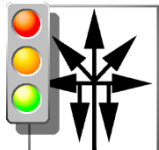


SUMO Tutorial

Jakob Erdmann

SUMO2016, Berlin



Knowledge for Tomorrow

Outline

- Prerequisites
- 3-Click scenario generation with `osmWebWizard.py`
- Fixing the network with Netedit
- Adapting and calibrating demand
- MESO - The mesoscopic simulation
- Multimodal and Intermodal scenarios
- Conclusion



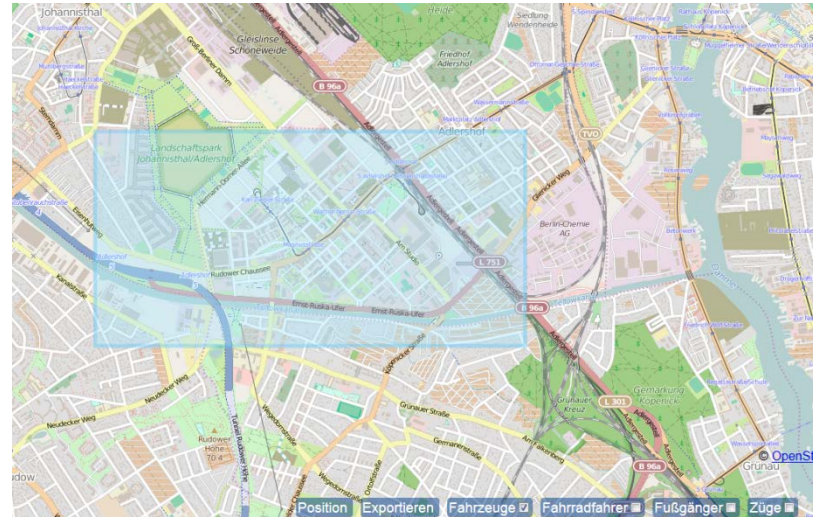
Prerequisites

- SUMO 0.26.0 or latest development version sumo.dlr.de/wiki/Downloads
- Python: www.python.org/download/releases/2.7/
- Text Editor (i.e. notepad-plus-plus.org/)
- Data files: sumo.dlr.de/daily/sumo2016_tutorial.zip
- Recommended:
 - Set environment variable SUMO_HOME
 - Add *.../sumo/bin* to environment variable PATH



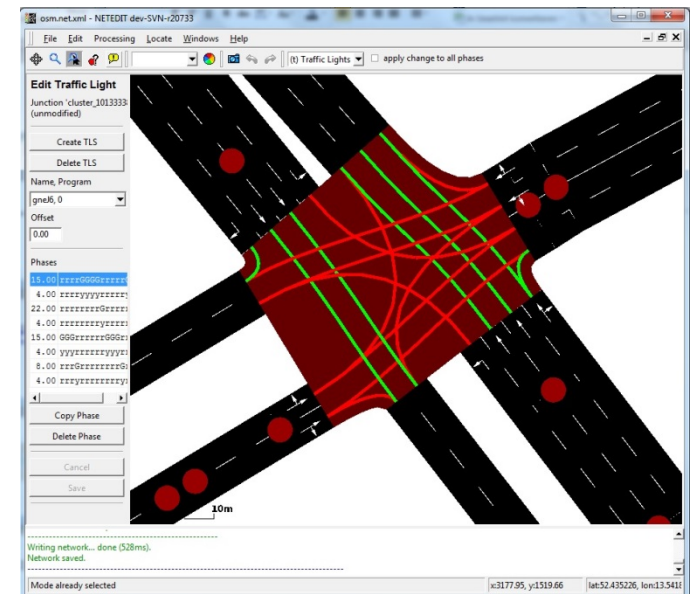
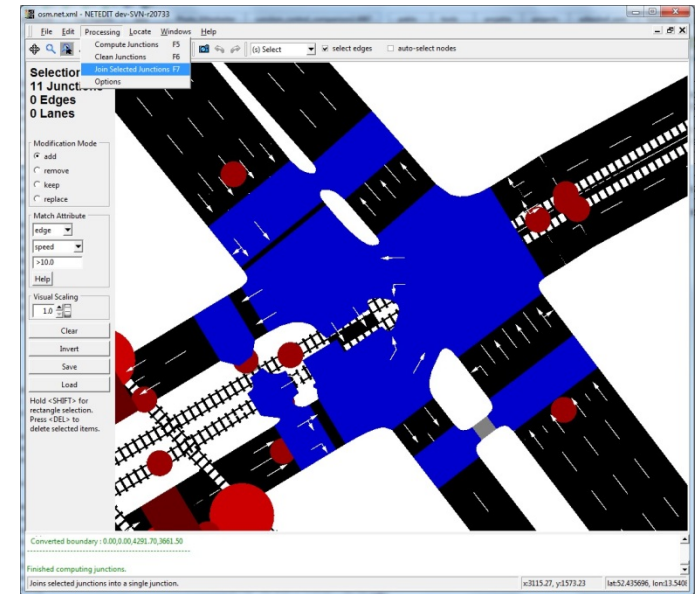
osmWebWizard

- Getting a basic scenario with [tools/osmWebWizard.py](#)
 - Mode-specific network options
 - Random traffic
- Configure
 - Area
 - Traffic modes
 - Traffic volume
 - Fraction of through-traffic
- Generated files allow rebuilding and adapting the scenario



Fixing network problems

- Open network in NETEDIT (ctrl-t in SUMO-GUI)
- Remove edges that disallow passenger cars
 - (selection mode, filter, delete)
- Join complex junctions (select, F7)
 - Save joined nodes
- Fix connections at major intersections
 - (connection mode)
- Adapt traffic light phase duration
 - (tls mode)
- Remove geometry-like junctions
 - `netconvert -s osm.net.xml -R -o osm.net.xml`



Adapting Demand

- regenerate random demand (edges changed when patching network)
 - Run **build.bat** (calls randomTrips.py with preset parameters)
 - `--period 0.5` (increased demand)
 - `--fringe-factor 8` (more traffic from network borders)
 - `--speed-exponent 2` (more traffic on fast edges)
 - `--lanes` (more traffic on multi-lane roads)



Calibrating Demand

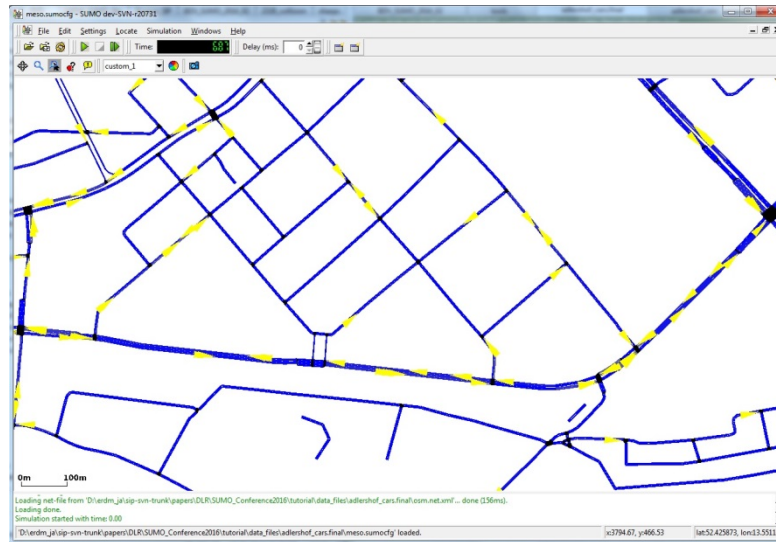
```
<routeProbe id="probe_-318210369#1" edge="-318210369#1" freq="60"
  file="calibrator_out.xml"/>

<calibrator id="cali_-318210369#1" lane="-318210369#1_0" pos="0"
  output="detector.xml"
  routeProbe="probe_-318210369#1">
  <route id="-318210369#1_fallback" edges="-318210369#1"/>
  <flow begin="0" end="1800" route="-318210369#1_fallback"
    vehsPerHour="3000" speed="20.0"
    departPos="free" departSpeed="max" color="blue"/>
  <flow begin="1800" end="3600" route="call_fallback"
    vehsPerHour="50" speed="5.0"
    departPos="free" departSpeed="max" color="blue"/>
</calibrator>
```



MESO

- Uses the same inputs as SUMO
- Running time of microsim ~15s (avg vehicle TimeLoss ~95s)
- Run scenario again with option `--mesosim ~1s`
 - MESO is fast!
 - TimeLoss < 1? Add option `--meso-junction-control`
 - -> TimeLoss 50.0.
 - MESO does not model vehicle acceleration, impact on urban dynamics



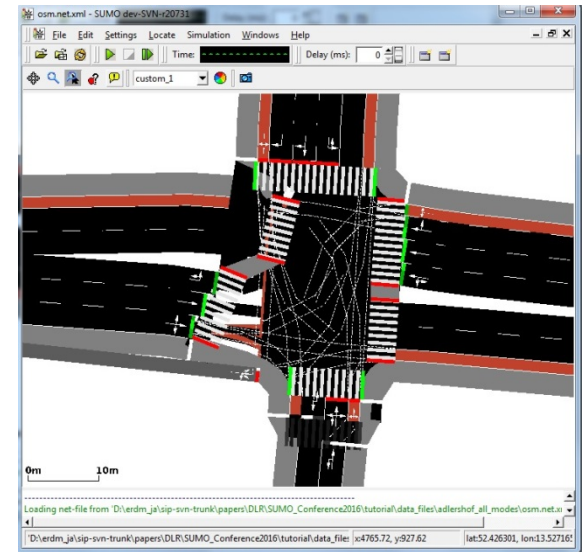
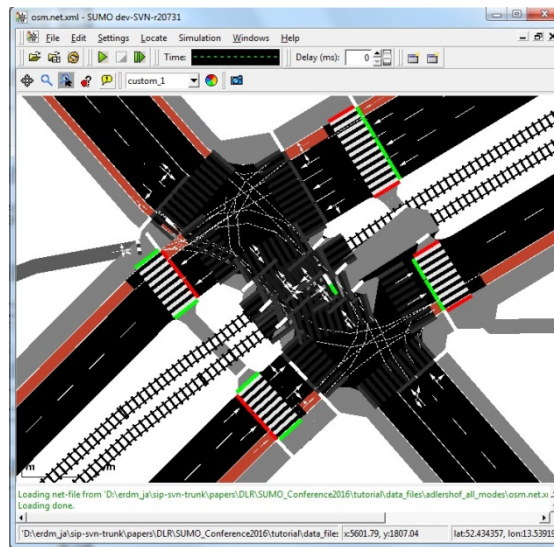
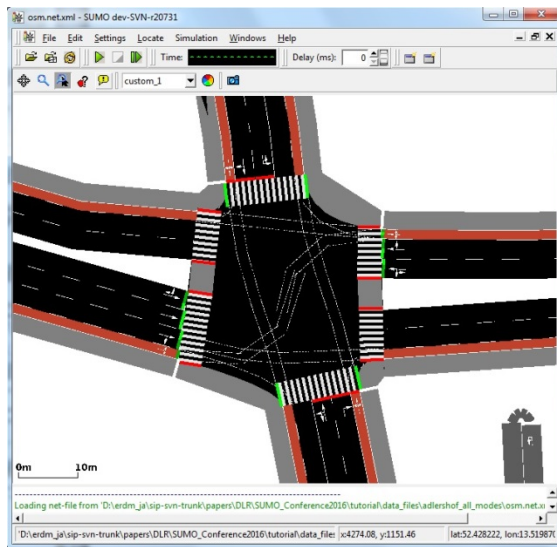
The current state of intermodal simulation (last years slide)

- Capabilities are ready in principle
- Some tools are still missing:
 - Intermodal demand import (which data sources?)
 - **Intermodal router** ✓
- TODOs
 - Enhance intermodal network import from OSM (and other sources?)
 - Explicit sidewalks
 - **Cycleways** ✓
 - Public transport
 - *Complex Intersections*
 - *Better support for bicycles* ✓
 - **Better support for trains (i.e. railway crossings)** ✓



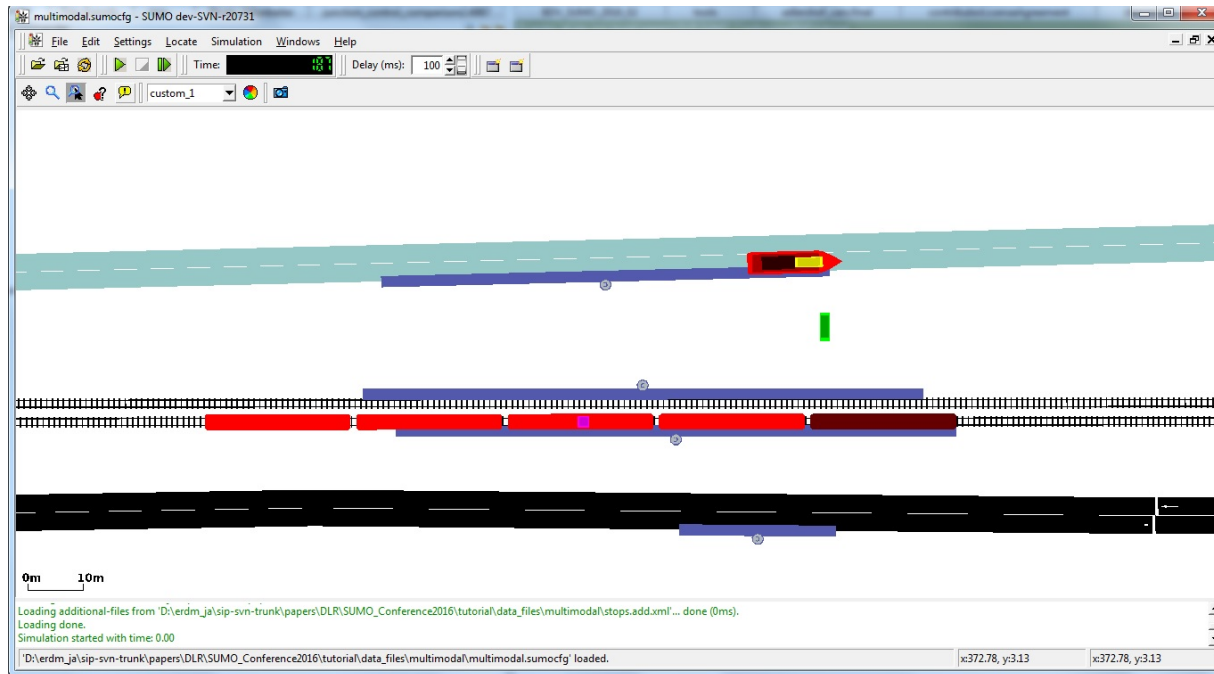
The current state of intermodal junctions

- The Good
- The Bad
- The Ugly



An intermodal logistics scenario

- Create a network with roads, railways and waterways
- Define rail road crossing
- Define `<containerStop>` elements
- Define multimodal demand including containers



Conclusion

- Use [tools/osmWebWizard.py](#) to get a quick start
 - Read the documentation / FAQ at <http://sumo.dlr.de/wiki>
 - Report any bugs you find to sumo-user@lists.sourceforge.net
 - Share your scenarios and results
-
- Talks to us. We are always looking for project partners! sumo@dlr.de

