Highway Network Conversion How-To Guide – TRANSIMS Version 5

Version 1.0

Revision History:

May - July 2013 – Created by the Volpe Center

Metropolitan Planning Organizations (MPOs) routinely use transportation planning models to plan and assess surface transportation investments and policies. Advances in desktop computing performance over the last 20 years, along with improved data availability, have provided the opportunity to produce a more detailed level of modeling and a more descriptive and granular set of network model data.

Network data inputs from geographic information systems or transportation analysis software packages can be converted into the input data format required by TRANSIMS 5. This data is often available from the Metropolitan Planning Organization for a given study area. More descriptive network data, where available, is also useful.

This document describes the basic processes, data, and business rules used in TRANSIMS 5 to convert a street network (e.g., link, node, and zone text files, or a link shapefile that describes an existing network) into a complete set of TRANSIMS 5 network files. These resultant files can be validated against turning movement counts (TMCs) and/or ground counts. The end result is validation of these network files as being suitable for trip or tour planning, routing, and microscopic simulation.

The procedures described herein can also apply to manually created[1] and/or synthetically-generated network files in most cases (in addition to the two methods noted above). Since many of the steps described in this document result in the production of synthetic network file outputs regardless of the input method used, iterative review and refinement are essential activities in TRANSIMS 5. These activities are further discussed where applicable. Also, the reader is referred to the relevant documents for more detailed discussion of various parameters, files, and module-specific procedures.

[1] Refer to the Tiny Network test case examples. In some cases, manually creating a few records describing a small number of links and nodes is a useful application of this software. For most real-world scenarios, the model size and number of records will be too big for large-scale manual record specification.

Table of Contents

1 Assumptions and Prerequisites

This document assumes you have installed TRANSIMS Version 5.0 on a Windows or Linux computer system and that you understand the basic procedures and terminology for executing TRANSIMS programs.

The TRANSIMS software can be downloaded from http://sourceforge.net/projects/transims/. Documentation is available at http://code.google.com/p/transims/ .

You should have a basic understanding of traditional transportation planning networks and link-node network concepts. A rudimentary understanding of traffic signal systems is desirable.

Text files are used to store the input and output information. You need to be able to review and edit these files using a standard text editor (e.g., vi, NotePad, NotePad++, WordPad) or other software that can manipulate tab-delimited files (e.g., Excel).

This document also describes how to generate Geographic Information System (GIS) shapefiles for displaying and editing network data in map format[2]. Familiarity with software that can read and display shapefiles is desirable. A brief description of network visualization using ArcGIS 9.1 is provided.

1.1 Download Network Data

A number of test data sets are available.

A small (20 link) test network can be accessed at
http://code.google.com/p/transims/wiki/TinyExampleInfo

A larger network (Alexandria, Virginia) is available at
http://sourceforge.net/projects/transims/files/test%20data/Alexandria%202008-06-10/

You should create a directory with a name such as

```
c:\TRANSIMS\Alexandria          (Windows)
/home/TRANSIMS/Alexandria       (Linux)
```

and unzip the downloaded file into that directory. By convention, the following subdirectories are typically used in a TRANSIMS implementation:

- **\batch** - executable batch files which automate certain TRANSIMS procedures
- **\control** - control files that direct the operation of various TRANSIMS components
- **\demand** - contains trip tables, trip times, and vehicle types - used by TRANSIMS to generate trips

[2] A shapefile is a popular format, developed by ESRI, for representing geographic data. A given GIS layer (e.g., a link layer) will typically include shapefiles with the following suffixes: .shp, .shx, .dbf. Shapefiles may represent points (e.g., nodes), lines (e.g., links) or polygons (e.g., transportation analysis zones)

- **\inputnetwork** - text files defining nodes, links, zones, and shapes used by TransimsNet to generate the network

- **\network** - the network generated by TransimsNet will be stored here
    - **\network\arcview** - initially empty; ArcNet output will be stored here

- **\results** - initially empty; results from each model iteration will be stored here

1.2 Network Data Source

Network data can be derived from a variety of sources. For example, traditional regional travel demand forecasting networks have been used as basic input to the TRANSIMS process. A number of utility programs are available to help convert data in other formats to the standard input network format described in this document.

2 TRANSIMS Network Overview

A TRANSIMS network is a collection of 20 or more files that define various aspects of the highway and transit facilities and operational characteristics. Although a given application may not require all of these files, it will require most of them in order to perform a detailed microscopic simulation. Most transportation modelers are familiar with node, zone, and link file types from traditional travel demand forecasting software packages. Many of the other file types will likely be new to them. It is possible to create the data required for each file through a manual coding process, similar to the way regional modeling networks have been created in the past. For a large network, however, this can be a very time consuming process.

This document presents an alternate approach that starts from network data that are generally available from regional travel demand forecasting models or geographic information systems and applies general traffic engineering principles and operational assumptions to synthesize a complete set of TRANSIMS network files. The synthetic data can then be refined as necessary to more accurately replicate localized conditions.

Figure 1 shows the typical execution sequence of TRANSIMS 5.0 network programs and file inputs and outputs. The other network programs (blue boxes in Figure 1) are discussed in detail in their respective Program References. Starting with TRANSIMS 5.0, NetPrep is used instead of TransimsNet to convert input network files (nodes, links, and zones) into TRANSIMS 5.0 network format compatible files. The output files from TransimsNet are then used as inputs to IntControl to generate the remaining network files required for a TRANSIMS simulation. Not readily apparent from Figure 1 is the iterative nature of the process, subsequent review, manipulation, and generation of the remaining TRANSIMS network files.

The raw input files may be from an existing TRANSIMS 4.0 network, from raw network files exported from commercial traffic modeling software (e.g., TransCAD) or may be manually created given that the file names, file types /formats /extensions, location, layout (header row in particular), and contents are consistent with what is expected by TRANSIMS 5.0 and the relevant control files for each program noted above.

Table 1 summarizes the network files. Most transportation modelers are familiar with node, zone, and link file types from traditional travel demand forecasting software packages. Many of the other file types will likely be new to them. It is possible to create the data required for each file through a manual coding process, similar to the way regional modeling networks have been created in the past. For a large network, however, this can be a very time consuming and data-collection intensive process.

Note that the toll and signal coordinator files from TRANSIMS version 4 are no longer used.

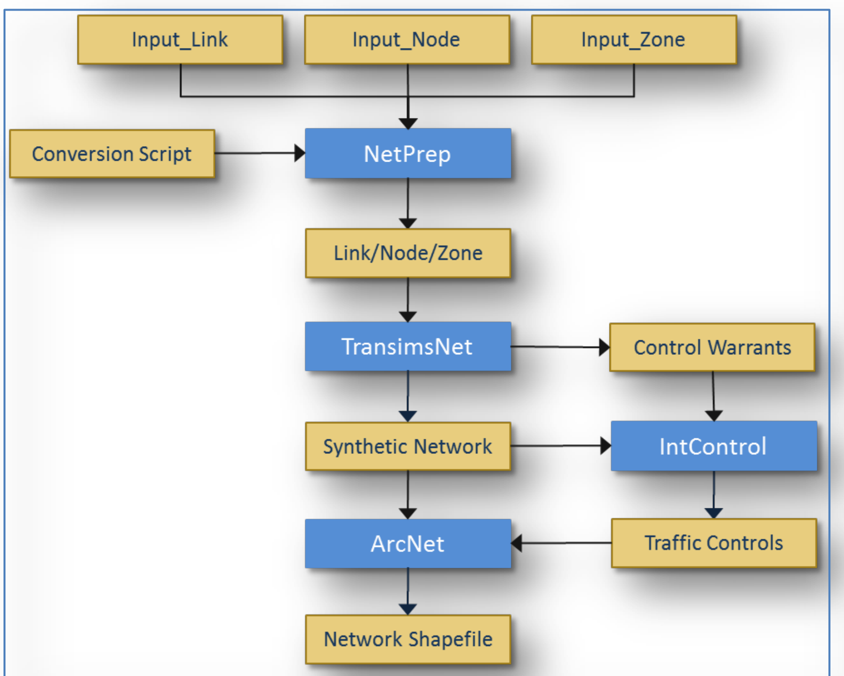

Figure 1 High Level Data Flow for Network Construction

Table 1 Network Filenames

Typical Filename	Purpose	How the file is typically created
Input_Node.txt	Nodes for the network	From an existing planning model or GIS database
Input_Link.txt	Links for the network	From an existing planning model or GIS database
Input_Zone.txt	Zone centroids for the network	From an existing planning model or

Typical Filename	Purpose	How the file is typically created
		GIS database
Input_Shape.txt	Shape points for the links in the network	Netprep (formerly GISNet) is used to create this shape file from an .shp shape file
Turn_Prohibition.txt	Optional file of turn restrictions and penalties	From an existing planning model
Node.txt	Nodes for the network	Netprep conversion from Input_Node.txt
Link.txt	Links for the network	Netprep conversion from Input_Link.txt
Zone.txt	Zone centroids for the network	Netprep conversion from Input_Zone.txt
Shape.txt	Shape points for the links in the network	Netprep conversion from Input_Shape.txt
Location.txt	Locations on the network where flow enters or exits. These take the place of zone centroids	TransimsNet
Parking.txt	Parking lots	TransimsNet
Process_Link.txt	Process links	TransimsNet
Pocket.txt	Pocket lanes	TransimsNet
Connection.txt	Connections within intersections	TransimsNet
Lane_Use.txt	Optional file of lane use restrictions and tolls	Manually created
Sign_Warrants.txt	Nodes that should have stop or yield controls	TransimsNet
Signal_Warrants.txt	Nodes that should have traffic signals	TransimsNet
Sign.txt	Nodes with stop or yield controls	Intcontrol
Signal.txt	Nodes with signals	Intcontrol
Timing_Plan.txt	Timing plans for the signals	Intcontrol
Phasing_Plan.txt	Phasing plans for the signals	Intcontrol
Detector.txt	Detectors for the signals	Intcontrol

Additional methods will typically be needed in order to generate a valid and verifiable TRANSIMS 5.0 synthetic network suitable for trip table generation, and eventually, for accurate simulation of traffic flows. This includes both pre- and post-processing methods to review, edit, and reiterate steps as needed. Errors and warnings should be addressed as they arise, and resolved whenever feasible. Warnings don't necessarily prevent subsequent use of a file by another program. Batch files can be used to good effect in iterating some repetitive tasks such as regenerating the network from scratch using updated files. Additionally, scripting methods (e.g., Visual Basic, VBA, Python, etc.) can also prove quite useful for automating many repetitive tasks (iterative program execution, file manipulation, file input/output, etc.), especially with larger scale efforts.

3 How to Synthesize a TRANSIMS Highway Network

Figure 2 is an elaboration of Figure 1 that depicts the process for building a TRANSIMS network. The steps in this figure are numbered, and correspond to sections in this document.

The first step in synthesizing a TRANSIMS network is to run TransimsNet using the aforementioned files as inputs. NetPrep may be required or optionally used prior to TransimsNet or IntControl (as described above), depending on the needs of the particular TRANSIMS implementation. The successful execution of TransimsNet generally requires both pre- and post-processing edits, manual review of results, and reiteration of TransimsNet (and possible creation and addition of a Keep Node file) before proceeding to run IntControl. Post-processing resolution of network warnings and errors, manual adjustment of sign and signal warrant location placement, and similar will be necessary in most cases. Likewise, successful execution of IntControl requires a similar manual review and editing process similar to TransimsNet.

The TRANSIMS network modeling process, iterative refinement process, and validation of the final network model are all aided by using a combination approach of text- and graphics-based network file review/editing/refinement. Typical text-based approaches range from using a basic text editor to review and edit individual files all the way up to relational database management systems (RDBMS)-based approaches for file and data management. In addition, macros and scripts can be useful in automating some of the repetitive processes inherent to TRANSIMS. Graphics-based review and revision approaches employ visualization of ESRI-compatible network shape files created by ArcNet and subsequently loaded and rendered as vector layers using a GIS program.

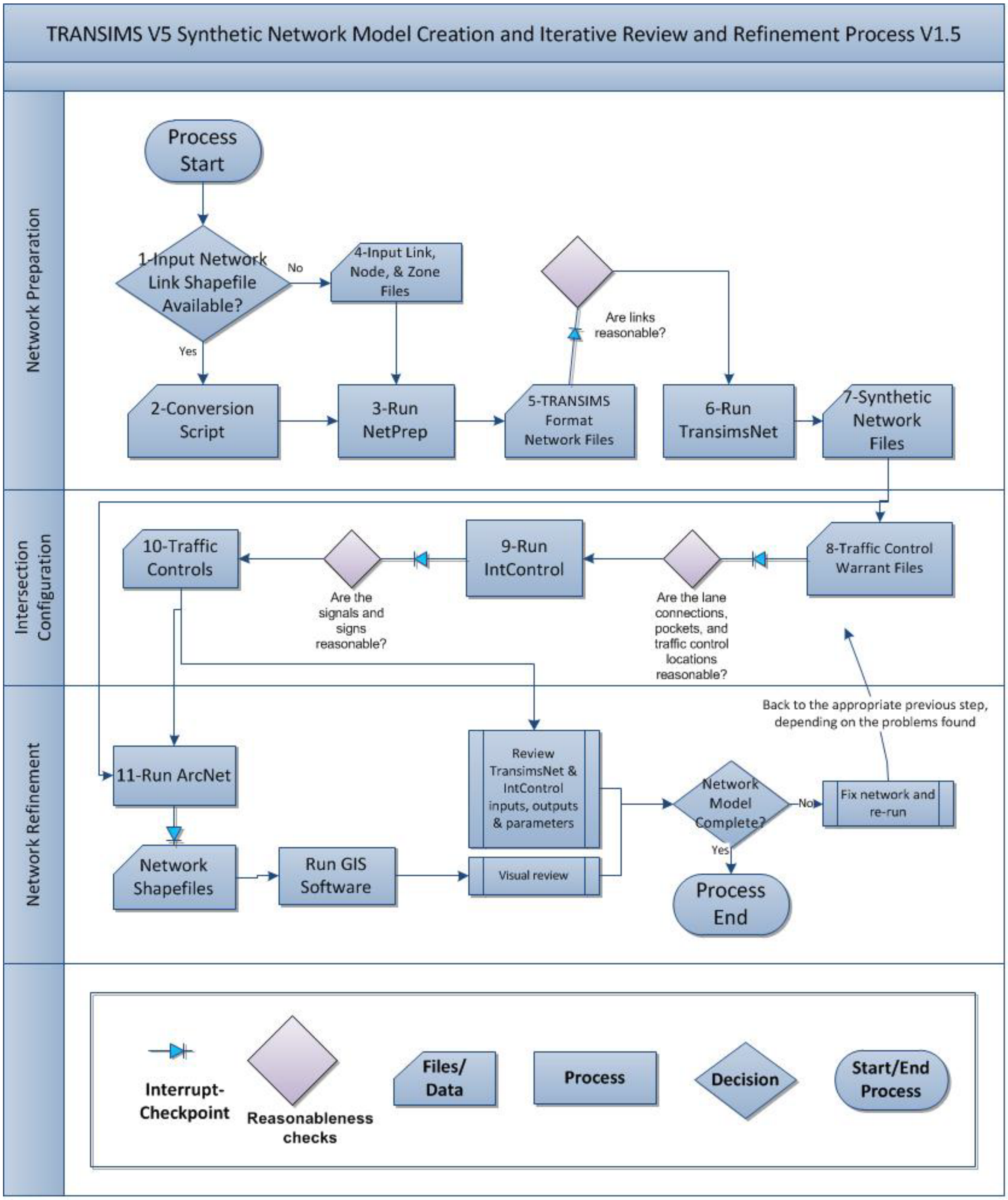

Figure 2 TRANSIMS Network Synthesis

3.1 Input Network Shapefile

In many cases, a routable GIS shape file will be available. Netprep is used to convert this shape file to TRANSIMS input node, input link and input shape files. Figure 3 shows an example of a few links in a shapefile available from a state GIS. It is displayed in QGIS (an open-source Geographic Information

System tool), and includes a satellite overlay. Example 2 in the NetPrep program reference also shows an example of such a conversion.

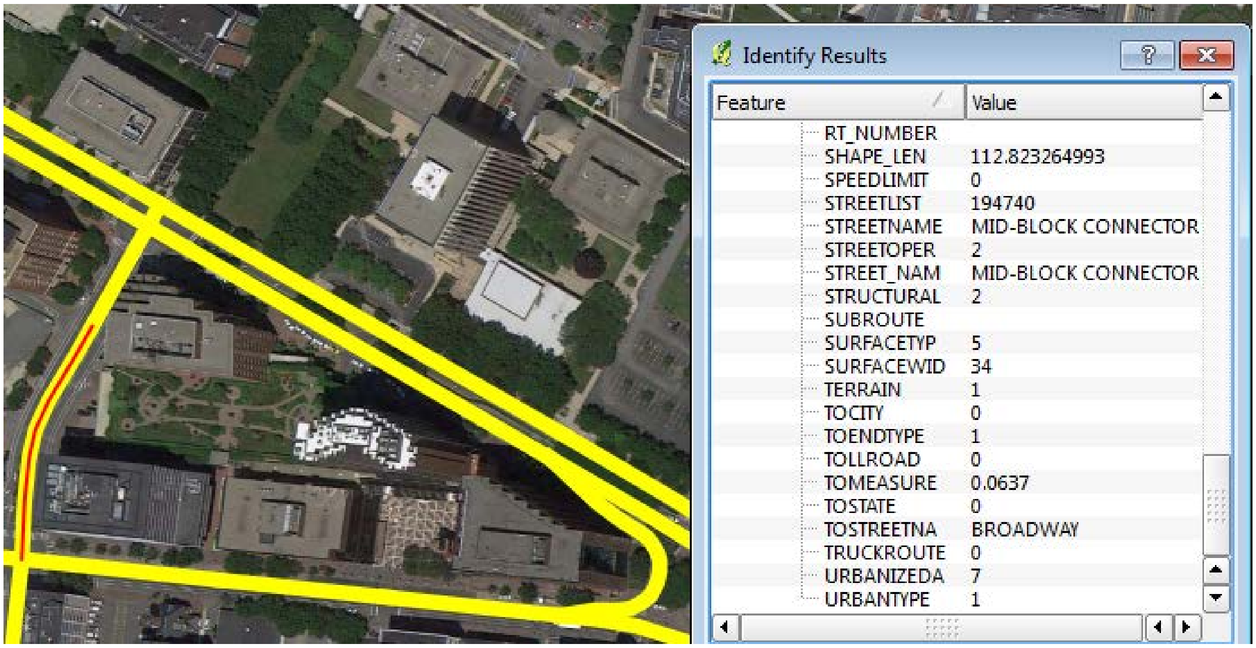

Figure 3 Input Shapefile from a State GIS

3.2 Conversion Script

The conversion script is used by Netprep to convert fields in the shapefile to TRANSIMS field names. An example appears below. The conversion script depends on the fields and data available in the source shapefile. It maps the fields in the source shapefile (Link.*) to fields in a TRANSIMS link file (NewLink.*).

```
NewLink.USE = "ANY"
NewLink.NAME = Link.STREETNAME
NewLink.LENGTH = Link.SHAPE_LEN
// STREETOPER is a code for one-way street
// NUMBEROFTR is number of travel lanes
IF (Link.STREETOPER == 1) THEN
NewLink.LANES_BA = 0
NewLink.LANES_AB = Link.NUMBEROFTR
ELSE
NewLink.LANES_BA = Link.NUMBEROFTR / 2
NewLink.LANES_AB = Link.NUMBEROFTR / 2
ENDIF
NewLink.FSPD_AB = Link.SPEEDLIMIT
NewLink.FSPD_BA = Link.SPEEDLIMIT
NewLink.TYPE = "LOCAL"
IF (Link.FUNCTIONAL == 2 || Link.FUNCTIONAL == 3) THEN
NewLink.TYPE = "PRIARTER"
ENDIF
```

```
IF (Link.FUNCTIONAL == 5) THEN
NewLink.TYPE = "SECARTER"
ENDIF
IF (Link.FUNCTIONAL == 6) THEN
NewLink.TYPE = "COLLECTOR"
ENDIF

NewLink.NOTES = FORMAT("%",Link.SURFACEWID)
RETURN (1)
END
```

Example 2 in the NetPrep program reference shows another example of a conversion script.

3.3 NetPrep

NetPrep is a TRANSIMS program used to prepare networks. It takes as inputs either a shape file (.shp) as discussed above, or node, link and TRANSIMS shape files. An example of a NetPrep control file appears below:

```
TITLE                        NetPrep GISNet-Type Network Conversion

PROJECT_DIRECTORY            ./
UNITS_OF_MEASURE             METRIC

INPUT_LINK_FILE              InputShapeFile.shp  // From State or MPO GIS
CONVERSION_SCRIPT            NetPrepScript.txt   // The conversion script

NEW_NODE_FILE                network/Input_Node.txt  // Outputs from this script
NEW_LINK_FILE                network/Input_Link.txt  // which become inputs
NEW_SHAPE_FILE               network/Input_Shape.txt // to TRANSIMS

OUTPUT_COORDINATE_SYSTEM     UTM, 19T, METERS

NOTES_AND_NAME_FIELDS            YES
NETPREP_REPORT_1             CONVERSION_SCRIPT
NETPREP_REPORT_2             CONVERSION_STACK
```

Netprep may also optionally be used for other forms of link manipulation, including

- Assigning local streets to a new local-thru facility type based on the length of a series of local links
- Use LOCAL_SELECTION_SPACEING to select representative local streets from an all-streets network to include in the TRANSIMS network
- Use the DROP_DEAD_END_LINKS and DROP_SHORT_LINKS parameters to ensure that dead end and links with length of less than a certain value are dropped
- Use the SPLIT_LARGE_LOOPS parameter to split loops (links where the origin and destination nodes are the same) into two links.

3.4 Input Link, Node and Zone Files

3.4.1 Node Data

The node file contains information about the location of each network node or intersection. The sample node file provided in the network directory is called "Input_Node.txt." This file includes three fields: NODE, X_COORD, and Y_COORD. TRANSIMS expects coordinates to be provided as UTM coordinates, in meters. These coordinates are rounded to the nearest decimeter (i.e., one decimal point, or the equivalent of four inches) within the TRANSIMS programs.

Node numbers must be unique positive integers. They do not need to be sequential or sorted. If zone centroids are included in the node file, they must be the lowest node numbers. If external stations are included, they should be the highest zone numbers.

3.4.2 Link Data

The link file contains information about roadways, walkways and transit guideways; in a two-directional coding convention. The beginning of the link is identified by an "A" node number, and the end of the link is identified by a "B" node number. The number of lanes, speeds, and other directional attributes are then identified as "AB" or "BA." TRANSIMS permits the user to code separate links in each direction if necessary, but this tends to complicate the coding and data processing. Links should not have the same "A" node and "B" node.

The sample link file provided in the network directory is called "Input_Link.txt." This file includes 11 fields: LINK, STREET, ANODE, BNODE, LENGTH, TYPE, LANES_AB, SPEED_AB, LANES_BA, SPEED_BA, and USE. All but the LINK and STREET fields are required. It is recommended that hourly capacities (CAP_AB, CAP_BA) be provided, if the data are available. If they are not, capacity will be estimated based on the facility type and the number of lanes. Capacities are typically used to allocate green time in the signal generation process and to estimate travel times based on traditional volume-delay functions prior to the first simulation. The Simulator does not use the capacity data to estimate network performance or throughput.

The link length is traditionally defined in meters, and the speed values are defined in meters per second. The facility type and vehicle use codes are provided as text strings. The options for facility type include:

FREEWAY, EXPRESSWAY, PRINCIPAL, MAJOR, MINOR, COLLECTOR, LOCAL, FRONTAGE, RAMP, BRIDGE, EXTERNAL, XPRESSWAY, PRIARTER, SECARTER, ZONECONN, OTHER, WALKWAY, BIKEWAY, BUSWAY, LIGHTRAIL, HEAVYRAIL, FERRY

The vehicle use code is a combination of the following values, entered in any order and separated by slashes (e.g., CAR/TRUCK/BUS).

ANY, WALK, BIKE, CAR, TRUCK, BUS, RAIL, SOV, HOV2, HOV3, HOV4, LIGHTTRUCK, HEAVYTRUCK, RESTRICTED, AUTO, BICYCLE, TAXI, TROLLEY, STREETCAR, LIGHTRAIL, RAPIDRAIL, REGIONRAIL

Note that use codes represent the default access permissions for a link over the full simulation period. Access restrictions that only affect certain times of day or certain directions (e.g., peak period HOV restrictions) are identified in the lane use file. The meaning of each of these codes can be found in the description of the Lane Use table in the TRANSIMS File Reference.

3.4.3 Zone Data

A zone file is optional, but recommended for most applications that cover a diverse geographic area or that need to be linked to zone-based trip tables. If a zone file is provided, every activity location generated in the conversion process will be associated with the closest zone centroid. This information is used to distribute zone-to-zone trips to appropriate activity locations.

The sample zone file provided in the network directory is called “Input_Zone.txt.” This file includes four fields: ZONE, X_COORD, Y_COORD, and AREATYPE. The X and Y coordinates represent the zone centroid and are defined in UTM meters. AREATYPE is a user-defined integer code used by the synthetic network generation process to control signal warrants and intersection configurations. The density and type of development in a given area (i.e., area type) often influences signal density and turn pocket construction.

3.4.4 Shape Data

A shape file is optional. If provided, it includes any additional shape points in UTM meters between the “A” node and the “B” node of the link (in the A→B direction). The sample shape file provided in the network directory is called “Input_Shape.txt.” This file contains two nested records for selected links. The first record has the link number and the number of shape points that follow. The second record includes the X and Y coordinates for each shape point. Link shapes are used in the synthetic network generation process to accurately identify relationships between entering and exiting links at an intersection. They are also useful in generating network maps.

(Note: the TRANSIMS shape file is different from an ArcView Shapefile. An ArcView Shapefile representation of the network links can be converted to TRANSIMS link, node, and shape files using the GISNet program. The ArcNet program can be used to convert TRANSIMS link, node, and shape files to ArcView Shapefiles.)

3.5 TRANSIMS Format Network Files

These are the Input_Node, Input_Link, Input_Zone and Input_Shape files suitable for loading into TransimsNet. Before loading into TransimsNet, the following items should be checked for each link:

1. LENGTH should be >= MIN_LINK_LENGTH[3]
2. LENGTH should be reasonable: typically, between 1 and 2 times the straight line distance from Node A to Node B (there may be exceptions for shorter links, such as cloverleaf interchanges)
3. LANES_AB + LANES_BA must be > 0
4. The TYPE must be a valid functional class[4]
5. SPEED_AB, FSPD_AB must be > 0, and should be reasonable based on the intended use of the link.
6. CAP_AB should be reasonable based on the number of lanes and the functional class[5]
7. If the link is a two-way link, the LANES_BA must be > 0, and SPEED_BA, FSPD_BA, and CAP_BA should be reasonable based on the previous rules
8. If the link is a one-way link, LANES_BA and CAP_BA should be 0
9. USE must be a valid use code
10. Divided roads are handled appropriately. Note that divided roads may be represented in the input GIS network as two links, one for each direction. This makes for complicated multi-node signalized intersections with extremely short links within the intersection. Best practice is to treat the segments of such roads as single two-way links. On the other hand, typical practice is to represent each side of a freeway as a one-way link.

3.6 TransimsNet

This section describes how to set up and run the TransimsNet program. The TransimsNet program uses basic link and node information to synthesize the data fields and network files needed by TRANSIMS. The additional network files contain information about the number and location of turn pockets, the location of parking lots (i.e.,the start and end points of vehicle trips) and activity sites and the process links that connect the two, the lane connections at intersections, and traffic control warrants. The program also removes unnecessary nodes, updates the shape points, and converts external station zone connectors to roadways with the appropriate number of lanes and travel speeds.

In the TRANSIMS network design, trips start and end at locations (formerly known as activity locations). Unlike zones in traditional modeling systems, locations are directly related to links. Each side of a link (or block face) can contain one or more locations. Locations are accessed by walking on the associated link or utilizing a process link. Process links can be used to connect activity locations directly to parking lots or transit stops. Parking lots are also associated with links, and all vehicle trips must start and end at a parking lot. The TransimsNet program automates the generation of location-parking lot combinations at regular intervals on both sides of streets that permit auto access (and that are not freeways, ramps, or bridges).

---

[3] In TRANSIMS, MIN_LINK_LENGTH is a parameter, typically equal to the cell size of 7.5 meters
[4] Examples of functional classes might include Freeway, Primary Arterial, Secondary Arterial, Collector, Local
[5] More specifically, CAP_AB should be less than HOURLY_PER_LANE * LANES_AB, where HOURLY_PER_LANE is a maximum hourly capacity that one might expect on a lane: 2100 for freeway or expressway, 1200 for arterial, 800 otherwise

"Pocket lane" refers to a turn lane entering an intersection, a merge or diverge lane on a freeway, and pull-out lanes at mid-block locations. The TransimsNet program adds pocket lanes based on the intersection configuration and length criteria defined by facility and area types. For example, a freeway merge lane is likely to be longer in suburban areas than in dense urban areas; freeway pocket lanes will be much longer than turn pockets on local streets; and turn pockets in the CBD are likely to be very different from turn pockets on suburban arterials.

Pocket lanes and regular lanes are then numbered and linked to lanes on other links using lane connectivity records. These connections differ if the movement represents a through movement, left turn, right turn, or U-turn. A given entry lane can be assigned one or more connections to exit links and lanes. A turn prohibition file can be provided to eliminate these connections at specific intersections.

The program then uses the intersection configuration and lane connectivity to create connections in an intersection, and assign signal or sign warrants. The modeler defines the signal warrants based on the primary facility types that come together at the intersection in a given area type. The resulting signal and sign warrants are typically reviewed and edited prior to executing the IntControl program to generate the full set of traffic control data files required by TRANSIMS.

An example of a TransimsNet control file appears below:

```
TITLE                             TransimsNet Test

PROJECT_DIRECTORY                 ../

LINK_FILE                         input/Input_link.txt
NODE_FILE                         input/Input_node.txt
SHAPE_FILE                        input/Input_shape.txt
ZONE_FILE                         input/Input_zone.txt

NEW_NODE_FILE                     network/node.txt
NEW_ZONE_FILE                     network/zone.txt
NEW_LINK_FILE                     network/link.txt
NEW_SHAPE_FILE                    network/shape.txt
NEW_LOCATION_FILE                 network/location.txt
NEW_PARKING_FILE                  network/parking.txt
NEW_POCKET_FILE                   network/pocket.txt
NEW_CONNECTION_FILE               network/connection.txt
NEW_SIGN_FILE                     network/sign_warrant.txt
NEW_SIGNAL_FILE                   network/signal_warrant.txt

POCKET_LANE_WARRANT_1             FREEWAY, ALL, 1, RIGHT, 100 meters, 1
POCKET_LANE_WARRANT_2             FREEWAY, ALL, 2, RIGHT, 200 meters, 1
POCKET_LANE_WARRANT_3             FREEWAY, ALL, 3, RIGHT, 300 meters, 1
POCKET_LANE_WARRANT_4             FREEWAY, ALL, 4, RIGHT, 400 meters, 1

POCKET_LANE_WARRANT_5             EXPRESSWAY, ALL, 1, RIGHT, 75 meters, 1
POCKET_LANE_WARRANT_6             EXPRESSWAY, ALL, 2, RIGHT, 150 meters, 1
POCKET_LANE_WARRANT_7             EXPRESSWAY, ALL, 3, RIGHT, 200 meters, 1
POCKET_LANE_WARRANT_8             EXPRESSWAY, ALL, 4, RIGHT, 250 meters, 1

POCKET_LANE_WARRANT_9             PRINCIPAL, ALL, 1, LEFT, 50 meters, 1
POCKET_LANE_WARRANT_10            PRINCIPAL, ALL, 2, LEFT, 100 meters, 1
POCKET_LANE_WARRANT_11            PRINCIPAL, ALL, 3, LEFT, 150 meters, 1
```

```
POCKET_LANE_WARRANT_12          PRINCIPAL, ALL, 4, LEFT, 200 meters, 1

POCKET_LANE_WARRANT_13          MAJOR, ALL, 1, LEFT, 25 meters, 1
POCKET_LANE_WARRANT_14          MAJOR, ALL, 2, LEFT, 50 meters, 1
POCKET_LANE_WARRANT_15          MAJOR, ALL, 3, LEFT, 100 meters, 1
POCKET_LANE_WARRANT_16          MAJOR, ALL, 4, LEFT, 150 meters, 1

POCKET_LANE_WARRANT_17          RAMP, ALL, 1, LEFT, 50 meters, 1
POCKET_LANE_WARRANT_18          RAMP, ALL, 2, LEFT, 100 meters, 1
POCKET_LANE_WARRANT_19          RAMP, ALL, 3, LEFT, 150 meters, 1
POCKET_LANE_WARRANT_20          RAMP, ALL, 4, LEFT, 200 meters, 1

TRAFFIC_CONTROL_WARRANT_1  PRINCIPAL..MAJOR, PRINCIPAL..COLLECTOR, ALL, SIGNAL, 30 
feet, 1
TRAFFIC_CONTROL_WARRANT_2  FREEWAY..EXPRESSWAY, RAMP, ALL, YIELD, 30 feet
TRAFFIC_CONTROL_WARRANT_3  PRINCIPAL..MAJOR, RAMP, ALL, YIELD, 30 feet

MAXIMUM_CONNECTION_ANGLE   150
FACILITY_ACCESS_WARRANT_1       PRINCIPAL..LOCAL, ALL, 50 feet, 150 feet, 3

EXTERNAL_ZONE_RANGE        10..15
```

The facility codes are in a hierarchy, that runs from freeways to the various types of arterials, to collectors, then to local streets:

FREEWAY
EXPRESSWAY (also known as XPRESSWAY)
PRINCIPAL
MAJOR (also known as PRIARTER)
MINOR (also known as SECARTER)
COLLECTOR
LOCAL_THRU (also known as THRU or LOCAL_ACCESS)
LOCAL
FRONTAGE
RAMP

3.7 TransimsNet outputs

The TransimsNet program can be executed using one of the batch files included in the control directory:

TransimsNet.bat	(Windows)
./TransimsNet.run	(Linux)

The printout file "TransimsNet.prn" will be created by the process, along with the new data files stored in the network directory. The printout file will include warning messages about nodes that have entry links but no exit links and other issues. It is advisable to check each of these warnings to ensure that a coding error has not been made. For example, a one-way link might have been coded in the wrong direction. The relative approach angles could also be the cause of the warning message. In this case, the user could adjust the MAXIMUM_CONNECTION_ANGLE parameter to permit sharper connection angles or could manually correct the coding.

The user should review and edit the synthetic network to confirm the coding logic and correct site specific issues. The best way to do this review is typically through a network map. The ArcNet utility converts the TRANSIMS network to a series of ArcView shape files that can be displayed and edited using GIS software (see section 3.11). Particular emphasis should be placed on ensuring that intersections are configured correctly (manual review of sign and signal placement and lane connectivity directionality in particular) before and after creating and updating any of the interdependent intersection files.
In most cases, TransimsNet creates reasonable pocket lanes and connections between links at intersections. However, the following should be checked:
- Intersections with multiple turn lanes (e.g., 2 lanes turning left). TransimsNet does not create these automatically.
- Pocket lanes: do they exist where they should and are they of reasonable length?
- Freeway off ramps. Are left turns inappropriately allowed at an off-ramp where traffic should be turning right?

As noted earlier, these checks can be conveniently performed by running ArcNet on the network files and then examining the resulting shape files (Figure 4).

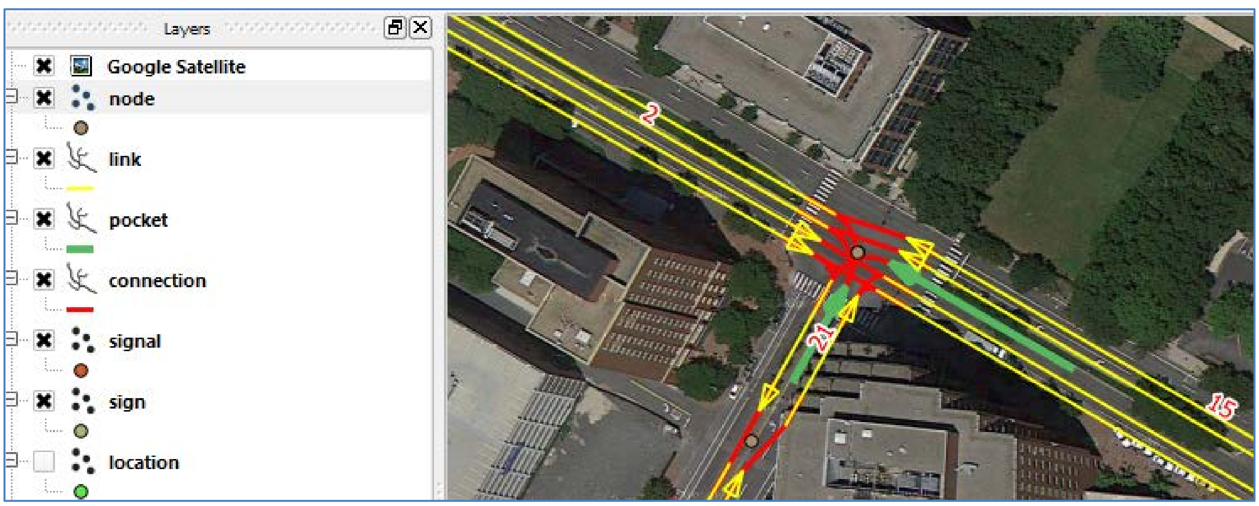

Figure 4 Network Geometry

3.8 Traffic Control Warrant Files

You will also want to review the intersections where the program recommends including signals and signs. Adjusting the area types assigned to various zones is one way to make area wide adjustments to the signal warrants. Site specific corrections can be made by adding or deleting records in the sign or signal files. Most of this review and editing can be accomplished relatively easily in map format. The ArcView shapefiles described in Section 6 are a convenient way to generate network maps.

3.9 Intcontrol

The IntControl program uses intersection configuration and lane connectivity data to populate the traffic control files required by TRANSIMS. The sign and signal warrant files identify the intersections and approach links that require traffic controls. The program validates and completes the unsignalized node file. It then builds signal controls for each signalized node. This includes the timing plan, phasing plan, detectors, and signal coordinator files. The number of phases and the green time allocated to each phase is based on the intersection configuration and the number of lanes or lane capacity assigned to each phase. Fixed timed and demand-actuated signals with one, two, or three rings and block groups can be synthesized based on the input signal warrant.

Users will typically want to review and edit the sign and signal warrants produced by TransimsNet before running the IntControl program. After IntControl has been run, deleting a signal entails deleting the associated timing and phasing plans, and the detectors. This involves deleting multiple records from numerous files with as many interdependencies. Such a manual editing process is both feasible and possibly desirable in some TRANSIMS implementations.

An example of an IntControl control file appears below:

```
TITLE                             IntControl Test
INPUT_SIGN_FILE                   network/sign_warrant.txt
INPUT_SIGNAL_FILE                 network/signal_warrant.txt
NODE_FILE                         network/node.txt
LINK_FILE                         network/link.txt
SHAPE_FILE                        network/shape.txt
POCKET_FILE                       network/pocket.txt
CONNECTION_FILE                   network/connection.txt

NEW_SIGN_FILE                     network/sign.txt
NEW_SIGNAL_FILE                   network/signal.txt
NEW_TIMING_PLAN_FILE              network/timing_plan.txt
NEW_PHASING_PLAN_FILE             network/phasing_plan.txt
NEW_DETECTOR_FILE                 network/detector.txt

SIGNAL_TYPE_CODE_1                ACTUATED
NUMBER_OF_RINGS_1                 1
SIGNAL_TIME_BREAKS_1              2:00, 10:00
SIGNAL_CYCLE_LENGTH_1             90 seconds
MINIMUM_PHASE_TIME_1              5 seconds
YELLOW_PHASE_TIME_1               3 seconds
RED_CLEAR_PHASE_TIME_1            1 seconds
SIGNAL_SPLIT_METHOD_1             CAPACITY
MINIMUM_LANE_CAPACITY_1           500
MAXIMUM_LANE_CAPACITY_1           1500
POCKET_LANE_FACTOR_1              0.5
SHARED_LANE_FACTOR_1              0.5
TURN_MOVEMENT_FACTOR_1            0.9
PERMITTED_LEFT_FACTOR_1           0.5
GENERAL_GREEN_FACTOR_1            1.0
EXTENDED_GREEN_FACTOR_1           0.5
MAXIMUM_GREEN_FACTOR_1            2.0
SIGNAL_DETECTOR_LENGTH_1          30 feet
```

3.10 Traffic Controls

The IntControl program can be executed using one of the batch files included in the control directory:

IntControl.bat (Windows)
./IntControl.run (Linux)

The printout file "IntControl.prn" should be created by the process along with the new data files stored in the network directory. The printout file will contain a number of warning messages about missing traffic controls and problematic intersections. It is advisable to review these warnings to ensure that a coding error has not been made.

3.11 Using ArcNet for Network Checking

You should review and edit the synthetic network to confirm the coding logic and correct site specific issues. The best way to do this review is typically through a network map. The ArcNet utility converts the TRANSIMS network to a series of ArcView shapefiles that can be displayed and edited in ArcGIS or other mapping software. This section describes how to set up and run the ArcNet program.

The ArcNet program can generate an ArcView shapefile representation of most of the TRANSIMS network files. Since the dBase file included in the ArcView shapefile is a complete copy of the data fields included in the corresponding TRANSIMS file, it is possible to edit or manipulate the ArcView shapefile data fields and use the edited data in TRANSIMS programs. The dBase file can be converted back to a standard text file using the FileFormat utility or other software such as Excel. Alternatively, TRANSIMS programs can be configured to read the dBase file directly.

The ArcNet program also provides ways of controlling the relative placement and offset of data items in map format. This includes generating parallel shapes for each direction of travel and user-controlled offset distanced for displaying parking lots, activity locations, and traffic control features. The utility can also convert coordinates from UTM meters to the state plane coordinate system or the latitude-longitude system for integration with other data sources.

ArcNet can be applied in a number of ways for a number of purposes. It does not need to convert all of the TRANSIMS network files at one time. Most conversions will need to include at least the link and node files to provide geographic references for displaying data objects that are defined based on link-offsets. Map representations (see, for example, Figure 5) of the initial input files and the output of TransimsNet are good ways of reviewing intermediate products and addressing warning and error messages generated along the way.

An example of an ArcNet control file appears below:

```
TITLE                           ArcNet Network Tiny Example
NODE_FILE                       network/Node.txt
```

```
ZONE_FILE                   network/Zone.txt
SHAPE_FILE                  network/Shape.txt
LINK_FILE                   network/Link.txt
CONNECTION_FILE             network/Connection.txt
PARKING_FILE                network/Parking.txt
LOCATION_FILE               network/Location2.txt
POCKET_FILE                 network/Pocket.txt
SIGNAL_FILE                 network/Signal.txt
TIMING_PLAN_FILE            network/Timing_Plan.txt
PHASING_PLAN_FILE           network/Phasing_Plan.txt
DETECTOR_FILE               network/Detector.txt
SIGN_FILE                   network/Sign.txt
VEHICLE_TYPE_FILE           ../input/Vehicle_Type_V5.txt

NEW_ARC_NODE_FILE                  network/arcview/node.shp
NEW_ARC_ZONE_FILE                  network/arcview/zone.shp
NEW_ARC_LINK_FILE                  network/arcview/link.shp
NEW_ARC_POCKET_FILE                network/arcview/pocket.shp
NEW_ARC_CONNECTION_FILE            network/arcview/connection.shp
NEW_ARC_PARKING_FILE               network/arcview/parking.shp
NEW_ARC_LOCATION_FILE              network/arcview/location.shp
NEW_ARC_SIGN_FILE                  network/arcview/sign.shp
NEW_ARC_SIGNAL_FILE                network/arcview/signal.shp
NEW_ARC_TIMING_PLAN_FILE           network/arcview/timing_plan.shp
NEW_ARC_PHASING_PLAN_FILE          network/arcview/phasing_plan.shp
NEW_ARC_DETECTOR_FILE              network/arcview/detector.shp


DRAW_NETWORK_LANES                 TRUE
LANE_WIDTH                         12 feet
CENTER_ONEWAY_LINKS                FALSE
LINK_DIRECTION_OFFSET              6 feet
DRAW_AB_DIRECTION                  FALSE
POCKET_SIDE_OFFSET                 6 feet
PARKING_SIDE_OFFSET                60 feet
LOCATION_SIDE_OFFSET               100 feet
SIGN_SIDE_OFFSET                   6 feet
SIGN_SETBACK                       6 feet
DRAW_ONEWAY_ARROWS                 TRUE
ONEWAY_ARROW_LENGTH                20 feet
ONEWAY_ARROW_SIDE_OFFSET           6 feet
CURVED_CONNECTION_FLAG             TRUE
INPUT_COORDINATE_SYSTEM             UTM, 18N, METERS
INPUT_COORDINATE_ADJUSTMENT         0.0,0.0,1.0,1.0
OUTPUT_COORDINATE_SYSTEM            UTM, 18N, METERS
OUTPUT_COORDINATE_ADJUSTMENT        0.0, 0.0, 1.0, 1.0
```

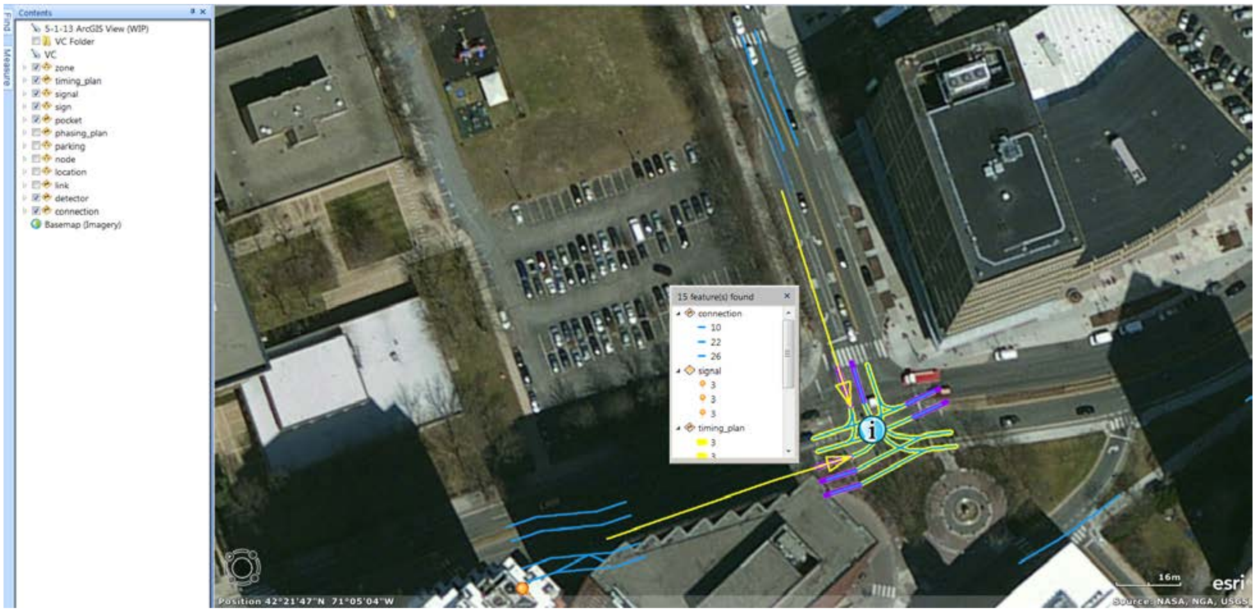

Figure 5 Intersection Detail from a GIS

The ArcNet program can be executed using one of the batch files included in the control directory:

ArcNet.bat	(Windows)
./ArcNet.run	(Linux)

The printout file "ArcNet.prn" should be created by the process along with the new ArcView shapefiles in the arcview subdirectory of the network directory. The ArcView shapefile will have the same name as the input network files with extensions "shp," "shx," and "dbf." The *.shp and *.shx files are binary files that include the shape coordinates and index data. The *.dbf file is the dBase file that includes the data fields from the TRANSIMS data file.

A GIS program such as ArcGIS/ArcMap, or QGIS can display these files in map format. The following graphics show some of the ways TRANSIMS portrays the Alexandria data. The first graph (Figure 6) depicts the network using different colors for roads of different facility type. On top of the road network, colored circles represent the different area types defined at zone centroids. The second graph (Figure 7) shows signals, locations, parking lots, and pocket lanes in downtown Alexandria. The third graph (Figure 8) shows an OpenStreetMap overlay. Close examination reveals an issue with the configuration of the interchange in the lower left corner (the tight cloverleaf in the TRANSIMS network has been replaced with a flyover).

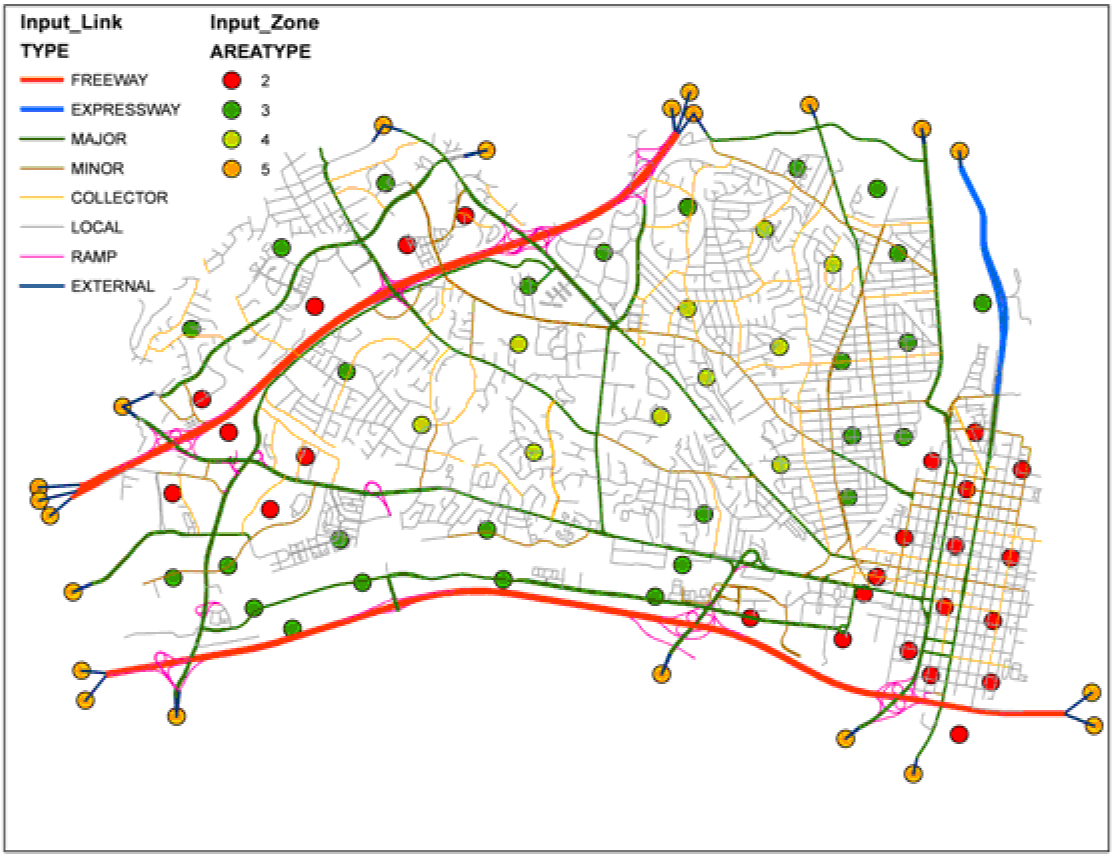

Figure 6 Alexandria Network Overview

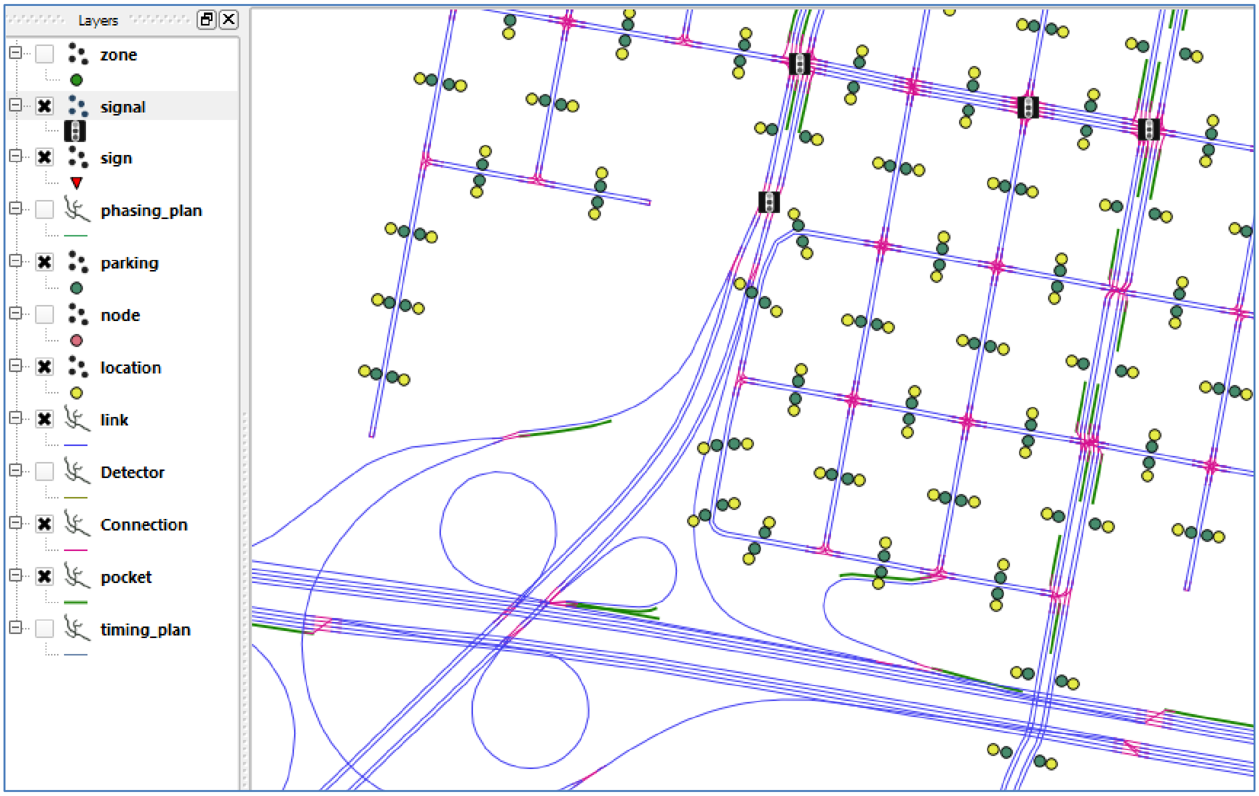

Figure 7 Alexandria Network Detail

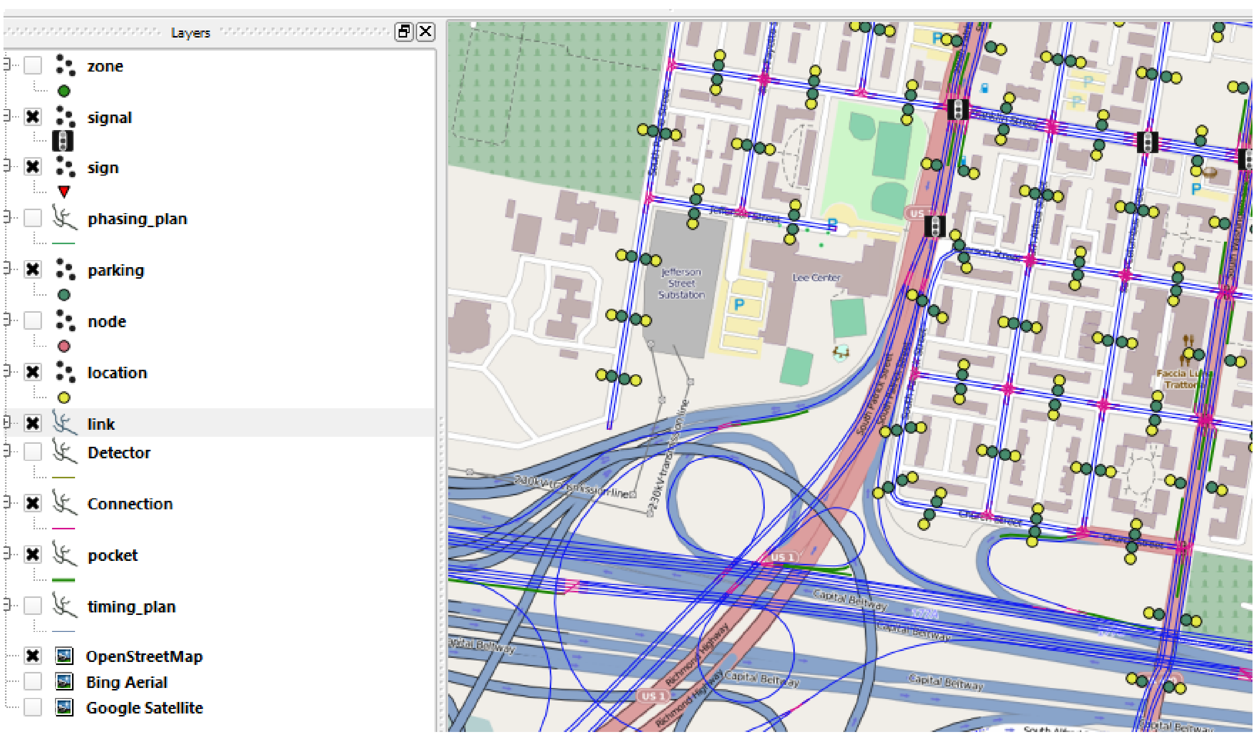

Figure 8 Alexandria Network Detail with OpenStreetMap Overlay

4 Troubleshooting

Traditional travel demand forecasting models tend to use link capacity rather than lanes to define the performance of a roadway. This means that many of the networks used as input to TRANSIMS will not include accurate or consistent lane information. Because the number of lanes is much more important in TRANSIMS, these inaccuracies can cause problems or serious distortions in the way lane connectivity is implemented at intersections and link transitions.

Traditional models also do not pay much attention to the approach and departure angles at intersections or merge points. This often results in very sharp angles that have no impact on the connection within the traditional model, but are not connected within TRANSIMS. In such situations, it is often desirable to set the MAXIMUM_CONNECTION_ANGLE parameter in TransimsNet to 180 degrees to permit connections in all directions at a given intersection.

The way freeway interchanges are coded can have a significant impact on the performance of the Microsimulator. We recommend coding freeways with one-way directional links and accurate ramp connections. The user should also verify that the TransimsNet program placed merge and diverge lanes in the right places.

The way arterials are coded can also cause problems for TRANSIMS. It is important that all of the links entering a signalized intersection come together at a single point. Directional coding with multiple nodes representing the intersection will create lane connectivity and traffic control problems within the Microsimulator.

5 Frequently Asked Questions

Are there utility programs to convert files from other software packages to a TRANSIMS input network?

Utilities for EMME/2, TP+, and ArcView shapefiles have been written to translate data from these packages to TRANSIMS link-node format. TransCAD’s interactive mode can also be used to generated the need input file.

Do we need to have every roadway in the network for TRANSIMS to work properly?

TRANSIMS is designed to simulate networks that include all roadways, but it is not limited to these applications. Applications based on MPO-level networks (i.e., collectors and above) have been successfully implemented. In fact, it is often easier to work with less dense networks because this enables the user to focus on coding the significant facilities more accurately and to not get bogged down with the connections to all of the local streets.