## Trajectory Conversion Algorithm Version 1.90.5

## May 2014

Version 1.90.3, Summary of Modifications

-reconfigured to accommodate multi-vehicle trajectory files

-time-dependent multi-vehicle snapshot generation and transmission

-supports Build 0 of TCA 2.0

Version 1.90.4, Summary of Modifications

-clarified Gap settings and options

-removed second option for Start and Stop Triggers

-write to ALLSNAP at deletion, transmission, vehicle network departure and end-of-simulation

-added clarifying text for the deletion/prioritization of snapshots in the buffer

Version 1.90.5, Summary of Modifications

-re-structured to loop through timesteps instead of each vehicle

-added BSM snapshot generation/transmission and cellular communication

-added Build 2 details such as: dual BSM-PDMs, dual communication, regions, etc.

-generalized ALLSNAP to snapshot log to accommodate PDMs and BSMs

Roadside Equipment (RSE) Location File

 the set of all RSEs deployed in the network

 RSE located at lat/lon position 

 communications range for an RSE (feet), default = 492

 out-of-range distance for an RSE (feet), default = 492

# Vehicle Trajectory File

For each time step we have a record containing vehicle id, position and speed (instantaneous acceleration optional) for every vehicle in motion in the network.

 vehicle located at position  at time  with velocity 

 unique vehicle located within the trajectory file (identifier)

 the set of all unique vehicle IDs in the trajectory file

 the subset of vehicles currently active in the network

Vehicle Relative Position Functions

 cumulative roadway distance traveled (feet) between time and 

 range (feet) from current vehicle location to a particular RSE

Cellular Region Variables

 the set of all cellular regions defined by user

 cellular region name

 cellular latency

 cellular loss rate

 minimum number of PDMs needed to transmit

Regions File

 the set of all user defined event regions

 one region name in the set of regions

 the set of timeperiods the event region is active

 the event name

 the probability type (mean, probability, or Poisson)

 the probability of an event occurrence

 the recheck value of the event region

 the Poisson recheck value

 the next time check value for the event

 the previous check time for the event

 the default value for the event (often FALSE, or numerical such as default temp)

# Probe Message Process Descriptor File

# Snapshot Association Strategy

 PSN (Temporary ID) rollover strategy, select one of:

1. Maximum Time and Distance, parameters: ; (seconds), (miles)
2. Time, parameters: , ID persistence (seconds)
3. Distance:, ID persistence (miles)

Privacy Gap Parameters: can be ON or OFF with any of the three PSN rollover options

Privacy protection gap policy:  ,(default = TRUE) if gaps are “ON”

Maximum gap time and distance, parameters: ; (seconds), (miles)  
time is uniformly distributed between and 

distance is uniformly distributed between and 

(defaults = 3-13 seconds, 164-820 feet)

Gap initiated (time): 

# Start/Stop Snapshot Trigger Mechanism

 Trigger mechanism type, of:

1. Time and Speed, parameters:
   * 1. stop threshold (seconds), 
     2. stop lag (seconds), 
     3. start threshold (mph), 

# Periodic Snapshot Trigger Mechanism

 Trigger mechanism type, of:

1. Speed Interpolation, parameters:
   * 1. low threshold (mph), 
     2. short interval (seconds), 
     3. high threshold (mph), 
     4. long interval (seconds), 
2. Speed Exception, parameters:
   * 1. low threshold (mph), 
     2. max delta (%), 
     3. standard interval (seconds), 

# Event Snapshot Trigger Mechanism

 Event snapshot list or parameters, read in from file:

Area-based with optional randomness

Read in range of coordinates that represent the area for events,

Let be the set of all points within the defined area

Read in the probability of an event occurring in that area (0-1).

Brake Trigger Mechanism

 Brake threshold that defines the deceleration value after which brakes are considered

applied

PDM Buffer Capacity and Management Strategies

 Sub-Buffer Utilization:

1. None, parameters: capacity (snapshots) 
2. Sub-buffer by snapshot type, parameters:
   * 1. Stop/start capacity (snapshots), 
     2. Periodic capacity (snapshots), 

 Snapshot retention prioritization scheme.  
Event snapshots always have highest priority. Start and Stops have second highest priority within the vehicle buffer. Periodic snapshots have the lowest priority.

Options: include keeping most recent (newest) snapshots subject to priority level OR for periodic snapshots, deleting data from the middle of the set of periodic snapshots in the buffer (this keeps the oldest and newest snapshots deletes the “middle” periodic snapshot  
.  
0 = Event, Stop/Start, Periodic with most recent data retained

1 = Event, Stop/Start, Periodic with every other snapshot retained up to SOI

 , the divisor in the Start-Over-Index (SOI) calculation, default = 2

 Snapshot reporting prioritization scheme,  
0 = Event, Stop/Start, Periodic with most recent data retained  
1 = Event, Stop/Start, Periodic with every other snapshot retained

Vehicle/RSE Interaction Policy

 Vehicle/RSE interaction policy, consisting of:

1. Permitted consecutive RSE interactions while in range of an RSE (reports),   
   (default = 1)
2. Minimum number of snapshots in buffer to trigger RSE interaction,   
   (default = 1)
3. Within-range of a particular RSE flag, , 0=false, 1=true

# PRIMARY LOOP:

0.1 **Control and Strategy Parameters Initialization**.

Initialize Control and Strategy parameters using Control file values.

*0.2* **RSE Parameters Initialization**.  
 Initialize RSE parameters using RSE Location File values.

0.3 **Random Variable Initialization.**

Initialize random variables using Control file values.

*0.4* **Vehicle and Snapshot Buffer Initialization**.

Let , the master clock time (one time step before the first time record in the trajectory file)

Let, the set of current vehicles in the network be empty.

*0.5* **Snapshot Output File Initialization.**

Initialize output file locations using Control file values.

*0.6* **Read Trajectory File Record**.  
Read in the entire trajectory file (or VISSIM .fzp file)

1. **Read One Timestep of Vehicles**

FOR EACH Time Period

 : BSM buffer reset each timestep

FOR EACH  : for all active vehicles

 : note we have seen the vehicle this timestep

IF THEN : is this a new vehicle?

GOTO 1.5 : initialize new vehicle parameters

END IF

FOR EACH  : for all vehicles not currently active



GOTO 1.1 : update event regions

*1.1* **Update Event Region Variables**

IF  : are there are user-defined event regions?

FOR EACH  : for all active vehicles

FOR EACH  : for all regions

FOR EACH  : for each event in that region, assign value from probability

GOTO 1.6

*1.2* **Update Brakes Info**

FOR EACH  : for all active vehicles

IF : deceleration is greater than brake threshold (brakes on)



ELSE : brakes are not applied



IF  : if vehicle is decelerating, update the brake pressure



IF  : deceleration is greater than J2735 standard for hard braking



ELSE : vehicle is not in a hard braking state



GOTO 2.0 : check for snapshot generation

*2.0* **Check Snapshot Triggers**.

1. **Check for RSE Interaction**
2. **Check for Cellular Interaction**
3. **Manage PSN and Privacy Gap initiation/expiration**

GOTO 2.0 until no more timesteps in the simulation

*6.0* **Output to Log contents of buffers from vehicles still on the network**

FOR EACH  : for all vehicles still underway

FOR EACH TO  : for each veh-snap log entry

WRITE  : write a single snapshot record

END FOR EACH 

END FOR EACH 

7.0 STOP

*1.5* **New Vehicle Parameter Initialization**:

 : add new vehicle to set currently active vehicles

Initialize strategy approaches and strategy parameters using the process descriptor file.

Initialize vehicle record: 

Vehicle position, x and y

Indicate the vehicle has not departed the network, 

Vehicle seen in network in this time step, 

Vehicle snapshot ID buffer is empty/undefined, 

Snapshot Log for this Vehicle is empty/undefined, 

counter for total snapshots generated by this vehicle, 

counter for number of snapshots in buffer, periodic snapshots in buffer, 

time of next periodic snapshot (seconds), .

number of seconds vehicle has been motionless, .

time last stop recorded (seconds), .

distance traveled since last stop recorded (feet), .

time stamp of current temporary ID (seconds), 

generate a temporary 4-digit random ID, 

we are not looking for a start (flag variable), 

we are not in a privacy blackout gap (flag variable), 

time of the privacy gap start, time-in-gap, distance-in-gap: 

let the last RSE visited be undefined, 

Instantaneous acceleration, 

Average acceleration, 

Brake status, 

Brake pressure, 

Hard braking, 

Set of regions variables (user defined) 

Indicate whether the vehicle is equipped for BSMs, PDMS, or both

Indicate whether the vehicle is equipped to communicate via DSRC, cellular, or both

RETURN

*1.6* **Regions value assignment**

IF  : if time for the event has expired, re-assign next check time

IF 

 : re-check time is static (add to current time)

ELSE

 : re-check time is Poisson distribution value

 : update previous check time to current time

IF and vehicle position inside region (vehicle is active in region)

IF  : if previous check occurred this time step, assign new event value

IF 

 : generate random num based on mean

ELIF 

IF  : if random number less than event probability

= TRUE

ELSE

= FALSE

ELSE RETURN -1234 : dummy value to signify that event value does not change

ELSE : vehicle is not active in that event

RETURN  : return default value

*1.8* **Account for Vehicles Not Seen in Last Time Step**

Have we seen all of the vehicles from the last time step? If not, then count them as departed.

For those that have departed, WRITE to log all of the snapshots still remaining in the buffer for this vehicle.

FOR EACH 

IF THEN

 : update vehicle record, vehicle has departed

FOR EACH TO  : for each vehicle buffer entry

WRITE  : write snapshot record

END FOR EACH 

ELSE

 : re-set active flag for new time period

END IF

END FOR EACH 

RETURN

1. **Check Snapshot Triggers**.

IF Timeperiod is a whole second, check for PDM snapshot triggers

GOTO 2.0.5

ELSE

GOTO 2.0.4

*2.0.4* **PDM Collection Check**

FOR EACH 

IF total time in network or distance travelled is >= minimum required before PDM collection

GOTO 2.0.5

ELSE

RETURN

*2.0.5* **Event Snapshot Trigger**



IF AND then

GOTO 9.0 with  : Generate event (m=0) snapshot

GOTO 3.0. : only one snapshot generated per second

END IF

*2.1* **Stop Trigger**

CASE  (Time and Speed) then

IF then : Vehicle has motion  
  : Cumulative sec motionless is 0  
ELSE

 : increment cumulative motionless sec  
IF  AND  AND 

: No stop recorded in lag interval : AND sufficient seconds of no motion

: AND we are not looking for a start

GOTO 9.0 with  : Generate stop (m=1) snapshot

 : Record time-stamp of last stop

 : Begin looking for starts

GOTO 3.0. : only one snapshot generated each second

END IF

END IF

END CASE (2.1)

2.2 **Start Trigger**

CASE  (Time and Speed) then

IF then : We are looking for a start

IF then : Vehicle has sufficient speed

GOTO 9.0 with  : Generate start (m=2) snapshot

 : Stop looking for starts

GOTO 3.0.

END IF

END IF

END CASE (2.2)

*2.3* **Periodic Trigger**

CASE  (Speed Interpolation) then

IF then : current time is slated for snapshot

IF  then : we are not currently in “stop” status

GOTO 9.0 with  : Generate periodic (m=3) snapshot

IF then : vehicle moving under slow threshold

 : set for shortest interval

ELSE

IF  then : vehicle moving over fast threshold

 : set for longest interval

ELSE

 : interpolation

END IF

END IF

 : set clock for next periodic snapshot

END IF

END IF

CASE  (Speed Exception) then

IF then : current velocity out of range

IF  then : we are not currently in “stop” status

GOTO 9.0 with  : Generate periodic (m=3) snapshot

 : set clock for next periodic snapshot

 : set new baseline speed

END IF

ELSE

IF  then : time for a periodic snapshot anyway

IF  then : we are not currently in “stop” status

GOTO 9.0 with  : Generate periodic (m=3) snapshot

 : set clock for next periodic snapshot

 : set new baseline speed

END IF

END IF

END IF

END CASE (2.3)

*3.0***Check for RSE Interaction**

* 1. Identify vehicles equipped to transmit via DSRC that are not in a privacy gap

Let , the set of vehicle ID lists where each list corresponds to vehicles in range of each RSE, to be the empty set.

* 1. Check to see which vehicle IDs are in range of each RSE and assemble a list of vehicle IDs in range to each RSE. Combine lists into set 
  2. Transmission occurs if there is an eligible RSE and we have snapshots to send

FOR EACH  : Each RSE

FOR EACH 

: Each eligible vehicle in range to the RSE

IF  is equipped to transmit BSMs,

WRITE  to log

:Generate and transmit BSM via DSRC

IF  then : enough PDM snapshots to transmit

GOTO 10.0 with  : Transmit buffer contents to RSE

END IF

END FOR EACH

END FOR EACH

1. **Check for Cellular Interaction**

Check for user-defined cellular regions or user-defined defaults, else assign default loss rate (0%), latency (0 sec), and minimum PDMs to transmit (4 snapshots)

FOR EACH 

WRITE  to log

: record BSM in log as cellular transmission

IF  : if whole second timestep

FOR EACH 

: check for cellular transmission for each vehicle

FOR EACH  : loop through each cellular region

IF  is located in the area of 

 : set transmission latency, loss rate, and region name





IF minimum number to transmit : if buffer size is large enough

IF 

GOTO 10.0 : Transmit PDMs adding (latency) to *t*

ELSE

FOR EACH  : transmission was lost

WRITE 

: write snapshot record with delete reason 5

 : set buffer contents to 0

END IF

1. **Manage PSN and Privacy Gaps**

5.0 IF then GOTO 4.5. :if in a privacy gap, don’t manage PSNs

5.1 Check for expiration of PSN because of distance and/or time

 : set local flag for PSN change to FALSE

Compute  : distance traveled under current PSN

CASE  : Max Time/Distance Association Case

IF AND then

 : set flag for PSN expiry, rollover

END IF

CASE  : Time-Based Association Case

IF then

 : set flag for PSN expiry, rollover

END IF  
  
 CASE  : Distance-Based Association Case

IF then  
  : set flag for PSN expiry, rollover

END IF

END CASE (5.1)

5.2 Check for expiration of PSN because of RSE interaction

IF  then : if we interacted with an RSE this time step

 : set flag for PSN expiry, transmission

END IF (5.2)

* 1. Generate dummy PSN if flag for expiry has been set in Steps 4.1 or 4.2.

IF then

 : create a dummy (null) PSN

 : record timestamp for dummy PSN

END IF (5.3)

5.4 Check to see if we need to institute a privacy gap

Put in a new gap if the policy is for gaps, and one of the PSN change criteria have been met

IF  AND then

 : gap 1 if rollover, 2 if trans

 : timestamp for gap start

 : set time for gap timeout

 : set distance (feet) for gap

END IF (5.4)

5.5 Check to see if privacy gap has expired; generate new PSN

IF then : we are currently in a gap

Compute  : distance traveled in gap period

IF OR then : either time OR distance gap expired

 : privacy gap no longer in effect

 : create new PSN

 : reset timestamp for new PSN

END IF

END IF (5.5)

9.0 **Snapshot Generation and Buffer Management**

Add new record to snapshot Log

 : increment vehicle snapshot counter

 : snapshot generation time

 : init. time/reason for deletion; trans. time

 : initialize RSE transmission location (null)

Add entry to vehicle snapshot log, , with initialized inputs and vehicle position, ID and speed

 : Vehicle snapshot log entry

* 1. Check to see if Privacy Gap currently in place

IF AND  then : we are currently in a non-event gap

 : record time of snapshot deletion

 : deletion type: rollover (2), transmission (3)

WRITE  : record deleted snapshot in log

GOTO 9.3 : snapshot not placed in buffer

END IF (9.1)

* 1. **Manage Buffer Contents**

If the vehicle ID does not have a Buffer (first time seeing the vehicle), create a buffer to hold starts, stops, and periodic snapshots as well as the current pointer (for deletion methods) and last RSE transmitted to.

ELIF then : if buffer isn’t full

 : increment buffer count

IF  THEN : if the new snapshot is a periodic snapshot

 : increment periodic snapshot in buffer count

END IF

 : temporary counter to look through buffer

 : consecutive deletion counter reset to 0

9.2.5  : puts new snapshot into position (9.2.5)

IF  AND : priority and timestamp check

FOR DOWN TO  : move records up in list

 : to make room for snapshot

END FOR

 : snapshot stored in buffer at this spot

ELSE

IF 

GOTO 9.2.5

ELSE

 : snapshot takes last position

END IF

END IF

ELSE (9.2) : buffer is full, some snapshot to be deleted

 : ID of snapshot to be deleted (null)

FOR TO  : search for slot to insert snapshot

CASE  : ESP, most recent retention policy

IF 

AND : priority and timestamp check

Let  : recover ID for deleted snapshot

Let  : store type of deleted snapshot

 : go to last position in buffer

END IF

CASE  : ESP, delete every other periodic

IF 

AND  : priority and timestamp check

Let  : store type of deleted snapshot

IF  THEN

Let  : delete oldest, lowest priority

 : re-sort from last position in buffer

ELSE

IF  : delete counter less than SOI?

 : delete position, save last

 : increment delete counter

ELSE

 : set delete position

 : reset delete counter

END IF

Let  : recover ID for deleted snapshot

END IF

END IF

END CASE

IF  THEN : spot found for new snapshot

 : record time of deletion in log

 : buffer overflow deletion (type 1)

IF  THEN

 : reduce per. snapshot count

END IF

FOR DOWN TO  : move records up in list

 : to make room for snapshot

END FOR (j\*\*)

 : snapshot stored in buffer

LEAVE FOR(J\*) : leave FOR-loop associated with j\*

ELSE  
 IF  then : new snapshot has lowest priority

 : record time of deletion in log

 : buffer overflow deletion (1)

END IF  
END IF

END FOR (j\*)

END IF (9.2)

9.3 RETURN to calling routine

10.0  **Snapshot Transmission**

This routine adds records to the TLOG (Transmitted Snapshot Log) from each of the snapshots contained in the current vehicle buffer. The transmitted snapshot is also recorded in the log. The vehicle buffer is reset after this data transmission.

FOR TO  : transmit all contents of the vehicle buffer

WRITE  : write current time, limited snapshot data to TLOG

 : recover ID, transmitted snapshot

 : record time of transmission in log

IF RSE transmission

 : record RSE ID in log

ELSE

 : record Cellular region name in log

WRITE  : full record written to log

END FOR

 : set buffer contents to 0

 : set last RSE to RSE that received the transmission

RETURN to calling routine

# 