Remixer

# Modes of Use

## Standalone Mode (deprecated)

This mode of use allows to process a single csv file at a time without considering the TOR magnitude (time of reporting) indicated in the csv.

So, each row is processed one at a time, sent to an udp server and then the program sleeps a default time (sleep) of 1000msec before grab the next row.

Simultaneously, binary output data also is written to a parameterizable output file name (binfile). You can choose the udp server destination port (default 2200) and address server (default is 127.0.0.1).

### Line Parameters

*/help*  help message

*/csv=fname* csv input file name. It’s mandatory parameter in this mode of use. Only it has sense in this mode of use. ***It can not be a null value.***

*/src=source* adsb, radar or unknown. Default value is unknown. According to this value, adsbPort, radarPort, truthPort as destination port will be used respectively.

*/binfile=xxx* Output binary file, default value is “output.bin”.

Simultaneously, the binary data sent by udp it is written to a binary output file.

*/adsbport=p* destination port when src=adsb. Default value is 2200.

*/radarport=p* destination port when src=radar. Default value is 2200.

*/truthport=p* destination port when src=unknown. Default value is 2200.

*/server=addr* destination udp server address. Default value is 127.0.0.1

*/sleep=msecs* sleep time between rows in mecs. Only it has sense in this mode. Default value is 1000 msecs.

*/callsign* callsign C string. Default value is “XX\_INTR”

i.e.:

*remixer /csv=C1\_ADSB\_ADSB\_adsbLog\_TVInt1.csv /src=adsb*

*remixer /csv=C1\_TruthInt1.csv /src=unkwnown /truthport=2333 /sleep=10*

### Config file (remixer.ini)

####################### common config #######################

[common]

# used by: trasim\_traffic\_engine\_sub

ownship\_id=H2\_INTR

#out\_bin\_file=output.bin

#callsgin=H2\_INTR

#2017-01-01

#in seconds, 2017-01-01 equivalen to date in traffic.ini, but this is general

toa\_offset=1483228800

outdir= out

alertsDir= alerts

[kml]

#http://www.zonums.com/gmaps/kml\_color/

#KML Colours, the first byt is transparency: 0xff=0% 0x00=100%

# Red ff00ffff.

# Blue ff00ff00.

# Purple ff0080ff.

# Brown ffff00ff.

#

#opacity

opacity=80

#black

ownshipColor = 000000

#white

ownshipAvoidanceColor = ffffff

#yellow

ownshipAlert1Color = 00ffff

#orange

ownshipAlert2Color = 1484ff

#red 5

ownshipAlert3Color = 0000ff

#blue

intruderColor = ff0000

ownshipIcon= http://earth.google.com/images/kml-icons/track-directional/track-0.png

intruderIcon= http://earth.google.com/images/kml-icons/track-directional/track-0.png

linewidht=1

# used by: trasim\_traffic\_engine\_app and trasim\_simtime\_engine\_app

[dds\_config]

config=dds\_common\_config.ini

simtraffic\_domain=1

simtime\_domain=0

simtime\_topic=simtime

simtraffic\_topic=simtraffic

####################### common\_config #######################

####################### trasim\_traffic\_engine\_app.exe #######################

[simtraffic]

#flights=mops\_mixed\_flights2.ini

flights=mops/converging/C1\_flights.ini

####################### trasim\_traffic\_engine\_app.exe #######################

####################### trasim\_simtime\_engine\_app.exe #######################

[simtime]

date=2017-01-01

time=00:00:00

frequency=1

fast\_factor=1

####################### trasim\_simtime\_engine\_app.exe #######################

####################### trasim\_sensors\_app.exe #######################

[sensors]

#server address

ip\_address = 127.0.0.1

#ip\_address = 127.0.0.1

ownship\_port = 2100

adsb\_track\_port = 2200

radar\_track\_port = 2200

camera\_track\_port = 2200

clean\_track\_port= 2300

#adsb\_csv\_int\_file = C1\_ADSB\_ADSB\_adsbLog\_TVInt1.csv

#radar\_csv\_int\_file = C1\_RADAR\_NONE\_radarLog\_TVInt1.csv

truth\_csv\_int\_file = C1\_Truth\_TVInt1.csv

# following parameters define what sensors must be connected

#adsb=on/off indicates that adsb is conected/disconected.

#adsb\_track indicates that the sensor will create a binary file in daa format

#adsd\_disturbances indicates that the internal truth adsb values will be distorted

adsb=on

adsb\_track=on

adsb\_config=adsb\_sensor.json

adsb\_disturbances= on

radar=on

radar\_track=on

radar\_config=radar\_sensor.json

radar\_disturbances= on

camera=on

camera\_track=on

camera\_config=camera\_sensor.json

camera\_disturbances= on

clean\_track\_sensor=on

clean\_track\_sensor\_track=on

clean\_track\_sensor\_config=clean\_track\_sensor.json

nav\_ownship=on

nav\_ownship\_track=on

nav\_ownship\_config=nav\_ownship\_sensor.json

#sensor\_output\_file=mops\_mixed\_flights2.bin

sensor\_output\_file=mops\_convergingC1.bin

[logger]

# Valid values are:

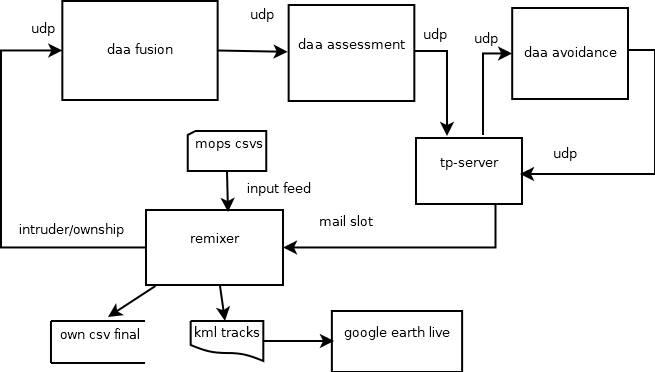
# trace, debug, information, notice, warning, error, critical, fatal, none (turns off logging)

level = trace

file = \_remixer.log

####################### trasim\_sensors\_app.exe #######################

## Remixer Mode (“batidora”)



This mode allows up to process three intruder mops (csv) files (adsb, radar, intruder, truth) and one ownship csv simultaneously in simulated time indexed by the TOR or TOA parameter (when TOR doesn’t exist) indicated in the csv.

In this mode all the csv files (up to 4) are parsed and loaded in memory then them are sorted indexed by the TOR or TOA column and sent by udp to daa-fusion or saved to a binary file.

**WARNING: In order to invoke this mode at least one of these *adsbCsv*, *radarCsv,* *truthCsv* or *ownshipCsv* parameters *must have a not empty value* (not “”).**

Each row can be sent to a different port (*adsbPort*, *radarport*, *truthport* or *ownshipport*) according to the source.

Binary output data also is written to an output file name (binfile) instead of send by udp.

This program creates a mailslot where it will listen messages from tp-server program in order to change ownship track vector with the new recalculated track from tp.

Further the remixer also can work without TPserver mailslot messages feedback, it’s not mandatory. The only difference is that with the tp feedback remexier rewrites the trajectory of the ownship when an avoidance path is generated.

At the end of the simulation a mops csv final ownship, kml ownship final trajectory and alerts report will be generated. In order to generate intruder kml trajectory you can use mops2kml tool.

In “debug” version you have ownship available sent to udp port 2101. Use udpserver app with this porpouse.

Mailslot Messages arrived from TP server could be:

### “STOP”

Stops the simulation in order to wait the csv ownship track file name coming from TPserver. When this message is received remixer don’t update tracks and stops just waiting for the CSV message containing csv trajectory file name.

### “PAUSE”

Pause the simulation in order to wait the csv ownship track file name coming from TPserver. But tracks will continue being updated until new csv file name is received and parsed. So, a few (one or two perhaps) ownship registers could be send in this short break with original data (instead the recalculate track). When CSV\_xxx arrives the new ownship data will be send instead original.

### “CSV\_XXX”

Csv file name received from tp-server in order to reload new ownship trajectory with avoidance path. After parser it the simulation will continue with the new ownship values, discarting original track values from previous TOA. “XXX” is the file name. It should be in the same host.

### “ALERT\_XXX”

Different alert levels received from tp-server coming from daa assessment module. This alerts will be log with ownship/intruder positions (alert report) and time. Also them will be show in “real time” in google ownship kml track color.

### TP-server functionality

This application has basically two main functions (in two different threads). It is mandatory that this application must coexist in the same host with the remixer because of the windows mailslot.

First function is to receive (udp-server, port 4001) avoidance path directly from daa-avoidance app. With this received (6 or 7) points path ***tp.exe*** is called (system call) in order to generate a new csv trajectory (*CSV\_XXX* message containing file name). This csv file name will be send to remixer app using a windows mailslot in order that remixer load it in memory (vector container) discarding old (original) ownship track data.

The second function is to capture (udp packet) alerts directly from daa-assessment app and resend it to daa-avoidance app (udp packet) working as a proxy, in this way these alerts can be send to remixer (using mailslot messages), in order to log them and change ownship track color in google earth acording to the actual alert level.

### Tpserver.ini

* + 1. Config file.

[common]  
tpdir=tp  
googledir=google  
outdir=out  
  
[logger]  
# Valid values are:  
# trace, debug, information, notice, warning, error, critical, fatal, none (turns off logging)  
level = trace  
#file = tpserver.log  
  
[tp]  
trap\_executable = trap\_exec.exe  
#mode can be: udp or mailslot  
#in udp mode avoidance data is received directly from avoidance by udp port 10000  
#in mailslot mode avoidance data is received from avoidance\_suscriber app by mailslot  
  
[avoidance]  
outPort = 10000  
#this 10003 value is for test propose only  
#outPort = 10003  
inPort= 4000  
address = 127.0.0.1  
  
[assessment]  
outPort = 4001  
mode = udp  
#not used  
address = 127.0.0.1  
  
[tp\_xml]  
#default parameters for xml input  
APM\_id=C208  
#masa  
m=2150  
#speed  
tas=40  
#this value is set by program argument because it is very used  
#stage=H14

### Remixer Line Parameters

* + 1. These parameters have priority over those of configuration (*remixer.ini*) file.

*/help*  help message

*/****adsbcsv****=fname* adsb csv input file name, can be an empty value. Not default value.

These rows will be send to *server*:*adsbport*.

*/****radarcsv****=fname* radar csv input file name, can be an empty value. Not default value.

These rows will be send to *server:radarport.*

*/****truthcsv****=fname* intruder truth csv input file name, can be an empty value. Not default value.

These rows will be send to *server:truthport.*

Warning: at least one of above 3 input csv file names should be not empty.

*/****ownshipcsv****=fname* ownship truth csv input file name, can be an empty value. Not default value.

These rows will be send to *server:ownshiport.*

Warning: at least one of above 3 input csv file names should be not empty.

*/****adsbport****=p* adsb destination port. Each row from *adsbcsv* file will be sent to this port. Default value is 2200.

*/****radarport****=p radar* destination port. Each row from *radarcsv* file will be sent to this port. Default value is 2201.

*/****truthport****=p* *truth* destination port. Each row from *truthcsv* file will be sent to this port. Default value is 2300.

*/****ownshipport****=p* *ownship* destination port. Each row from *ownshipcsv* file will be sent to this port. Default value is 2100.

*/****server****=addr* destination udp server address. Default value is 127.0.0.1

*/****callsign***callsign C string. Default value is “XX\_INTR”. The first two chars could be taken in order to set stage (“XX”);

*/****inifile*** configuration file name (.ini). Defauult: “remixer.ini”. See config file.

/***toa\_offset*** toa offset in seconds from 1-1-1970. Default: 1483228800 (2017-1-1)

/**stage** scenary. If not set will be grab from callsign (first two chars).

/***level*** log leve

*/****binfile****=xxx* Output binary file, default value is “output.bin”.

Simultaneously, the binary data sent by udp it is written to a binary output file.

i.e.:

*remixer /adsbcsv=C1\_ADSB\_ADSB\_adsbLog\_TVInt1.csv /radarcsv=C1\_RadarLog.csv /truthcsv=C1\_truthTVint.csv*

### Config file (remixer.ini)

* + 1. Some of these values or all could be set in .ini file (inifile parameter). But if the same parameter is set in line parameter, it has priority.
    2. Config file is organized in sections/key→value.

# [common]

*ownship\_id* callsign parameter, from here stage name would be take.

*toa\_offset* toa offset, this value will be added to csv toa values.

*outdir* output dir, log files will be generated and ownship csv final output file here.

*alertdir* alerts dir, alert reports will be generated here

*googledir* google earth kml files will be generated here.

# [logger]

*level*  log level: : can be none, fatal, critical, error, warning, notice, information, debug, trace. Default: notice.

*file* if not set log will be send to only to console. This file will be created in outdir directory.

# [sensors]

*ownship\_port* to this udp port remixer will send true ownship registers

*adsb\_track\_port* to this udp port remixer will send adsb intruder registers if those exist

*radar\_track\_port* to this udp port remixer will send radar intruder registers if those exist

*camera\_track\_port* to this udp port remixer will send camera intruder registers if those exist

*clean\_track\_port* to this udp port remixer will send ownship true registers if those exist

*ip\_address* ip or udp server name of the before ports.

# [kml]

#http://www.zonums.com/gmaps/kml\_color/

#KML Colours, the first byt is transparency: 0xff=0% 0x00=100%

# Red – 0000ff.

# Yellow – 00ffff.

# Blue – ff0000.

# Green – 00ff00.

# Purple – 800080.

# Orange – 0080ff.

# Brown – 336699.

# Pink – ff00ff.

*opacity* opaqueness, ff is max, 00 is min.

*ownshipColor*  ownship color; track, arrow, labels, etc.

*ownshipAvoidanceColor* ownship color after receive avoidance track

*ownshipAlert1Color* alert 1 color

*ownshipAlert2Color* alert 2 color

*ownshipAlert3Color* alert 3 color

*intruderColor* intruder color track, arrow, labels, etc.

*ownshipIcon* ownship icon, href, usually an arrow.

*IntruderIcon* intruder icon, href, usually an arrow.

*Linewidth track line width*

### Google Earth “live update” Conexion

The remixer app updates every second ownship and intruder kml track files (*intruderTrack.kml* and *ownshipTrack.kml* respectively in “google” directory), in order to see these tracks in “live mode”. It’s mandatory to load *loadIntruderTrack.kml* and *loadOwnshipTrack.kml* files at the beginning ot the simulation in order that google earth application updates both tracks every second. In this way you can see the simulation live over google earth screen. The ownship track color changes in simulated time according to alerts level (configurable in remixer.ini).



Also you can load ownship final trajectory kml file after the simulation in order to analyze tested stage (in google directory). Also you can load intruder mops trajectory, previously converted to kml using csv2kml app tool.

## Batch files

* 1. In order to invoke this batchs all apps (daa, remixer y tp-server) should be in the same windows host. These batch will launch the daa apps (fusion, assessment and avoidance), tp-server (who will call win tp.exe with a system call) and the remixer (“batidora”). Only remixer will be called in foreground. Please set corresponding config files (remixer.ini, tpserver.ini, fussion\_app.ini, assessment\_app.ini and avoidance\_app.ini). Be careful with the udp ports, these must be “chained” because remixer must send udp packets to fusion input, assesment output to tp-server input, tp-server output to avoidance input, and finally avoidance output to tp-server.
  2. Open-dds is not necessary here, because the daa-input and output es udp.

### setdaaenv.bat

Sets daa enviroment (library dependencies)

### stagetest.bat

Test a stage. It needs a stage parameter (i.e. H1)

### testall.bat

Calls stagetest.bat in order to test all available mops stages.