Package 'soccermatics'

August 9, 2021

Version 0.9.5

Title Visualise football (soccer) tracking and event data

Description Provides tools to visualise x,y-coordinates of soccer players and event data (e.g. passes, shots). Uses ggplot to draw soccer pitch and overplot expected goal maps, pass maps, average player positions, player heatmaps, individual player paths, player flow fields, and more.

Depends R (>= 3.4.1)

Imports cowplot, dplyr, forcats, ggforce, ggplot2, ggrepel, magrittr, MASS, plyr, rlang, scales, tidyr, xts, zoo

License GPL (>=3.0) Note: Use of the name 'soccermatics' was kindly permitted by David Sumpter and is protected from commercial use under EU copyright law.

Encoding UTF-8

Collate 'package.R'
'soccerFlipDirection.R'
'soccerPitch.R'
'soccerHeatmap.R'
'soccerSpokes.R'
'soccerFlow.R'
'soccerPassmap.R'
'soccerPath.R'
'soccerPitchBG.R'
'soccerPitchFG.R'
'soccerShotmap.R'
'soccerPitchHalf.R'
'soccerPositionMap.R'
'soccerResample.R'
'soccerShortenName.R'
'soccerStandardiseCols.R'
'soccerTransform.R'
'soccerVelocity.R'
'soccermatics-deprecated.R'
'soccerxGTimeline.R'
'statsbomb.R'
'tromso.R'
'tromso_extra.R'

RoxygenNote 7.1.1

R topics documented:

`soccerFlipDirection` *Flips x,y-coordinates horizontally in one half to account for changing sides at half-time*

Description

Normalises direction of attack in both halves of both teams by flipping x,y-coordinates horizontally in either the first or second half; i.e. teams attack in the same direction all game despite changing sides at half-time.

Usage

```
soccerFlipDirection(
  df,
  lengthPitch = 105,
  widthPitch = 68,
  teamToFlip = NULL,
  periodToFlip = 1:2,
  team = "team",
  period = "period",
  x = "x",
  y = "y"
)
```

Arguments

df	dataframe containing unnormalised x,y-coordinates
lengthPitch, widthPitch	length, width of pitch in metres
teamToFlip	character, name of team to flip. If NULL, all x,y-coordinates in df will be flipped
periodToFlip	integer, period(s) to flip
team	character, name of variables containing x,y-coordinates
period	character, name of variable containing period labels
x, y	character, name of variables containing x,y-coordinates

Value

a dataframe

Examples

```
library(dplyr)

# flip x,y-coords of France in both halves of statsbomb data
data(statsbomb)
statsbomb %>%
  soccerFlipDirection(team = "team.name", x = "location.x", y = "location.y",
                      teamToFlip = "France")

# flip x,y-coords in 2nd half of Tromso, based on a dummy period variable
data(tromso)
tromso %>%
  mutate(period = if_else(t > as.POSIXct("2013-11-07 21:14:00 GMT"), 1, 2)) %>%
  soccerFlipDirection(periodToFlip = 2)
```

---

soccerFlow *Draw a flow field of passing direction on a soccer pitch*

Description

A flow field to show the mean angle and distance of passes in zones of the pitch

Usage

```
soccerFlow(
  df,
  lengthPitch = 105,
  widthPitch = 68,
  xBins = 5,
  yBins = NULL,
  x = "x",
  y = "y",
  angle = "angle",
  distance = "distance",
```

```
  col = "black",
  lwd = 0.5,
  arrow = c("none", "r", "l"),
  title = NULL,
  subtitle = NULL,
  theme = c("light", "dark", "grey", "grass"),
  plot = NULL
)
```

Arguments

df	dataframe of event data containing fields of start x,y-coordinates, pass distance, and pass angle
lengthPitch, widthPitch	numeric, length and width of pitch in metres.
xBins, yBins	integer, the number of horizontal (length-wise) and vertical (width-wise) bins the soccer pitch is to be divided up into; if yBins is NULL (default), it will take the value of xBins
x, y, angle, distance	names of variables containing pass start x,y-coordinates, angle, and distance
col	colour of arrows
lwd	thickness of arrow segments
arrow	adds team direction of play arrow as right ('r') or left ('l'); 'none' by default
title, subtitle	adds title and subtitle to plot; NULL by default
theme	palette of pitch background and lines, either light (default), dark, grey, or grass
plot	base plot to add path layer to; NULL by default

Value

a ggplot object of a heatmap on a soccer pitch

See Also

soccerHeatmap for drawing a heatmap of player position, or soccerSpokes for drawing spokes to show all directions in each area of the pitch.

Examples

```
library(dplyr)
data(statsbomb)

# transform x,y-coords, filter only France pass events,
# draw flow field showing mean angle, distance of passes per pitch zone
statsbomb %>%
  soccerTransform(method = 'statsbomb') %>%
  filter(team.name == "France" & type.name == "Pass") %>%
  soccerFlow(xBins=7, yBins=5,
             x="location.x", y="location.y", angle="pass.angle", distance="pass.length")

# transform x,y-coords, standarise column names,
```

```
# filter only France pass events
my_df <- statsbomb %>%
  soccerTransform(method = 'statsbomb') %>%
  soccerStandardiseCols(method = 'statsbomb') %>%
  filter(team_name == "France" & event_name == "Pass")

# overlay flow field onto heatmap showing proportion of team passes per pitch zone
soccerHeatmap(my_df, xBins=7, yBins=5) %>%
  soccerFlow(my_df, xBins=7, yBins=5, plot = .)
```

`soccerHeatmap` *Draw a heatmap on a soccer pitch using any event or tracking data.*

Description

Draws a heatmap showing player position frequency in each area of the pitch and adds soccer pitch outlines.

Usage

```
soccerHeatmap(
  df,
  lengthPitch = 105,
  widthPitch = 68,
  xBins = 10,
  yBins = NULL,
  kde = FALSE,
  arrow = c("none", "r", "l"),
  colLow = "white",
  colHigh = "red",
  title = NULL,
  subtitle = NULL,
  x = "x",
  y = "y"
)
```

Arguments

`df`	dataframe containing x,y-coordinates of player position
`lengthPitch, widthPitch`	numeric, length and width of pitch in metres.
`xBins, yBins`	integer, the number of horizontal (length-wise) and vertical (width-wise) bins the soccer pitch is to be divided up into. If no value for `yBins` is provided, it will take the value of `xBins`.
`kde`	use kernel density estimates for a smoother heatmap; FALSE by default
`arrow`	adds team direction of play arrow as right (`'r'`) or left (`'l'`); `'none'` by default
`colLow, colHigh`	character, colours for the low and high ends of the heatmap gradient; white and red respectively by default

`title, subtitle`
: adds title and subtitle to plot; NULL by default

`x, y`
: name of variables containing x,y-coordinates

Details

uses `ggplot2::geom_bin2d` to map 2D bin counts

Value

a ggplot object of a heatmap on a soccer pitch.

Examples

```
library(dplyr)

# tracking data heatmap with 21x5 zones(~5x5m)
data(tromso)
tromso %>%
  filter(id == 8) %>%
  soccerHeatmap(xBins = 10)

# transform x,y-coords, filter only France pressure events,
# heatmap with 6x3 zones
data(statsbomb)
statsbomb %>%
  soccerTransform(method='statsbomb') %>%
  filter(type.name == "Pressure" & team.name == "France") %>%
  soccerHeatmap(x = "location.x", y = "location.y",
                xBins = 6, yBins = 3, arrow = "r",
                title = "France (vs Argentina, 30th June 2016)",
                subtitle = "Defensive pressure heatmap")

# transform x,y-coords, standardise column names,
# filter player defensive actions, plot kernel density estimate heatmap
statsbomb %>%
  soccerTransform(method='statsbomb') %>%
  soccerStandardiseCols() %>%
  filter(event_name %in% c("Duel", "Interception", "Clearance", "Block") &
         player_name == "Samuel Yves Umtiti") %>%
  soccerHeatmap(kde = TRUE, arrow = "r",
                title = "Umtiti (vs Argentina, 30th June 2016)",
                subtitle = "Defensive actions heatmap")
```

---

soccerPassmap *Draw a passing network using StatsBomb data*

Description

Draw an undirected passing network of completed passes on pitch from StatsBomb data. Nodes are scaled by number of successful passes; edge width is scaled by number of successful passes between each node pair. Only passes made until first substition shown (ability to specify custom minutes will be added soon). Total number of passes attempted and percentage of completed passes shown. Compatability with other (non-StatsBomb) shot data will be added soon.

Usage

```
soccerPassmap(
  df,
  lengthPitch = 105,
  widthPitch = 68,
  minPass = 3,
  fill = "red",
  col = "black",
  edgeAlpha = 0.6,
  edgeCol = NULL,
  label = TRUE,
  shortNames = TRUE,
  maxNodeSize = 30,
  maxEdgeSize = 30,
  labelSize = 4,
  arrow = c("none", "r", "l"),
  theme = c("light", "dark", "grey", "grass"),
  title = NULL
)
```

Arguments

df	dataframe containing x,y-coordinates of player passes
lengthPitch, widthPitch	numeric, length and width of pitch, in metres
minPass	minimum number of passes between players for edge to be drawn
fill, col	fill and border colour of nodes
edgeAlpha	transparency of edge lines, from 0 - 1. Defaults to 0.6 so overlapping edges are visible.
edgeCol	colour of edge lines. Default is complementary to theme colours.
label	boolean, draw labels
shortNames	shorten player names to display last name as label
maxNodeSize	maximum size of nodes
maxEdgeSize	maximum width of edge lines
labelSize	size of player name labels
arrow	optional, adds team direction of play arrow as right ('r') or left ('l')
theme	draws a light, dark, grey, or grass coloured pitch
title	adds custom title to plot. Defaults to team name.

Examples

```
# France vs. Argentina, minimum of three passes
library(dplyr)
data(statsbomb)

# transform x,y-coords,
# Argentina pass map until first substituton with transparent edges
statsbomb %>%
  soccerTransform(method='statsbomb') %>%
```

```
  filter(team.name == "Argentina") %>%
  soccerPassmap(fill = "lightblue", arrow = "r",
                title = "Argentina (vs France, 30th June 2018)")

# transform x,y-coords,
# France pass map until first substitution with opaque edges
statsbomb %>%
  filter(team.name == "France") %>%
  soccerTransform(method='statsbomb') %>%
  soccerPassmap(fill = "blue", minPass = 3,
                maxEdgeSize = 30, edgeCol = "grey40", edgeAlpha = 1,
                title = "France (vs Argentina, 30th June 2018)")
```

`soccerPath` *Draw a path of player trajectory on a soccer pitch using any tracking data*

Description

Draws a path connecting consecutive x,y-coordinates of a player on a soccer pitch.

Usage

```
soccerPath(
  df,
  lengthPitch = 105,
  widthPitch = 68,
  col = "black",
  arrow = c("none", "r", "l"),
  theme = c("light", "dark", "grey", "grass"),
  lwd = 1,
  title = NULL,
  subtitle = NULL,
  legend = FALSE,
  x = "x",
  y = "y",
  id = NULL,
  plot = NULL
)
```

Arguments

`df`	dataframe containing x,y-coordinates of player position
`lengthPitch, widthPitch`	length and width of pitch in metres
`col`	colour of path if no `'id'` is provided; if an `'id'` is present, uses ColorBrewer's 'Paired' palette by default
`arrow`	adds team direction of play arrow as right (`'r'`) or left (`'l'`); `'none'` by default
`theme`	draws a `light`, `dark`, `grey`, or `grass` coloured pitch
`lwd`	player path thickness

title, subtitle	adds title and subtitle to plot; NULL by default
legend	boolean, include legend
x, y	name of variables containing x,y-coordinates
id	character, the name of the column containing player identity (only required if 'df' contains multiple players)
plot	base plot to add path layer to; NULL by default

Value

a ggplot object

Examples

```
library(dplyr)
data(tromso)

# draw path of Tromso #8 over first 3 minutes (1800 frames)
tromso %>%
  filter(id == 8) %>%
  top_n(1800) %>%
  soccerPath(col = "red", theme = "grass", arrow = "r")

# draw path of all Tromso players over first minute (600 frames)
tromso %>%
  group_by(id) %>%
  slice(1:1200) %>%
  soccerPath(id = "id", theme = "light")
```

---

soccerPitch *Plot a full soccer pitch*

Description

Draws a soccer pitch as a ggplot object for the purpose of adding layers such as player positions, player trajectories, etc..

Usage

```
soccerPitch(
  lengthPitch = 105,
  widthPitch = 68,
  arrow = c("none", "r", "l"),
  title = NULL,
  subtitle = NULL,
  theme = c("light", "dark", "grey", "grass"),
  data = NULL
)
```

Arguments

lengthPitch, widthPitch	length and width of pitch in metres
arrow	adds team direction of play arrow as right ('r') or left ('l'); 'none' by default
title, subtitle	adds title and subtitle to plot; NULL by default
theme	palette of pitch background and lines, either light (default), dark, grey, or grass
data	a default dataset for plotting in subsequent layers; NULL by default

Value

a ggplot object

Examples

```
library(ggplot2)
data(statsbomb)

# transform Statsbomb coordinates to metre units for plotting
my_df <- soccerTransform(statsbomb, method = "statsbomb")

# filter events of interest (France defensive pressure events vs. Argentina)
my_df <- my_df %>%
  dplyr::filter(team.name == "France" & type.name == "Pressure")

# add custom layers to soccerPitch base
soccerPitch(data = my_df,
            arrow = "r", theme = "grass",
            title = "France (vs. Argentina)",
            subtitle = "Pressure events") +
  geom_point(aes(x = location.x, y = location.y),
             col = "blue", alpha = 0.5)
```

---

soccerPitchFG *Helper function to draw soccer pitch outlines over an existing ggplot object*

Description

Adds soccer pitch outlines (with transparent fill) to an existing ggplot object (e.g. heatmaps, passing maps, etc..)

Usage

```
soccerPitchFG(
  plot,
  lengthPitch = 105,
  widthPitch = 68,
  colPitch = "black",
```

```
  arrow = c("none", "r", "l"),
  title = NULL,
  subtitle = NULL
)
```

Arguments

`plot`	an existing ggplot object to add pitch lines layer to
`lengthPitch, widthPitch`	length and width of pitch in metres
`colPitch`	colour of pitch markings
`arrow`	adds team direction of play arrow as right (`'r'`) or left (`'l'`); `'none'` by default
`title, subtitle`	adds title and subtitle to plot; NULL by default

Value

a ggplot object

See Also

`soccerPitch` for plotting a soccer pitch as background layer

---

soccerPitchHalf *Draws a vertical half soccer pitch for the purpose of plotting shotmaps*

Description

Adds soccer pitch outlines (with transparent fill) to an existing ggplot object (e.g. heatmaps, passing maps, etc..)

Usage

```
soccerPitchHalf(
  lengthPitch = 105,
  widthPitch = 68,
  arrow = c("none", "r", "l"),
  theme = c("light", "dark", "grey", "grass"),
  title = NULL,
  subtitle = NULL,
  data = NULL
)
```

Arguments

`lengthPitch, widthPitch`	length and width of pitch in metres
`arrow`	adds team direction of play arrow as right (`'r'`) or left (`'l'`); `'none'` by default
`theme`	palette of pitch background and lines, either `light` (default), `dark`, `grey`, or `grass`;

`title, subtitle`
: adds title and subtitle to plot; NULL by default

`data`
: a default dataset for plotting in subsequent layers; NULL by default

Value

a ggplot object

See Also

`soccerShotmap` for plotting a shotmap on a half pitch for a single player or `soccerPitch` for drawing a full size soccer pitch

Examples

```
library(ggplot2)
library(dplyr)
data(statsbomb)

# normalise data, get non-penalty shots for France,
# add boolean variable 'goal' for plotting
my_df <- statsbomb %>%
  soccerTransform(method = 'statsbomb') %>%
  filter(team.name == "France" &
         type.name == "Shot" &
         shot.type.name != 'penalty') %>%
  mutate(goal = as.factor(if_else(shot.outcome.name == "Goal", 1, 0)))

soccerPitchHalf(data = my_df, theme = 'light') +
  geom_point(aes(x = location.y, y = location.x,
                 size = shot.statsbomb_xg, colour = goal),
             alpha = 0.7)
```

---

soccerPositionMap *Plot average player position using any event or tracking data*

Description

Draws the average x,y-positions of each player from one or both teams on a soccer pitch.

Usage

```
soccerPositionMap(
  df,
  lengthPitch = 105,
  widthPitch = 68,
  fill1 = "red",
  col1 = NULL,
  fill2 = "blue",
  col2 = NULL,
  labelCol = "black",
  homeTeam = NULL,
```

```
  flipAwayTeam = TRUE,
  label = c("name", "number", "none"),
  labelBox = TRUE,
  shortNames = TRUE,
  nodeSize = 5,
  labelSize = 4,
  arrow = c("none", "r", "l"),
  theme = c("light", "dark", "grey", "grass"),
  title = NULL,
  subtitle = NULL,
  source = c("manual", "statsbomb"),
  x = "x",
  y = "y",
  id = "player_id",
  name = "player_name",
  team = "team_name"
)
```

Arguments

df	a dataframe containing x,y-coordinates of player position and a player identifier variable
lengthPitch, widthPitch	numeric, length and width of pitch in metres
fill1, fill2	character, fill colour of position points of team 1, team 2 (team 2 NULL by default)
col1, col2	character, border colour of position points of team 1, team 2 (team 2 NULL by default)
labelCol	character, label text colour
homeTeam	if df contains two teams, the name of the home team to be displayed on the left hand side of the pitch (i.e. attacking from left to right). If NULL, infers home team as the team of the first event in df.
flipAwayTeam	flip x,y-coordinates of away team so attacking from right to left
label	type of label to draw, player names (name), jersey numbers (number), or none
labelBox	add box around label text
shortNames	shorten player names to display last name as label
nodeSize	numeric, size of position points
labelSize	numeric, size of labels
arrow	optional, adds team direction of play arrow as right ('r') or left ('l')
theme	draws a light, dark, grey, or grass coloured pitch
title, subtitle	optional, adds title and subtitle to plot
source	if statsbomb, uses StatsBomb definitions of required variable names (i.e. 'location.x', 'location.y', 'player.id', 'team.name'); if manual (default), respects variable names defined in function arguments x, y, id, name, and team.
x, y, id, name, team	names of variables containing x,y-coordinates, unique player ids, player names, and team names, respectively; name and team NULL by default

Examples

```
library(dplyr)
data(statsbomb)

# average player position from tracking data for one team
# w/ jersey numbers labelled
data(tromso)
tromso %>%
  soccerPositionMap(label = "number", id ="id",
                    labelCol = "white", nodeSize = 8,
                    arrow = "r", theme = "grass",
                    title = "Tromso IL (vs. Stromsgodset, 3rd Nov 2013)",
                    subtitle = "Average player position (1' - 16')")

# transform x,y-coords, standarise column names,
# average pass position for one team using 'statsbomb' method
# w/ player name as labels
statsbomb %>%
  soccerTransform(method='statsbomb') %>%
  filter(type.name == "Pass" & team.name == "France" & period == 1) %>%
  soccerPositionMap(source = "statsbomb",
                    fill1 = "blue", arrow = "r", theme = "light",
                    title = "France (vs Argentina, 30th June 2018)",
                    subtitle = "Average pass position (1' - 45')")

# transform x,y-coords, standarise column names,
# average pass position for two teams using 'manual' method
# w/ player names labelled
statsbomb %>%
  soccerTransform(method='statsbomb') %>%
  soccerStandardiseCols(method='statsbomb') %>%
  filter(event_name == "Pass" & period == 1) %>%
  soccerPositionMap(fill1 = "lightblue", fill2 = "blue",
                    title = "Argentina vs France, 30th June 2018",
                    subtitle = "Average pass position (1' - 45')")
```

`soccerResample` *Resample the frames per second of any tracking data using linear interpolation*

Description

Downsample or upsample any tracking data containing x,y,t data using linear interpolation of x,y-coordinates (plus constant interpolation of all other variables in dataframe)

Usage

```
soccerResample(df, r = 10, x = "x", y = "y", t = "t", id = "id")
```

Arguments

`df`	a dataframe containing x,y-coordinates and time variable

r	resampling rate in frames per second
x, y	name of variables containing x,y-coordinates
t	name of variable containing time data
id	name of variable containing player identifier

Value

a dataframe with interpolated rows added

Examples

```
data(tromso)

# resample tromso dataset from ~21 fps to 10 fps
soccerResample(tromso, r=10)
```

---

soccerShortenName *Extract player surname*

Description

Helper function to extract last name (including common nobiliary particles) from full player names

Usage

```
soccerShortenName(names)
```

Arguments

names	vector of strings containing full player name

Examples

```
data(statsbomb)
statsbomb$name <- soccerShortenName(statsbomb$player.name)
```

---

soccerShotmap *Draw an individual, team, or two team shotmap using StatsBomb data*

Description

If df contains two teams, draws a shotmap of each team at either end of a full pitch. If df contains one or more players from a single team, draws a vertical half pitch. Currently only works with StatsBomb data but compatability with other (non-StatsBomb) shot data will be added soon.

Usage

```
soccerShotmap(
  df,
  lengthPitch = 105,
  widthPitch = 68,
  homeTeam = NULL,
  adj = TRUE,
  n_players = 0,
  size_lim = c(2, 15),
  title = NULL,
  subtitle = NULL,
  theme = c("light", "dark", "grey", "grass")
)
```

Arguments

df	dataframe containing x,y-coordinates of player passes
lengthPitch, widthPitch	length and width of pitch, in metres
homeTeam	if df contains two teams, the name of the home team to be displayed on the left hand side of the pitch. If NULL, infers home team as the team of the first event in df.
adj	adjust xG using conditional probability to account for multiple shots per possession
n_players	number of highest xG players to display
size_lim	minimum and maximum size of points, c(min,max)
title, subtitle	optional, adds title and subtitle to half pitch plot. Title defaults to scoreline and team identity when two teams are defined in df.
theme	draws a light, dark, grey, or grass coloured pitch with appropriate point colours

Value

a ggplot object

Examples

```
data(statsbomb)

# shot map of two teams on full pitch
statsbomb %>%
  soccerTransform(method='statsbomb') %>%
  soccerShotmap(theme = "gray")

# shot map of one player on half pitch
statsbomb %>%
  dplyr::filter(player.name == "Antoine Griezmann") %>%
  soccerTransform(method='statsbomb') %>%
  soccerShotmap(theme = "grass",
                title = "Antoine Griezmann",
                subtitle = "vs. Argentina, World Cup 2018")
```

soccerSpokes *Draw spokes of passing direction on a soccer pitch*

Description

Multiple arrows to show the distribution of pass angle and distance in zones of the pitch; similar to a radar plot but grouped by pitch location rather than player

Usage

```
soccerSpokes(
  df,
  lengthPitch = 105,
  widthPitch = 68,
  xBins = 5,
  yBins = NULL,
  angleBins = 8,
  x = "x",
  y = "y",
  angle = "angle",
  minLength = 0.6,
  minAlpha = 0.5,
  minWidth = 0.5,
  col = "black",
  legend = TRUE,
  arrow = c("none", "r", "l"),
  title = NULL,
  subtitle = NULL,
  theme = c("light", "dark", "grey", "grass"),
  plot = NULL
)
```

Arguments

df	a dataframe of event data containing fields of start x,y-coordinates, pass distance, and pass angle
lengthPitch, widthPitch	numeric, length and width of pitch in metres
xBins, yBins	integer, the number of horizontal (length-wise) and vertical (width-wise) bins the soccer pitch is to be divided up into; if yBins is NULL (default), it will take the value of xBins
angleBins	integer, the number of arrows to draw in each zone of the pitch; for example, a value of 4 clusters has direction vectors up, down, left, and right
x, y, angle	names of variables containing pass start x,y-coordinates and angle
minLength	numeric, ratio between size of shortest arrow and longest arrow depending on number of events
minAlpha, minWidth	numeric, minimum alpha and line width of arrows drawn
col	colour of arrows

legend	if TRUE, adds legend for arrow transparency
arrow	adds team direction of play arrow as right ('r') or left ('l'); 'none' by default
title, subtitle	adds title and subtitle to plot; NULL by default
theme	palette of pitch background and lines, either light (default), dark, grey, or grass
plot	base plot to add path layer to; NULL by default

Value

a ggplot object of a heatmap on a soccer pitch

See Also

soccerHeatmap for drawing a heatmap of player position, or soccerFlow for drawing a single arrow for pass distance and angle per pitch zone.

Examples

```
library(dplyr)
data(statsbomb)

# transform x,y-coords, filter only France pass events,
# draw flow field showing mean angle, distance of passes per pitch zone
statsbomb %>%
  soccerTransform(method = 'statsbomb') %>%
  filter(team.name == "France" & type.name == "Pass") %>%
  soccerSpokes(xBins=7, yBins=5, angleBins=12, legend=FALSE)

# transform x,y-coords, standarise column names,
# filter only France pass events
my_df <- statsbomb %>%
 soccerTransform(method = 'statsbomb') %>%
 soccerStandardiseCols(method = 'statsbomb') %>%
  filter(team_name == "France" & event_name == "Pass")

# overlay flow field onto heatmap showing proportion of team passes per pitch zone
soccerHeatmap(my_df, xBins=7, yBins=5,
              title = "France passing radar") %>%
  soccerSpokes(my_df, xBins=7, yBins=5, angleBins=8, legend=FALSE, plot = .)
```

---

soccerStandardiseCols *Rename columns in a dataframe for easier use with other soccermatics functions*

Description

Rename columns (e.g. "location.x" -> "x", "team.name" -> "team", etc...) to interface directly with other soccermatics functions without having to explicitly define column names as arguments. Currently only supports Statsbomb data.

Usage

```
soccerStandardiseCols(df, method = c("statsbomb"))
```

Arguments

df	a dataframe of Statsbomb event data
method	source of data; only "statsbomb" currently supported

Value

a dataframe with column names x, y, distance, angle, player_id, player_name, team_name, event_name

Examples

```
library(dplyr)
data(statsbomb)

# transform x,y-coords, standardise column names
my_df <- statsbomb %>%
  soccerTransform(method = 'statsbomb') %>%
  soccerStandardiseCols(method = 'statsbomb')

# feed to other functions without defining variables,
# x, y, id,distance, angle, etc...
soccerHeatmap(my_df)
```

---

soccerTransform *Normalises x,y-coordinates to metres units for use with soccermatics functions*

Description

Normalise x,y-coordinates from between arbitrary limits to metre units bounded by [0 < "x" < "pitchLength", 0 < "y" < "pitchWidth"]

Usage

```
soccerTransform(
  df,
  xMin,
  xMax,
  yMin,
  yMax,
  lengthPitch = 105,
  widthPitch = 68,
  method = c("manual", "statsbomb", "opta", "chyronhego", "ch"),
  x = "x",
  y = "y"
)
```

Arguments

df	dataframe containing arbitrary x,y-coordinates
xMin, xMax, yMin, yMax	range of possible x,y-coordinates in the raw dataframe
lengthPitch, widthPitch	length, width of pitch in metres
method	source of data, either "opta", "statsbomb", "chyronhego" (ChryonHego), or "manual"
x, y	variable names of x,y-coordinates. Not required when method other than "manual" is defined; defaults to "x" and "y" if manual.

Value

a dataframe

Examples

```
# Three examples with true pitch dimensions (in metres):
lengthPitch <- 105
widthPitch <- 68

# Example 1. Opta ----------------------------------------------------------

# limits = [0 < x < 100, 0 < y < 100]
opta_df <- data.frame(team_id = as.factor(c(1, 1, 1, 2, 2)),
                      x = c(50.0, 41.2, 44.4, 78.6, 76.7),
                      y = c(50.0, 55.8, 47.5, 55.1, 45.5),
                      endx = c(42.9, 40.2, 78.0, 80.5, 72.4),
                      endy = c(57.6, 47.2, 55.6, 48.1, 26.3))

soccerTransform(opta_df, method = "opta")


# Example 2. StatsBomb -----------------------------------------------------

# limits = [0 < x < 120, 0 < y < 80]
data(statsbomb)
soccerTransform(statsbomb, method = "statsbomb")


# Example 3. ChyronHego ----------------------------------------------------

# limits = [-5250 < x < 5250, -3400 < y < 3400]

xMin <- -5250
xMax <- 5250
yMin <- -3400
yMax <- 3400

ch_df <- data.frame(x = c(0,-452,-982,-1099,-1586,-2088,-2422,-2999,-3200,-3857),
                        y = c(0,150,300,550,820,915,750,620,400,264))

soccerTransform(ch_df, -5250, 5250, -3400, 3400, method = "chyronhego")
```

```
# Example 4. Manual ----------------------------------------------------

# limits = [0 < x < 420, -136 < y < 136]

my_df <- data.frame(team = as.factor(c(1, 1, 1, 2, 2)),
                    my_x = c(210, 173, 187, 330, 322),
                    my_y = c(0, 16, -7, 14, -12),
                    my_endx = c(180, 169, 328, 338, 304),
                    my_endy = c(21, -8, 15, -5, -65))

soccerTransform(my_df, 0, 420, -136, 136, x = c("my_x", "my_endx"), y = c("my_y", "my_endy"))
```

`soccerVelocity` *Compute instantaneous distance, speed and direction from x,y-coordinates*

Description

Compute instantaneous distance moved (in metres), speed (in metres per second), and direction (in radians) between subsequent frames in a dataframe of x,y-coordinates.

Usage

```
soccerVelocity(dat)
```

Arguments

`dat`	dataframe containing unnormalised x,y-coordinates `x` and `y`, time variable `'t'`, and player identifier `'id'`

Value

a dataframe with columns `'dist'`, `'speed'`, and `'direction'` added

Examples

```
data(tromso)

# calculate distance, speed, and direction for \code{tromso} dataset
soccerVelocity(tromso)
```

soccerxGTimeline *Draw a timeline showing cumulative expected goals (xG) over the course of a match using StatsBomb data.*

Description

Draw a timeline showing cumulative expected goals (xG, excluding penalties and own goals) by two teams over the course of a match, as well as plotting the scoreline and goalscorer at goal events. Currently only works with StatsBomb data but compatability with other (non-StatsBomb) shot data will be added soon.

Usage

```
soccerxGTimeline(
  df,
  homeCol = "red",
  awayCol = "blue",
  adj = TRUE,
  labels = TRUE,
  y_buffer = 0.3
)
```

Arguments

df	a dataframe containing StatsBomb data from one full match
homeCol, awayCol	colours of the home and away team, respectively
adj	adjust xG using conditional probability to account for multiple shots per possession
labels	include scoreline and goalscorer labels for goals
y_buffer	vertical space to add at the top of the y-axis (a quick and dirty way to ensure text annotations are not cropped).

Value

a ggplot object

Examples

```
library(dplyr)
data(statsbomb)

# xG timeline of France vs. Argentina
# w/ goalscorer labels, adjusted xG data
statsbomb %>%
  soccerxGTimeline(homeCol = "blue", awayCol = "lightblue", y_buffer = 0.4)

# no goalscorer labels, raw xG data
statsbomb %>%
  soccerxGTimeline(homeCol = "blue", awayCol = "lightblue", adj = FALSE)
```

`statsbomb` *Sample StatsBomb event data containing the x,y-locations and identity of players involved in pass events, shot events, defensive actions, and more.*

Description

Sample StatsBomb event data from the France vs. Argentina World Cup 2018 game on the 30th June 2018, made publicly available by StatsBomb here. Data contains 145 variables in total, including x,y-coordinates (`location.x`, `location.y`). StatsBomb pitch dimensions are 120m long and 80m wide, meaning `lengthPitch` should be specified as `120` and `widthPitch` as `80`. Event data for all World Cup games (and other competitions) are accessible via the StatsBombR package available here.

Usage

```
data(statsbomb)
```

Format

A dataframe containing 12000 frames of x,y-coordinates and timestamps from 11 players.

Source

ZXY Sport Tracking

References

StatsBomb Open Data

`tromso` *x,y-coordinates of 11 soccer players over 12000 frames each*

Description

x,y-coordinates of 11 soccer players over 10 minutes (Tromsø IL vs. Anzhi, 2013-11-07), captured at 20 Hz using the ZXY Sport Tracking system and made available in the publication ZXY Sport Tracking.

Usage

```
data(tromso)
```

Format

A dataframe containing 12000 frames of x,y-coordinates and timestamps from 11 players.

Source

ZXY Sport Tracking

References

Pettersen et al. (2014) Proceedings of the International Conference on Multimedia Systems (MMSys)

`tromso_extra` *x,y-coordinates and additional positional information on 11 soccer players over 12000 frames each*

Description

x,y-coordinates of 11 soccer players over 10 minutes (Tromsø IL vs. Anzhi, 2013-11-07), plus additional information on player heading, direction, energy, speed, and total distance. Data captured at 20 Hz using the ZXY Sport Tracking system and made available in the publication ZXY Sport Tracking.

Usage

```
data(tromso_extra)
```

Format

A dataframe containing 12000 frames of x,y-coordinates and timestamps from 11 players.

Source

ZXY Sport Tracking

References

Pettersen et al. (2014) Proceedings of the International Conference on Multimedia Systems (MMSys) (pdf)

Index