ITS: Urban Mobility
Contents

• Reasons for an open-source traffic simulation
• Needed models, data, and its processing
• SUMO Overview
Open Source Road Traffic Simulation
Applications for a road traffic simulation

- reproducible, computerized evaluation of
  - real-world network performance
  - new concepts for
    - modern traffic signal control
    - traffic surveillance
    - traffic forecasting
    - traffic management
    - dynamic routing methods
    - car2car/car2infrastructure communication
      - traffic models
- instructive visualization
- planning
Open Source Road Traffic Simulation

Needed Components

road network  vehicles / flow  signalling
Open Source Road Traffic Simulation
Why an open source road traffic simulation?

Common procedure
• An academic organization wants to evaluate an aspect of traffic
• Builds an own traffic simulation; needs
  – A network representation including speed limits, right-of-way – rules, etc.
  – A representation of vehicles, their movement, routes, etc.
  – A representation of signals, variable message signs, etc.

... yields in
Many (incomplete) simulations which results can not be compared

→ **Solution:** an extendable simulation as a base for own development
Open Source Road Traffic Simulation
Design criteria

• Portability
  – Achieved by using C++ and portable libraries only

• Performance
  – High execution speed
  – No limitations in network size and number of simulated vehicles

• Extendibility

• Open source
  – Licensed under GPL
  – Hosted at sourceforge (http://sumo.sourceforge.net)

• Microscopic
  – Each vehicle is modeled explicitly
Road Traffic Simulation

Road Traffic Flow Dynamics (one of)

Fundamental diagram of traffic (counted through induction loops)

As interpreted by Kerner

As interpreted by Kim and Keller
Road Traffic Flow Simulation Classes

- Macroscopic
- Mesoscopic
- Microscopic
- Sub-microscopic
Road Traffic Flow Simulation
Microscopic Models

Mostly:

- discrete in time
- “Car Following Models”:
  vehicle’s speed depends on the preceding vehicle

$$d(v_l) + g \geq d(v_f) + v_f \tau$$
Road Traffic Flow Simulation
Car Following Model by Krauß*

Features:
• continuous in space
• discrete in time
• accident-free
• stochastic driver model

Parameters:
⇒ acceleration \( a \)
⇒ deceleration \( b \)
⇒ max. speed \( v_{max} \)
⇒ driver’s imperfection \( \epsilon \)

\[
v_{safe}(t) = v_i(t) + \frac{g(t) - v_i(t)\tau}{\frac{v}{b(v)} + \tau}
\]

\[
v_{des}(t) = \min\{v_{safe}(t), v(t - 1) + a, v_{max}\}
\]

\[
v(t) = \max\{0, \text{rand}[v_{des}(t) - \epsilon a, v_{des}(t)]\}
\]

Road Traffic Flow Simulation
Further Models needed

• Lane Changing
  – Navigation (I have to turn left on next junction)
  – Tactical (left lane allows me to move faster)

• Interaction with the network
  – Stops at red traffic lights
  – Decelerate if other vehicles have a higher right to move on
     the next junction

• Extensions
  – Vehicle classes, such as busses, which use own lanes
  – Bus stops
Microscopic Road Networks
Needed vs. given Information

Needed street attributes:
- Number of lanes
- Allowed speed
- Per-lane restrictions
- Allowed continuations

Given: a graph with only few information
→ Number of lanes (often vague)
→ Allowed speed (often vague)

Needed junction attributes:
- Right-of-way
- Traffic lights program
Microscopic Road Networks
Size and Complexity

Additional problems:
• very large networks
• complex junctions
Example: computation of lane-to-lane connections:

1. for each edge: compute turnaround edges
2. for each node: sort each node’s edges
3. for each node: compute each node’s type
4. for each node: set edge priorities
5. for each edge: compute edge-to-edge connections
6. for each edge: compute lanes-to-edge connections
7. for each node: compute lane-to-lane connections
8. for each edge: recheck lanes
9. for each edge: append turnarounds
Traffic Demand
Needed Information

Each vehicle is modeled explicitly

mandatory attributes:
  – ID (name)
  – Vehicle type (including Krauß-parameter)
  – Complete route through the network
  – Depart time

optional attributes:

```xml
<vehicle id="bus100_west_0d" type="BUS" depart="0" color="1,0,1">
  <route>-572658025 -572658026 -572658027 -572658024 ...
  </route>
  <stop bus_stop="west1" duration="20"/>
  <stop bus_stop="west2" duration="20"/>
</vehicle>
```
Traffic Demand
Possible Sources

- Real (physical) measures are done at induction loops
  - Accurate but sparse
  - No information about who is driving from which place to which place, only how many vehicles have passed a certain place
- Counts at junctions may provide flows and turning ratios
  - Less accurate than induction loops and sparse, too – mainly for some major inner-city junctions
  - Information about flow spread on junctions
- Traffic scientists estimate demands on district level
  - Even less accurate (estimated), but cover a complete area
Traffic Demand
Possible Sources

→ Real (physical) measures are done at induction loops

induction-loop measures → DFR ROUTER → SUMO-routes

→ Counts at junctions may provide flows and turning ratios

turning ratios → JTR ROUTER → SUMO-routes

flows

→ Traffic scientists estimate demands on district level

OD-matrix → OD2TRIPS

OD2TRIPS

trips → DUAR ROUTER → SUMO-routes
SUMO - Modules
Included Applications

- **SUMO**: command line simulation
- **GUISIM**: simulation with a graphical user interface
- **NETCONVERT**: network importer
- **NETGEN**: network generator
- **OD2TRIPS**: converter for O/D-matrices
- **JTRROUTER**: router based on turning ratios
- **DUAROUTER**: router based on a dynamic user assignment
- **DFROUTER**: router which uses detector data
SUMO Features

- Microscopic: all vehicles are modeled explicitly
- Time-discrete, space-continuous car-following model by S. Krauß
- Multi-lane traffic, right-of-way rules
- Around 100,000 cars in real-time (without graphical output)
- Traffic lights with time schedules, other traffic management devices
- Complex networks / Import support
- Dynamic Routing based on Dynamic User Assignment
- Other routing modules

Projects at the DLR where SUMO was used
2002-2006

• INVENT
  Implementation and verification of traffic management strategies for large urban areas
• OIS
  Verification of benefits arising from usage of new optical sensors
• Traffic Tower
  Virtual Traffic Management Environment
• WJT2005 / Soccer2006
  Integration of standard and airborne detectors into a traffic portal with forecast functionality
• TrafficOnline
  Traffic surveillance via in-vehicle GSM phones
WJT2005 / Soccer2006 → DELPHI

Description

• Used in the city of Cologne during
  – The pope’s visit (world youth day 2
  – The world soccer cup 2006
• Traffic surveillance using
  – Highway induction loops
  – Inner-city induction loops
  – Airborne traffic recognition system
    (mounted on a zeppelin)
• Traffic visualization
  – Integration of gained information into a
  – Presentation for the police
• Traffic forecast
  – 30 min into the future using a simulation
    (extended SUMO)
DELPHI
System Overview

ARGOS:
Access via db, webserver

Induction loops:

Floating Car Data:

Simulation:
• Areal extrapolation of measured data
• Forecast generation (30min)
• Fed from db using python

Datenbank:
• MySQL with InnoDB tables
• No built-in logic

Aggregation/Correction:
• Guessing missing values
• Data fusion
• Done using python scripts

Webserver:
• Tomcat Servlet Container
• Visualization (Images and traffic)

Webbrowser:
• JavaScript Browser
• asynchronous XML-Requests

Datenbank:
• MySQL with InnoDB tables
• No built-in logic
SUMO – Example Results
Flow Comparison
SUMO
Tasks ToDo

• Check the readings section in course webpage
• Download and install the right version of sumo on your computer.
• Try to run it to be ready for the lab session