Package ‘flightplanning’

February 24, 2021

Type Package
Title UAV Flight Planning
Version 0.8.4
Description Utility functions for creating flight plans for unmanned aerial vehicles (UAV), specifically for the Litchi Hub platform. It calculates the flight and camera settings based on the camera specifications, exporting the flight plan CSV format ready to import into Litchi Hub.

Imports graphics, grDevices, methods, rgdal, rgeos, sp
Depends R (>= 3.0)
Suggests testthat
License MIT + file LICENSE
Encoding UTF-8
LazyData true
RoxygenNote 7.1.0

URL https://github.com/caiohamamura/flightplanning-R

BugReports https://github.com/caiohamamura/flightplanning-R/issues

Author Caio Hamamura [aut, cre],
Danilo Roberti Alves de Almeida [aut],
Daniel de Almeida Papa [aut],
Hudson Franklin Pessoa Veras [aut],
Evandro Orfanó Figueiredo [aut]

Maintainer Caio Hamamura <caiohamamura@gmail.com>

Repository CRAN

Repository/R-Forge/Project flightplanning
Repository/R-Forge/Revision 17
Repository/R-Forge/DateTimeStamp 2021-02-23 14:15:25

Date/Publication 2021-02-24 09:40:02 UTC

NeedsCompilation no
### R topics documented:

- `adjustAcuteAngles` .................................................. 2
- `Flight Parameters-class` ........................................... 2
- `flight.parameters` .................................................. 3
- `getAngles` ............................................................ 4
- `getBBoxAngle` ....................................................... 4
- `getMinBBox` .......................................................... 5
- `litchi.plan` ........................................................... 5
- `outerCurvePoints` ................................................... 7

### adjustAcuteAngles

Given a xy matrix of points, adjust the points to avoid acute angles < 80 degrees.

**Description**

Given a xy matrix of points, adjust the points to avoid acute angles < 80 degrees.

**Usage**

```r
adjustAcuteAngles(xy, angle, minAngle = 80)
```

**Arguments**

- `xy` xy dataframe
- `angle` angle of the flight lines
- `minAngle` the minimum angle to below which will be projected

### Flight Parameters-class

*Class for Flight Parameters*

**Description**

Class for Flight Parameters
**flight.parameters**  
*Function to calculate flight parameters*

**Description**

This function will calculate the flight parameters by providing the camera settings target flight height or gsd, front and side overlap.

**Usage**

```r
flight.parameters(
  height = NA,
  gsd = NA,
  focal.length35 = 20,
  image.width.px = 4000,
  image.height.px = 3000,
  side.overlap = 0.8,
  front.overlap = 0.8,
  flight.speed.kmh = 54
)
```

**Arguments**

- **height**  
  target flight height, default NA
- **gsd**  
  target ground resolution in centimeters, must provide either 'gsd' or 'height'
- **focal.length35**  
  numeric. Camera focal length 35mm equivalent, default 20
- **image.width.px**  
  numeric. Image width in pixels, default 4000
- **image.height.px**  
  numeric. Image height in pixels, default 3000
- **side.overlap**  
  desired width overlap between photos, default 0.8
- **front.overlap**  
  desired height overlap between photos, default 0.8
- **flight.speed.kmh**  
  flight speed in km/h, default 54.

**Examples**

```r
params = flight.parameters(
  gsd = 4,
  side.overlap = 0.8,
  front.overlap = 0.8,
  flight.speed.kmh = 54
)
```
**getAngles**

*Get angles for each point considering the two neighbors points*

**Description**

Get angles for each point considering the two neighbors points

**Usage**

`getAngles(waypoints)`

**Arguments**

- **waypoints**
  the waypoints of the flight plan

---

**getBBoxAngle**

*Provided an angle, calculate the corresponding minimum bounding box*

**Description**

Provided an angle, calculate the corresponding minimum bounding box

**Usage**

`getBBoxAngle(vertices, alpha)`

**Arguments**

- **vertices**
  the vertices which to get the bounding box from
- **alpha**
  the angle to rotate the bounding box
**getMinBBox**

**Description**

Calculates the minimum oriented bounding box using the rotating calipers algorithm. Credits go to Daniel Wollschlaeger <https://github.com/ramnathv>.

**Usage**

getMinBBox(xy)

**Arguments**

- **xy**: A matrix of xy values from which to calculate the minimum oriented bounding box.

---

**litchi.plan**

*Function to generate Litchi csv flight plan*

**Description**

Function to generate Litchi csv flight plan

**Usage**

litchi.plan(
    roi,
    output,
    flight.params,
    gimbal.pitch.angle = -90,
    flight.lines.angle = -1,
    max.waypoints.distance = 2000,
    max.flight.time = 15,
    starting.point = 1
)

**Arguments**

- **roi**: range of interest loaded as an OGR layer, must be in a metric units projection for working properly
- **output**: output path for the csv file
- **flight.params**: Flight Parameters. parameters calculated from flight.parameters()
- **gimbal.pitch.angle**: gimbal angle for taking photos, default -90 (overriden at flight time)
litchi.plan

flight.lines.angle
  angle for the flight lines, default -1 (auto set based on larger direction)

max.waypoints.distance
  maximum distance between waypoints in meters, default 2000 (some issues have been reported with distances > 2 Km)

max.flight.time
  maximum flight time. If mission is greater than the estimated time, it will be splitted into smaller missions.

starting.point numeric (1, 2, 3 or 4). Change position from which to start the flight, default 1

Value
  A data frame with the waypoints calculated for the flight plan

Note
  this function will feed the csv flight plan with the 'gimbal.pitch.angle' and the 'photo time interval' for each waypoint, but those are not supported by Litchi yet, although they are present in the exported csv from the Litchi hub platform, though it may be supported in the future; when it does the function will already work with this feature.

Examples

library(flightplanning)
library(rgdal)

exampleBoundary = readOGR(
  system.file("extdata",
    "exampleBoundary.shp",
    package="flightplanning"
  ),
  "exampleBoundary")

outPath = tempfile(fileext=".csv")

flight.params = flight.parameters(
  gsd = 4,
  side.overlap = 0.8,
  front.overlap = 0.8,
  flight.speed.kmh = 54
)

litchi.plan(exampleBoundary,
  outPath,
  flight.params,
  flight.lines.angle = -1,
  max.waypoints.distance = 2000,
  max.flight.time = 15)
outerCurvePoints

Create outer curves for the flight lines

**Description**

Create outer curves for the flight lines

**Usage**

outerCurvePoints(waypoints, angle, flightLineDistance)

**Arguments**

- **waypoints**: the waypoints of the flight plan
- **angle**: angle for the flight lines
- **flightLineDistance**: the distance between the flight lines in meters
Index

adjustAcuteAngles, 2

Flight Parameters-class, 2
flight.parameters, 3

getAngles, 4
getBBoxAngle, 4
getMinBBox, 5

litchi.plan, 5

outerCurvePoints, 7