ArcGIS API to retrieve map imagery to overlay a rayshader map object. It requires a functioning internet connection to obtain the data.

## Usage

```
get_image_overlay(
  bbox,
  overlay = c("World_Imagery", "NatGeo_World_Map", "USA_Topo_Maps",
   "World_Physical_Map", "World_Shaded_Relief", "World_Street_Map", "World_Terrain_Base",
    "World_Topo_Map"),
  img.width = 600,
  img.height = 600,
  lat = NULL,
  lng = NULL,
  save.png = FALSE,
  png.filename = NULL,
  sr_bbox = 4326,
  sr_image = 4326
)
```

## Arguments

| | |
|---|---|
| `bbox` | The bounding box for your image. |
| `overlay` | One of the available overlays from https://services.arcgisonline.com/ArcGIS/rest/services. |
| `img.width` | Image width, in pixels |
| `img.height` | Image height, in pixels |

| | |
|---|---|
| lat | A quoted string indicating what named value in the bounding box represents latitude. If NULL, will be inferred from bounding box names. |
| lng | A quoted string indicating what named value in the bounding box represents longitude. If NULL, will be inferred from bounding box names. |
| save.png | Logical – should the overlay image be saved? |
| png.filename | Filename for the overlay if save.png is TRUE. |
| sr_bbox | Spatial reference code (ISO 19111) for bounding box. |
| sr_image | Spatial reference code (ISO 19111) for image. |

## Value

A matrix object provided by readPNG, suitable for use with rayshader and similar mapping utilities. Returned invisibly.

## Examples

```
## Not run:
bbox <- get_centroid_bounding_box(c(
  "lat" = 44.121268,
  "lng" = -73.903734
),
distance = 10
)

get_image_overlay(bbox, overlay = "World_Imagery")

## End(Not run)
```

---

# load_heightmap *Load an elevation map from file*

---

## Description

Load an elevation map from file

## Usage

```
load_heightmap(filename)
```

## Arguments

| | |
|---|---|
| filename | The path to the .tif file to import as an elevation map. |

## Value

A matrix of elevations for use with further mapping utilities.

## Examples

```
## Not run:
bbox <- get_centroid_bounding_box(c(
  "lat" = 44.121268,
  "lng" = -73.903734
),
distance = 10
)

heightmap_file <- tempfile("heightmap_file", fileext = ".tif")
get_heightmap(bbox, save.tif = TRUE, filename = heightmap_file)
heightmap <- load_heightmap(heightmap_file)

## End(Not run)
```

---

`load_overlay` *Import PNG textures for overlays*

---

## Description

This is an extremely thin wrapper for `readPNG`, in order for users to not realize they even needed png for this package's functionality at all.

## Usage

```
load_overlay(filename)
```

## Arguments

| | |
|---|---|
| `filename` | The path to the PNG file to be imported |

## Value

A matrix of values provided by `readPNG`

## Examples

```
## Not run:
bbox <- get_centroid_bounding_box(c(
  "lat" = 44.121268,
  "lng" = -73.903734
),
distance = 10
)

overlay_file <- tempfile("overlay_file", fileext = ".png")
get_image_overlay(bbox,
  save.png = TRUE,
```

```
  png.filename = overlay_file,
  overlay = "World_Imagery"
)
overlay <- load_overlay(overlay_file)

## End(Not run)
```

---

`rad_to_deg` *Convert radians to degrees*

---

## Description

Convert radians to degrees

## Usage

```
rad_to_deg(rad)
```

## Arguments

| | |
|---|---|
| `rad` | A vector of values, in radians, to convert to decimal degrees |

## Value

A vector of the same length in decimal degrees

## Examples

```
rad_to_deg(2 * base::pi)
rad_to_deg(deg_to_rad(360))
```

# Index