Travel Model Two Development: Sidewalks

Technical Paper

Metropolitan Transportation Commission with Parsons Brinckerhoff, Inc.

September 16, 2013

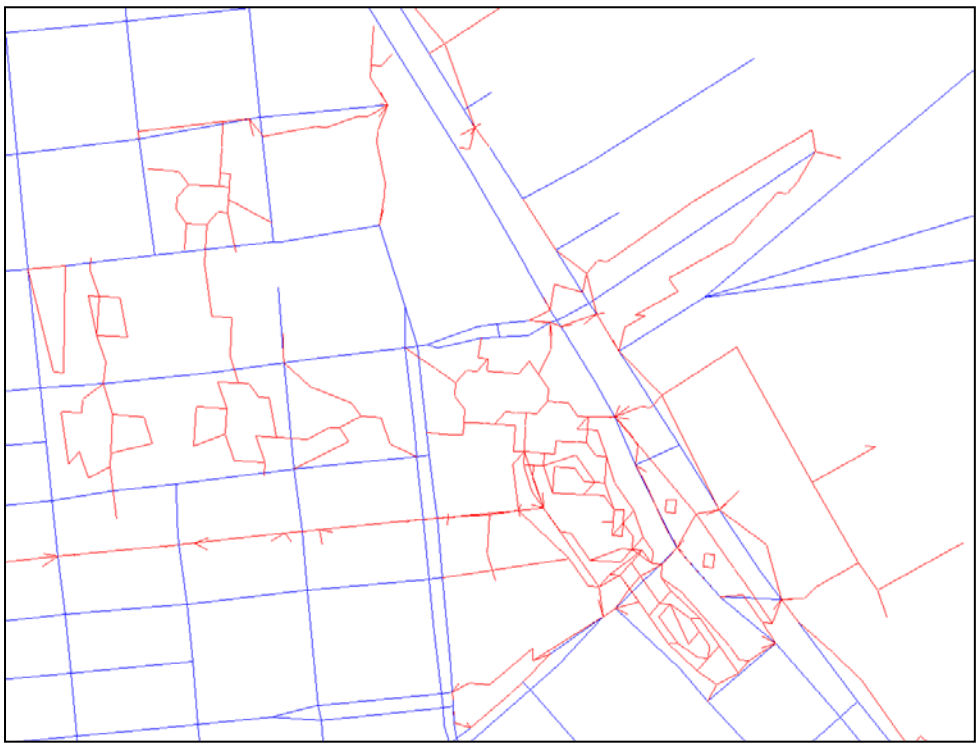
m:\development\travel model two\supply\deliverables\pedestrian\2013 09 16 release pedestrian network.docx

1 Overview

MTC is rebuilding the representation of supply in our travel model. When complete, the new representations of space, roadways, transit service, sidewalks, and bicycle ways will become part of the *Travel Model Two* modeling system. For an overview of the model design, please see the *Travel Model Two: Strategic Design* technical paper[1].

This technical paper describes the development of the *Travel Model Two* sidewalk (or pedestrian) network. Specifically, it discusses the steps for building the network from the *Travel Model Two* roadway network[2] and the open source Open Street Map (OSM) network[3]. We chose OSM for the basis of the pedestrian network because it (a) contains a lot of useful information on pedestrian facilities not available elsewhere; (b) is relatively easy-to-process; and (c) has no use restrictions. An illustrative example of the pedestrian network added to the model network is shown in Figure 1. Figure 2 shows the original network in Open Street Map. The network is created via a series of steps described in detail in this paper. The steps are implemented with MS-DOS batch files, Python scripts, and Cube scripts.

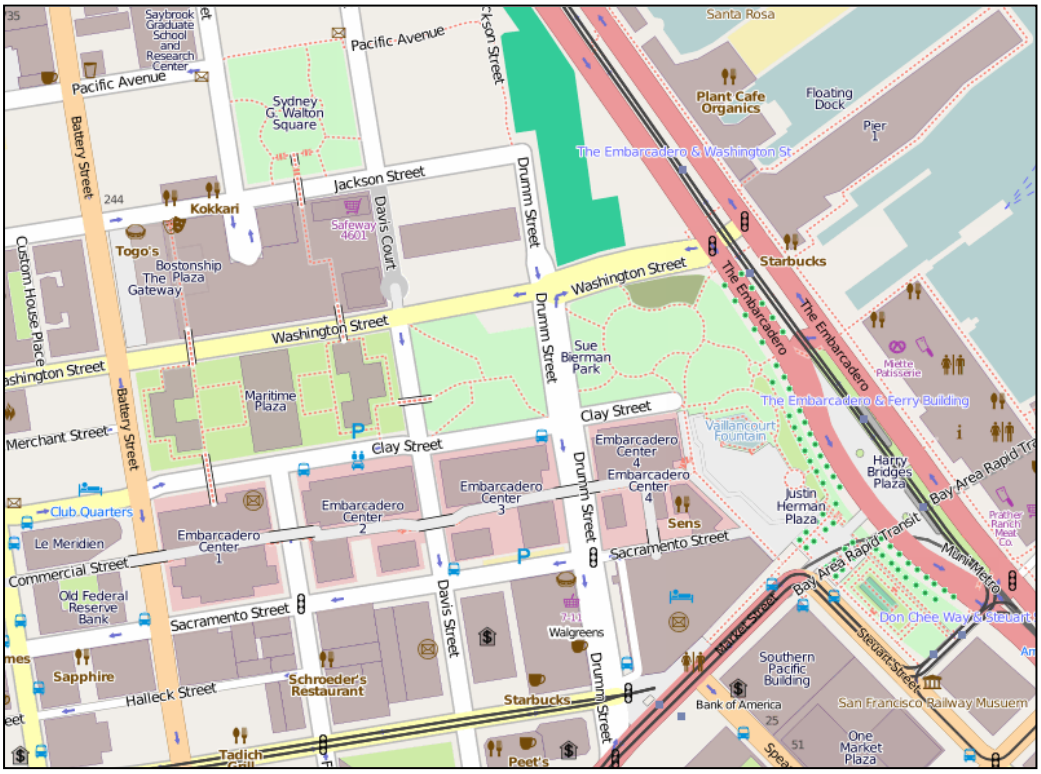
Figure 1 – Example Open Street Map Pedestrian Network Additions in Red (Roadway Network in Blue)



[1] http://analytics.mtc.ca.gov/foswiki/pub/Main/Documents/2012_08_24_RELEASE_Strategic_Design.pdf
[2] Please see the *Travel Model Two Development: Roadway Network* technical paper for details: http://analytics.mtc.ca.gov/foswiki/pub/Main/Documents/2013_09_09_RELEASE_Roadway_Network.pdf.

[3] See OpenStreetMap.org for more details.

Figure 2 – Example Open Street Map Pedestrian Network



2 Algorithm and Inputs

2.1 Algorithm

The procedure for building the pedestrian network proceeds via the following steps:

1. Import the OSM pedestrian network (in XML) into a Cube network file.
2. Export the Cube pedestrian network to a shapefile, and snap the link endpoints to those in the street network (if they are within a certain distance).
3. Determine the locations where the pedestrian network and street network intersect (besides the endpoints snapped to), and split the network links at these points.
4. Combine the pedestrian and roadway networks.

2.2 Inputs

To build the pedestrian network, the following data are required inputs:

1. `sf-bay-area.osm.bz2`: The Metropolitan San Francisco Area OSM extract is available at http://metro.teczno.com. This is an approximately 1.3 gigabyte (GB) XML file which is updated periodically.
2. `tana_sp_with_maz_taz_tap_centroids_connectors_routes.net`: The *Travel Model Two* network in Cube format. The nodes and links are exported to a shapefile for use by the pedestrian network creation scripts.

3 Build Scripts

The pedestrian network is created by the following Python and Cube scripts:

1. `osm_mtc.py`
2. `buildOSM.s`
3. `osm_ped.py`
4. `mergeOSM.s`
5. `osmToShapefile.s`
6. `intersectPedWithStreets.py`
7. `mergeOSMSplitLinks.s`

Each of these scripts is discussed in more detail in the remainder of this section.

3.1 osm_mtc.py

Purpose: Create a Cube node and link file from the OSM information.

This script, which is derived from similar scripts created by the Delaware Valley Regional Planning Commission (DVRPC) and the San Francisco County Transportation Authority (SFCTA), reads the large OSM XML file and converts the OSM nodes and "ways" to Cube nodes and links. Ways tagged as highway are converted to links and the highway type attribute[4] is maintained in order to identify pedestrian links. Only the highway types `footway`, `pedestrian`, or `steps` are read as pedestrian links. OSM nodes are renumbered as they are read in since the OSM node numbers extend beyond what is allowed in Cube. The OSM network is in the WGS 1984 lat/long projection. This script must be run with 64-bit Python on a machine with at least 32 GB of RAM since it requires a substantial amount of memory to process the large XML file.

Inputs: `sf-bay-area.osm`

Outputs: `cube_node.csv`
`cube_link.csv`

3.2 buildOSM.s

Purpose: Build a Cube network from the link and node CSV files created by `osm_mtc.py`.

This script reads in the node and link files created above and creates nodes and links in Cube for any link with highway type equal to `footway`, `pedestrian`, or `steps`.

[4] Details are available here: http://wiki.openstreetmap.org/wiki/Key:highway

After creating the Cube network, the nodes and links need to be exported to a shapefile for use by subsequent scripts. The OSM network also needs to be re-projected from WGS 1984 lat/long to California State Plane VI 1983 Feet, in order to match the projection of the *Travel Model Two* roadway network. Once the nodes have been re-projected, the user needs to add an X and Y field to the node DBF file with the X and Y coordinate of the node in the new projection. This can be done in ArcGIS using the Calculate Geometry option in the Attribute Table window.

Inputs: `cube_node.csv`
`cube_link.csv`

Outputs: `osm.net`

`osm_nodes_sp.shp`

`osm_links_sp.shp`

3.3 osm_ped.py

Purpose: Build text files of the OSM pedestrian links to be add to the new MTC network.

This script snaps the OSM nodes to the new MTC network nodes where applicable and also creates new OSM nodes and links beyond the snap tolerance. OSM nodes are snapped to network nodes within a range of 50 feet. A few snap tolerances were tested, and 50 feet returned the best results. Only nodes that have a link with facility type (FT) equal to 4 (collector) or 7 (major arterial) are available for snapping. If both nodes of an OSM link snap to the same network node, the closest node is snapped, and the further away node is created as a new node. New nodes are created in the 7,000,000+ range to avoid collision with existing node numbers.

Inputs: `tana_mtc_nodes.dbf – new MTC network nodes`

`tana_mtc_links.dbf – new MTC network links`

`osm_nodes_sp.dbf`

`osm_links_sp.dbf`

Outputs: `osm_nodes.txt`

`osm_links.txt`

3.4 mergeOSM.s

Purpose: Merge the new OSM nodes and links to the new MTC network.

This script merges in the new OSM nodes and links to the new MTC network and sets the link `CNTYPE` attribute to '`PED`'. All new nodes have IDs in the 7,000,000+ range.

Inputs: `tana_sp_with_maz_taz_tap_centroids_connectors_routes.net`
`osm_nodes.txt`
`osm_links.txt`

Outputs: `tana_sp_with_maz_taz_tap_centroids_connectors_routes_osm.net`

3.5 osmToShapefile.s

Purpose: Save the interim pedestrian network to a shapefile.

This script saves the network produced by `mergeOSM.s` to a shapefile so that the pedestrian-street intersections can be determined in Python.

Inputs: `tana_sp_with_maz_taz_tap_centroids_connectors_routes_osm.net`

Outputs: `tana_sp_with_maz_taz_tap_centroids_connectors_routes_osm`
`.shp/shx/dbf/prj`

3.6 intersectPedWithStreets.py

Purpose: Create intersections between the street and pedestrian parts of the network.

This script intersects the pedestrian (`CNTYPE` = '`PED`') links with the street network and splits links at the intersections. Each intersection becomes a new node as well, with node numbers starting as 1+ the highest node number used thus far.

Inputs: `tana_sp_with_maz_taz_tap_centroids_connectors_routes_osm.shp`

Outputs: `link_split_ped.csv`
`link_delete_ped.csv`
`new_nodes_ped.csv`

3.7 mergeOSMSplitLinks.s

Purpose: Deletes the street and pedestrian links which intersect, and replaces them with split links.

This script deletes the links that are determined to intersect in `PedWithStreets.py`, and merges in the split link segments, as well as the new nodes that define them.

Inputs: `tana_sp_with_maz_taz_tap_centroids_connectors_routes_osm.net`

`link_split_ped.csv`

`link_delete_ped.csv`

`new_nodes_ped.csv`

Outputs: `tana_sp_with_maz_taz_tap_centroids_connectors_routes_osm_split.net`

3.8 buildOSMPedLinks.bat

Purpose: Executes all the necessary scripts to build the pedestrian network. Note that the batch file requires user intervention after the `buildOSM.s` script in order to export the OSM network from Cube to shapefile. There is a bug in Cube that is preventing the export of the node and link shapefiles via a script.

4 Skimming the Pedestrian Network

In addition to the new OSM pedestrian links, pedestrians are allowed to use connectors (`FT=0`), collectors (`FT=4`), arterials (`FT=7`), and bridges that allow bikes and peds (`BIKEPEDOK=1`). See the bicycle network memorandum[5] for more on the `BIKEPEDOK` link attribute. For calculating MAZ-to-MAZ impedances, pedestrians are not allowed to use TAZ connectors (`CNTYPE='TAZ'`) or TAP connectors (`CNTYPE='TAP'`). The `shortestPathPed.s` script will generate MAZ-to-MAZ pedestrian impedances according to these constraints.

Purpose: Generate a shortest path cost file between MAZs using the pedestrian network.

Inputs: `tana_sp_with_maz_taz_tap_centroids_connectors_routes_osm.net`

Outputs: `maz_ped_costs.csv,` a `CSV` file with fields `I`, `J`, `NODE`, `COST` (FEET)

[5] Please see the *Travel Model Two Development: Bicycle Ways* technical paper for details: http://analytics.mtc.ca.gov/foswiki/pub/Main/Documents/2013_09_16_RELEASE_Bicycle_Network.pdf