Package 'mtconnectR'

January 7, 2019

Type Package

Title Read Data from Delimited 'MTConnect' Data Files and Perform some
Analysis

Version 1.2.1

Date 2019-01-03

Encoding UTF-8

Maintainer Ananthapadmanabhan P <ananthu@govimana.com>

Description Read data in the 'MTConnect'<http://www.mtconnect.org/> standard.
You can use the package to read data from historical 'MTConnect logs' along
with the 'devices.xml' describing
the device. The data is organised into a 'MTConnectDevice' S4 data structure
and some convenience methods are also provided for basic read/view operations.
The package also includes some functions for analysis of 'MTConnect' data. This includes
functions to simulate data (primarily position data, feed rate and velocities)
based on the G code and visualisation functions to compare the actual and simulated data.

Depends R (>= 3.0.0)

Imports methods, data.table (>= 1.9.6), XML (>= 3.98), stringr (>=
1.0), plyr (>= 1.8.3), dplyr (>= 0.5.0), ggplot2, magrittr,
lubridate, tidyr, dtw, proxy

License AGPL-3

LazyData TRUE

Collate common.R MTCDataItem.R merge.R MTCCycle.R MTCDevice.R
deviceXML.R dataExtraction_mtc.R map_gcode_mtc.R parse_gcode.R
simulate_gcode.R zzz.R mtconnectR.R

RoxygenNote 6.1.1

Suggests knitr, rmarkdown, testthat, roxygen2

BuildVignettes yes

VignetteBuilder knitr

NeedsCompilation no

Author Alex Joseph [aut],
          Sainath Adapa [aut],
          Nitin Madasu [aut],
          Ananthapadmanabhan P [aut, cre],
          Devaraj Phukan [aut],
          Subramanyam Ravishankar [aut]

R **topics documented:**

Index   **29**

---

add_data_item_to_mtc_device

Add a new data item to an existing MTC Device Class

Description

Add a new data item to an existing MTC Device Class

Usage

```
add_data_item_to_mtc_device(mtc_device, data_item_data, data_item_name,
  category = "EVENT")
```

Arguments

mtc_device	An existing object of MTCDevice Class
data_item_data	Data for the new data item to add
data_item_name	Name of the new data item
category	Category of the new data item. Can be EVENT or SAMPLE

Examples

```
data_item_data = data.frame(timestamp = as.POSIXct(c(0.5, 1, 1.008, 1.011) +
                                   1445579573,  tz = 'CST6CDT', origin = "1970-01-01"),
                         value = c("a", "b", "c", "d"))
data("example_mtc_device")
mtc_device_updated =
   add_data_item_to_mtc_device(example_mtc_device, data_item_data,
    data_item_name = "test", category = "EVENT")
print(mtc_device_updated)
```

---

calculated_feed_from_position

Calculate feed rate from the path position data items

Description

Returns a data.frame which contains the calculated feed rates and the corresponding time stamps

Usage

```
calculated_feed_from_position(mtc_device, pattern = "PATH_POSITION")
```

Arguments

mtc_device	is the MTCDevice object
pattern	is the pattern of the path postion data items in the device object

Examples

```
data("example_mtc_device_3")
calculated_feed_from_postion = calculated_feed_from_position(example_mtc_device_3)
```

---

clean_reduntant_rows   *Removes Redundant Rows in a data frame assuming statefulness*

Description

Removes Redundant Rows in a data frame assuming statefulness

Usage

```
clean_reduntant_rows(df, clean_colname = "value", echo = F,
  clean_na = F)
```

Arguments

df	Data.frame in timestamp, value1, value2,...
clean_colname	Name of the column to clean as basis
echo	Whether to return messages or not
clean_na	Whether to clean NA's when they are redundant

Examples

```
test_interval =
  data.frame(timestamp = as.POSIXct(c(0.5, 1, 1.008, 1.011), origin = "1970-01-01"),
           x     = c("a", "b", "b", "b"),
           y      = c("e", "e", "e", "f"))
clean_reduntant_rows(test_interval, "x")
```

---

convert_interval_to_ts

Convert Interval to Time Series

Description

Basically reverse the effect of [convert_ts_to_interval](#). Column names should be same as mentioned in the example

Usage

```
convert_interval_to_ts(df, time_colname = "start", end_colname = "end",
  remove_last = F)
```

Arguments

df	Data.frame in start, end, duration, value1, value2,...
time_colname	Name of the time column
end_colname	Name of the end time column
remove_last	Logical value to remove the last row in the result

See Also

[convert_ts_to_interval](#)

Examples

```
test_interval =
 data.frame(start = as.POSIXct(c(0.5, 1, 1.008, 1.011), tz = 'CST6CDT', origin = "1970-01-01"),
          end  = as.POSIXct(c(1, 1.008, 1.011, 2), tz = 'CST6CDT', origin = "1970-01-01"),
            duration = c(0.50, 0.01, 0.00, 0.99),
            y     = c("e", "e", "e", "f"))
convert_interval_to_ts(test_interval)
```

---

convert_ts_to_interval

Convert Time Series to Intervals

Description

Function to convert a continuous time series data to interval data. The last row which goes to infinity can be deleted, else will be given dump value.

Usage

```
convert_ts_to_interval(df, endtime_lastrow = as.POSIXct(NA),
  arrange_cols = T, time_colname = "timestamp", round_duration = 6)
```

Arguments

df	A data frame with continuous time series data
endtime_lastrow	
	POSIXct value for the last row. Defaults to NA
arrange_cols	Whether to add the interval and duration columns at the front or not
time_colname	Column name of the timestamp variable
round_duration	Number of decimals to rounds the duration to. Defaults to 2. If no rounding required, give NULL.

See Also

[convert_interval_to_ts](#)

Examples

```
ts_data = data.frame(ts = as.POSIXct(c(0.5, 1, 1.008, 1.011), tz = 'UTC', origin = "1970-01-01"),
                     x = c("a", "b", "c", "d"), y = c("e", "e", "e", "f"))
convert_ts_to_interval(ts_data, time_colname = "ts", endtime_lastrow = ts_data$ts[1] + 10)
```

```
create_mtc_device_from_dmtcd
```
Create MTCDevice class from Delimited MTC Data and log file

Description

Create MTCDevice class from Delimited MTC Data and log file

Usage

```
create_mtc_device_from_dmtcd(file_path_dmtcd, file_path_xml, device_name,
  mtconnect_version = NULL)
```

Arguments

file_path_dmtcd	
	Path to Delimited MTC Data file
file_path_xml	Path to the XML file
device_name	Name of the device in the xml. List of all the devices and their names can be got using the [get_device_info_from_xml](#) function
mtconnect_version	
	Specify MTConnect Version manually. If not specified, it is inferred automatically from the data.

Examples

```
file_path_dmtcd = "testdata/dataExtraction/test_dmtcd.log"
file_path_xml = "testdata/dataExtraction/test_devices.xml"
device_name = "test_device"
mtc_device = create_mtc_device_from_dmtcd(
  system.file(file_path_dmtcd, package = "mtconnectR"),
  system.file(file_path_xml, package = "mtconnectR"),
  device_name)
print(summary(mtc_device))
```

---

create_mtc_device_from_ts

Create a MTC device object from a merged time series data frame

---

Description

Create a MTC device object from a merged time series data frame

Usage

```
create_mtc_device_from_ts(merged_device, device_uuid = "unmerged_device")
```

Arguments

merged_device	An existing object of MTCDevice Class
device_uuid	UUID to be given to the device

Examples

```
data("example_mtc_device")
merged_device = merge(example_mtc_device)
create_mtc_device_from_ts(merged_device)
```

---

example_dmtcd   *Example data set showing MTC Log data*

Description

A manually created dataset showing a log data file, parsed and read into R. The columns are

- timestamp. Timestamp of the event
- data_item_name. Name of the data Item from the delimited MTC data. Can be empty.
- value. of the data item

Usage

```
example_dmtcd
```

Format

A data frame with some rows and 3 variables

---

example_gcode_parsed   *Example data set showing parsed G code data*

Description

A manually created dataset showing a raw gcode data file, parsed and read into R. The columns are

- line Line number
- single_block A single block of G code from a line
- value Value of the data item corresponding to the command
- priority Priority of the block as per the pre-written dictionary
- prefix Prefix of the block
- type Type
- subtype Subtype
- supported Whether the specific G code block is supported or not by the dictionary

Usage

```
example_gcode_parsed
```

Format

A data frame with some rows and 8 variables

---

example_mapped_plot   *ggplot object showing mapping between simulated and actual time series*

Description

ggplot object showing mapping between simulated and actual time series

Usage

```
example_mapped_plot
```

Format

An ggplot object

---

example_mtc_data_item   *Example data set showing a MTConnect DataItem*

Description

The data can be accessed using the @ function. The slots are:

- data Data for a single data item in a data.frame in timestamp, value format
- data_type Type of Data - can be event or sample
- path XML Xpath
- data_source Source from which the data item was created
- xmlID ID of the data item in the devices XML

Usage

```
example_mtc_data_item
```

Format

An MTCDevice data item

---

example_mtc_device   *Example data set showing a MTConnect Device*

Description

The data can be accessed using the @ function. The slots are:

- rawdata Original delimited MTC data (parsed from the file using which the data was created)
- metadata Metadata (if any) for the device
- data_item_list Processed data showing each data item as a separate device
- device_uuid UUID of the device

Usage

```
example_mtc_device
```

Format

An MTCDevice data item

---

example_mtc_device_2   *A bigger example data set showing a MTConnect Device with path position and conditions*

Description

The data can be accessed using the @ function. The slots are:

- rawdata Original delimited MTC data (parsed from which the data was created)
- metadata Metadata (if any) for the device
- data_item_list Processed data showing each data item as a separate device
- device_uuid UUID of the device

Usage

```
example_mtc_device_2
```

Format

An MTCDevice data item

---

example_mtc_device_3   *Example data set showing a MTConnect Device*

Description

The data can be accessed using the @ function. The slots are:

- rawdata Original delimited MTC data (parsed from the file using which the data was created)
- metadata Metadata (if any) for the device
- data_item_list Processed data showing each data item as a separate device
- device_uuid UUID of the device

Usage

```
example_mtc_device_3
```

Format

An MTCDevice data item

```
example_mtc_device_sim
```
MTCDevice object showing simulated G code data

Description

MTCDevice object created using the simulate_gcode function using parsed G code and convert_mtc_device_from_ts to convert data.frame to MTCDevice object.

Usage

```
example_mtc_device_sim
```

Format

A data frame with some rows and 13 variables

```
example_mtc_sim_mapped
```
MTCDevice object containing actual and simulated data and the mapping

Description

MTCDevice object containing actual and simulated data and the mapping

Usage

```
example_mtc_sim_mapped
```

Format

An MTCDevice object

---

example_parsed_device_xml

Example dataset showing the parsed xml for a device

Description

The data can be accessed using the @ function. The slots are:

- parsed_xml Raw XML
- device_details Name,uuid and id of the device
- mtconnect_version

Usage

example_parsed_device_xml

Format

An MTCDevice data item

---

example_simulated_gcode_data

Example data set showing simulated G code data

Description

Dataset created using the simulate_gcode function using parsed G code. The columns are

- timestamp Simulated timestamp
- lineid Line ID
- program Program name
- tool_id Tool ID
- pfr Simulated path feed rate
- rot_vel Simulated rotational velocity
- x_pos Simulated X axis position
- y_pos Simulated Y axis position
- z_pos Simulated Z axis position
- x_vel Simulated X axis velocity
- y_vel Simulated Y axis velocity
- z_vel Simulated Z axis velocity
- state_upcoming_tool State upcoming tool

Usage

```
example_simulated_gcode_data
```

Format

A data frame with some rows and 13 variables

---

example_xpath_info   *Example data set showing Xpaths from a device XML*

Description

Dataset showing a parsed DeviceXML file showing all the XPaths and the properties

- id ID of the data item
- name Name of the data Item from the delimited MTC data. Can be empty.
- type MTC Type of the data item
- category MTC Category of the data item
- subType MTC subType of the data item. Can be emoty
- xpath xpath showing the truncated path to the particular data item in the device XML

Usage

```
example_xpath_info
```

Format

A data frame with some rows and 6 variables

---

extract_param_from_xpath
                              Extract different parts of a xpath

---

Description

Returns a single parameter extracted from the xpath vector. It could be Data Item Name or Data Item type or name of the Device. If the character vector is not in xpath format, the original name is returned and a warning is given

Usage

```
extract_param_from_xpath(strName, param = "DIName", removeExtended = F,
  show_warnings = T)
```

Arguments

`strName`	is the xpath string
`param`	is the parameter to be extracted. Can be "DIName", "DIType" or "Device"
`removeExtended`	if True, then the x: prefix is removed from extended JSON class Types
`show_warnings`	if false, silences the warnings

Examples

```
xpaths = c("timestamp",
 "nist_testbed_Mazak_QT_1<Device>:avail<AVAILABILITY>",
 "nist_testbed_Mazak_QT_1<Device>:execution<EXECUTION>",
 "nist_testbed_Mazak_QT_1<Device>:Fovr<x:PATH_FEEDRATE-OVERRIDE>")

extract_param_from_xpath(xpaths, "DIName")
extract_param_from_xpath(xpaths, "DIType")
extract_param_from_xpath(xpaths, "DIType", TRUE)
extract_param_from_xpath(xpaths, "Device")
```

`filter_timestamps_mtc_device`
          Filter MTCDevice object based on time range

Description

Helper function to quickly filter based on time range

Usage

```
filter_timestamps_mtc_device(mtc_device, start_time, end_time)
```

Arguments

`mtc_device`	is the MTCDevice objectcc
`start_time`	is the Start time
`end_time`	is the End time

Examples

```
data("example_mtc_device_3")
start_time = as.POSIXct("2016-03-22 12:45:00.000")
end_time = as.POSIXct("2016-03-22 12:45:10.000")
filtered_data = filter_timestamps_mtc_device(example_mtc_device_3,start_time,end_time)
```

---

getData   *Get data from the object in a data frame form*

Description

Get data from the object in a data frame form

Usage

```
getData(.Object)
```

Arguments

.Object   A MTC Object

Examples

```
data("example_mtc_data_item")
getData(example_mtc_data_item)
```

---

getData,MTCCycle-method
                    Get Data from MTCDevice/MTCCycle Object as a data.frame

---

Description

Get Data from MTCDevice/MTCCycle Object as a data.frame

Usage

```
## S4 method for signature 'MTCCycle'
getData(.Object)
```

Arguments

.Object   Object of MTCCycle or MTCDevice Class

Examples

```
data("example_mtc_device")
getData(example_mtc_device)
```

---

getData,MTCDataItem-method
Get data from the object in a data frame form

Description

Get data from the object in a data frame form

Usage

```
## S4 method for signature 'MTCDataItem'
getData(.Object)
```

Arguments

.Object   A MTC Object

Examples

```
data("example_mtc_data_item")
getData(example_mtc_data_item)
```

---

getDataItem   *Get data on one or more data items from the class*

Description

Get data on one or more data items from the class

Usage

```
getDataItem(.Object, pattern)
```

Arguments

.Object   object of MTCCycle or MTCDevice Class

pattern   OPTIONAL can be used to query specific data items

Examples

```
data("example_mtc_device")
getDataItem(example_mtc_device)
```

```
getDataItem,MTCCycle,ANY-method
```
                              Get the first dataitem

Description

Get the first dataitem

Usage

```
## S4 method for signature 'MTCCycle,ANY'
getDataItem(.Object)
```

Arguments

.Object   Object of MTCCycle or MTCDevice Class

Examples

```
data("example_mtc_device")
getDataItem(example_mtc_device)
```

```
getDataItem,MTCCycle,character-method
```
                              Get one or more data items from the MTCCycle or MTCDevice using
                              a character pattern

Description

Get one or more data items from the MTCCycle or MTCDevice using a character pattern

Usage

```
## S4 method for signature 'MTCCycle,character'
getDataItem(.Object, pattern)
```

Arguments

.Object   Object of MTCCycle or MTCDevice Class

pattern   Regex of the pattern by which the data is queried

Examples

```
data("example_mtc_device")
getDataItem(example_mtc_device, "POSIT")
```

`getDataItem,MTCCycle,numeric-method`

Get one or more data items from the MTCCycle or MTCDevice using a numeric index

---

Description

Get one or more data items from the MTCCycle or MTCDevice using a numeric index

Usage

```
## S4 method for signature 'MTCCycle,numeric'
getDataItem(.Object, pattern)
```

Arguments

.Object	Object of MTCCycle or MTCDevice Class
pattern	Numeric index/indices of the data item to be queried

Examples

```
data("example_mtc_device")
getDataItem(example_mtc_device, 1:2)
```

`getMetaData`   *Get MetaData from the Object as a list*

Description

Get MetaData from the Object as a list

Usage

```
getMetaData(.Object)
```

Arguments

.Object	Object of MTCDataItem Class

Examples

```
data("example_mtc_data_item")
getMetaData(example_mtc_data_item)
```

---

getMetaData,MTCDataItem-method

Get MetaData from the Object as a list

Description

Get MetaData from the Object as a list

Usage

```
## S4 method for signature 'MTCDataItem'
getMetaData(.Object)
```

Arguments

.Object   Object of MTCDataItem Class

Examples

```
data("example_mtc_data_item")
getMetaData(example_mtc_data_item)
```

---

get_device_info_from_xml

Get info on all the devices in the xml file

Description

Device XML usually consists of the configuration details of multiple devices. This function can detail all the device info in the XML into a data.frame for easy reference

Usage

```
get_device_info_from_xml(file_path_xml, mtconnect_version = NULL)
```

Arguments

file_path_xml   Path to the XML file

mtconnect_version

                 Specify MTConnect Version manually. If not specified, it is inferred automatically from the data.

See Also

get_xpaths_from_xml

Examples

```
file_path_xml = "testdata/dataExtraction/test_devices.xml"
devices_info = get_device_info_from_xml(system.file(file_path_xml, package = "mtconnectR"))
print(devices_info)
```

---

get_xpaths_from_xml   *Get XML xpath info*

Description

Get info on all the xpaths for a single device from the xml file. Data is organized into a data.frame

Usage

```
get_xpaths_from_xml(file_path_xml, device_name, mtconnect_version = NULL)
```

Arguments

file_path_xml	Path to the XML file
device_name	Name of the device in the xml. List of all the devices and their names can be got using the [get_device_info_from_xml](#) function
mtconnect_version	
	Specify MTConnect Version manually. If not specified, it is inferred automatically from the data.

Examples

```
file_path_xml = "testdata/dataExtraction/test_devices.xml"
device_name = "test_device"
xpath_info = get_xpaths_from_xml(system.file(file_path_xml, package = "mtconnectR"), device_name)
print(xpath_info)
```

---

grep_subset   *Subset a data frame using regex matching on the column name and also on the value*

Description

Subset a data frame using regex matching on the column name and also on the value

Usage

```
grep_subset(dataFrame, colGrep, subGrep, echo = T, invert = F)
```

Arguments

`dataFrame`	is a data.frame
`colGrep`	is a regex pattern for finding the columns
`subGrep`	is a regex pattern to subset the values in the matched column
`echo`	If TRUE, messages are printed on the console
`invert`	If TRUE, returns everything other than the rows and columns matched using colGrep and subGrep

Examples

```
df = data.frame(type = c("sample","event","condition","sample"),value = c("value1","value2",
                "value3","value4"))
filtered_df = grep_subset(df,"type","sample")
```

---

map_gcode_mtc   *Create a mapping between simulated and actual data*

Description

Creates a timestamp based mapping to map every simulated timestamp to an actual timestamp based on real data

Usage

```
map_gcode_mtc(mtc_device_sim, mtc_device, elasticity = 2)
```

Arguments

`mtc_device_sim`	is the simlated version
`mtc_device`	is the actual log data
`elasticity`	is the maximum consecutive reference elements skippable (passed to dtw::mvmStepPattern())

Examples

```
data("example_gcode_parsed") # Parsed gcode
data("example_mtc_device_3") # MTCDevice object of actual log data
simulated_gcode_data = na.omit(simulate_data_from_gcode(example_gcode_parsed,
start_time = 0, data_res = 0.1, data_type = "HH"))
mtc_device_sim = create_mtc_device_from_ts(simulated_gcode_data)
mtc_sim_mapped = map_gcode_mtc(mtc_device_sim, example_mtc_device_3, elasticity = 200)
```

`merge,MTCCycle,ANY-method`

Merge all data items from the MTCCycle or MTCDevice

Description

Merge all data items from the MTCCycle or MTCDevice

Usage

```
## S4 method for signature 'MTCCycle,ANY'
merge(x)
```

Arguments

x   Object of MTCCycle or MTCDevice Class

Examples

```
data("example_mtc_device")
merge(example_mtc_device)
```

`merge,MTCCycle,character-method`

Merge one or more data items from the MTCCycle or MTCDevice using a character pattern

---

Description

Merge one or more data items from the MTCCycle or MTCDevice using a character pattern

Usage

```
## S4 method for signature 'MTCCycle,character'
merge(x, y)
```

Arguments

x   Object of MTCCycle or MTCDevice Class

y   Regex for picking data items which has to be merged

Examples

```
data("example_mtc_device")
merge(example_mtc_device, "POSIT")
```

---

merge,MTCCycle,numeric-method

Merge one or more data items from the MTCCycle or MTCDevice using an index

---

Description

Merge one or more data items from the MTCCycle or MTCDevice using an index

Usage

```
## S4 method for signature 'MTCCycle,numeric'
merge(x, y)
```

Arguments

x	Object of MTCCycle or MTCDevice Class
y	Numeric index/indices of the data items to be merged

Examples

```
data("example_mtc_device")
merge(example_mtc_device, 1)
```

---

mergeTS   *Merges all the data.frames in the list into single data.frame*

Description

Merges all the data.frames in the list into single data.frame

Usage

```
mergeTS(DF_list, output_DF = T, use_list_names = F,
  additional_ts = .POSIXct(integer(0)), ignore_tz = F)
```

Arguments

DF_list	is a list of data.frames. Each data.frame should be of type timestamp|value1|value2...
output_DF	if TRUE, then returns output in the form of data.frame instead of data.table
use_list_names	if TRUE, the names of the list are assigned the columns names
additional_ts	an POSIXct vector of timestamps which needs to be added into the table. The values are repeated from the previous timestamp
ignore_tz	timezone error is ignored.

Examples

```
data("example_mtc_device")
mtc_device = merge(example_mtc_device)
df_1 = mtc_device[seq(1,17,2),]
df_2 = mtc_device[seq(2,17,2),]
merged_df = mergeTS(list(df_1,df_2))
```

MTCCycle-class   *An S4 class to represent the different data items of a device*

Description

An S4 class to represent the different data items of a device

Slots

data_item_list List of data items with data(data.frame of timestamp,value), data_type,path,dataSource,xmlID

device_uuid UUID of the device

MTCDevice-class   *An S4 class to represent a device. It contains MTCCycle class and the slots below.*

Description

An S4 class to represent a device. It contains MTCCycle class and the slots below.

Slots

rawdata Delimited MTC data (parsed from the file using which the data was created)

metadata Metadata (if any about) the device

Examples

```
data('example_mtc_data_item')
example_mtc_data_item
```

---

mtconnectR	*mtconnectR: A package to read analyze data in the 'MTConnect' standard*

Description

You can use the package to read data from historical 'MTConnect logs' along with the 'devices.xml' describing the device. The data is organised into a 'MTConnectDevice' S4 data structure and some convenience methods are also provided for basic read/view operations. The package also includes some functions for analysis of MTConnect data. This includes functions to simulate data (primarily postion data, feed rate and velocities) based on the G code and visualisation functions to compare the actual and simulated data.

mtconnectR functions are divided into two categories

- Functions to read XML and log data
- Functions read Gcode and simulate data and visualize actual and simulated data of some analysis and visualization functions

---

parse_devicexml_for_a_device

Parse XML file for given device name

Description

Get all the info on a particular device present in the XML Returns a list of parsed_xml,device_details and mtconnect_version

Usage

```
parse_devicexml_for_a_device(file_path_xml, device_name,
  mtconnect_version = NULL)
```

Arguments

file_path_xml   File path of the XML file

device_name   Name of the device to be searched in the XML

mtconnect_version

                Version number of mtconnect standard

Examples

```
file_path_xml   = "testdata/dataExtraction/test_devices.xml"
device_name = "test_device"
parsed_device_xml = parse_devicexml_for_a_device(system.file(file_path_xml,
                                        package = "mtconnectR"),device_name)
```

---

parse_gcode   *Read the gcode and translate it as per the dictionary*

Description

Returns a data frame with each row referring to a block of G-code

Usage

```
parse_gcode(gcode_file_path)
```

Arguments

gcode_file_path
              Directory path of the file containing G-code

Examples

```
gcode_file_path = "extdata/raw_gcode_sample.NC"
gcode_parsed = parse_gcode(system.file(gcode_file_path,package = "mtconnectR"))
```

---

plot_twoway   *To plot the mapping between the simulated and actual versions*

Description

This function is a wrapper around the dtwPlotTwoWay from the dtw package in R.

Usage

```
plot_twoway(mtc_sim_mapped, mtc_device_sim, mtc_device, offset = 100,
  total_maps = 50, mtc_map_string = "path_pos_x",
  sim_map_string = "x_pos")
```

Arguments

mtc_sim_mapped	is the mapping between simulated and actual data
mtc_device_sim	is the simulated data
mtc_device	is the actual log data
offset	is the amount by which the simulated data is shifted in the y axis
total_maps	is the number of mapping line segment between the two plots
mtc_map_string	is the name of the data item to match in the actual data
sim_map_string	is the name of the data item to match in the simulated data

Details

In our case, this function can be used to compare the variation of a single variable in the simulated and actual data. The simulated and actual data are represented on the same y axis with the time representing the x axis. An offset is provided to the simulated data to easily disinguish the two. Data points that are mapped to each other are connected by a line segment between the two fraphs

Examples

```
data("example_mtc_device_3")   # MTCDevice object of actual log data
data("example_mtc_device_sim") # Simulated gcode
data("example_mtc_sim_mapped") # Mapping between simulated and actual data
mapping_ggplot = plot_twoway(example_mtc_sim_mapped, example_mtc_device_sim,
example_mtc_device_3,offset = 20, total_maps = 100)
```

---

read_dmtcd_file   *Function to load Log data into R as a data.frame*

Description

Function to load Log data into R as a data.frame

Usage

```
read_dmtcd_file(file_path_dmtcd, condition_names = c(),
  path_position_names = c())
```

Arguments

file_path_dmtcd

> Path to the file containing log data

condition_names

> A character string with the names of the data items that represent the conditions in the log data

path_position_names

> A character string with the names of the data items that represent the path_position data items

Examples

```
device_name = "test_device"
file_path_xml = "testdata/dataExtraction/test_devices.xml"
xpath_info = get_xpaths_from_xml(system.file(file_path_xml, package = "mtconnectR"), device_name)
```

---

simulate_data_from_gcode

Simulate position,velocity and other data from G-code

Description

Reads parsed gcode and returns simulated data

Usage

```
simulate_data_from_gcode(gcode_parsed, start_time = 0, data_res = 0.2,
  data_type = "ISO")
```

Arguments

gcode_parsed	Parsed gcode
start_time	Starting time (default 0)
data_res	Resolution for occurrence of a new data point. (seconds)
data_type	Data type

Examples

```
data("example_gcode_parsed")
simulated_data_from_gcode <- simulate_data_from_gcode(example_gcode_parsed,start_time = 0,
data_res = 0.2, data_type = "HH")
```

Index