SUMO's Road Intersection Model

Jakob Erdmann, Daniel Krajzewicz

Institute of Transportation Systems,
German Aerospace Center, Berlin, Germany,
{jakob.erdmann,daniel.krajzewicz}@dlr.de

Abstract. Besides basic models for longitudinal and lateral movement, a traffic simulation needs also models and algorithms for right-of-way rules. This publication describes how passing an intersection is modeled within SUMO, including a description of an earlier and the currently used model.

Keywords: Road Traffic Simulation, Intersection Model.

1 Introduction

SUMO [1, 2] is an open source road traffic simulation package developed at the Institute of Transportation Systems at the German Aerospace Center. SUMO is a microscopic traffic simulation which means that each vehicle is modeled explicitly. Independently configurable models are used to define different aspects of each vehicle's driving dynamics. Multiple car-following models which describe longitudinal movements are implemented and there are also multiple lane-changing models for defining lateral movements. The models used in SUMO were initially described in [3]. For simulation of real-life networks, further models are necessary. This paper describes the current implementation of the intersection control model used in SUMO. An earlier less detailed model which was tied to a fixed simulation time step length of 1 second is also described. The current model was implemented in order to overcome these limitations.

The rest of this document is structured as following: In section 2, the original and the currently used model for intersection control are described. One aspect of the intersection model is the estimation of time windows in which a vehicle will occupy the intersection. The accuracy of this estimation is the subject of the section 3. In section 4 we present our conclusions.

This documented presents an extension to the description of the intersection model given in [4]. It describes new features such as the modeling of stop signs and the intersection type all-way stop. Additionally, the revised dynamics in regard to link leaders is described. The topics of driver impatience and blocked intersections are further additions to the preceding work.

2 Intersection Model

Generally, road networks are represented as graphs in SUMO. An intersection ("node") consists of incoming and outgoing edges, where an "edge" represents a road with one or more lanes. Each lane has a unique id which is derived from the edge id and the numerical index of the lane starting with 0 at the rightmost lane. The lanes of incoming edges are called incoming lanes and the lanes of outgoing edges are called outgoing lanes. Within an intersection, lie so called "internal lanes" which connect the incoming lanes with the outgoing lanes. Vehicle movements across an intersection proceed along these internal lanes just as they would on regular lanes. Internal lanes were added to the simulation to obtain fine grained vehicle trajectories and to better model the capacity of intersections.

A given lane may have more than one successor lanes. The connectivity among lanes is defined with "links". This gives rise to the "lane-graph" where lanes are connected by links in contrast to the "intersection-graph" where intersections are connected with roads.

In older versions of SUMO, before the introduction of internal lanes, there was a link between an incoming lane and an outgoing lane. Vehicles would seemingly "jump" across an intersection. Since the introduction of internal lanes there is a link between an incoming lane and an internal lane (called an "entry link" and another link between the internal lane and an outgoing lane (called an "exit link") as shown in **Fig. 1**. The entry links of an intersection are numbered from 0 to n. Since there is exactly one exit link for each entry link, the link index uniquely defines a connection across the intersection from an incoming lane to an outgoing lane. The link indices are computed using the following scheme: first, the incoming edges are sorted in clockwise fashion. Then, the lanes, starting at the top-most are traversed. The links outgoing from a lane are then iterated, starting with the right-most destination, relative to the incoming edge. These link indices are used to uniquely identify a connection when specifying right of way rules.

At most intersections, vehicles wait at the stop line at the border of the intersection until they may cross conflicting streams of traffic. However, on some types of intersections, left-turning vehicles are allowed to wait in the middle of the intersection. This is modelled in SUMO by splitting internal lanes at the halting position and introducing an "internal intersection" that lies within the original intersection. Vehicles using these internal lanes always pass the entry link to the intersection and then wait at the internal intersection instead. The right-of-way computation for internal intersections follows the same principles as that of regular intersections.

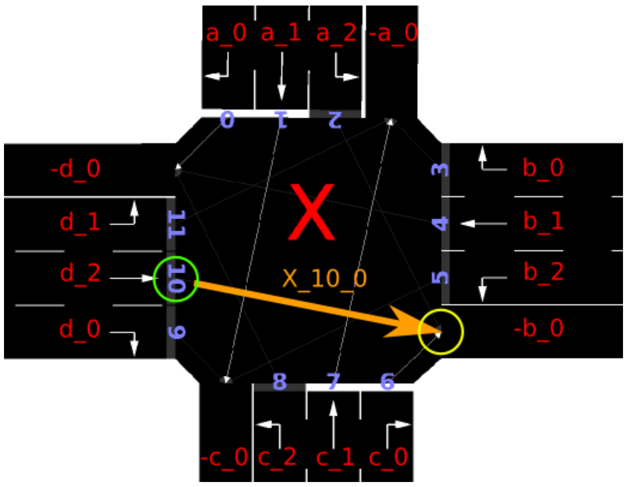


Fig. 1. Intersection model terminology in SUMO. The intersection has id X and features the incoming roads a, b, c, d and the outgoing edges, -a, -b, -c, -d. The connection from incoming lane d_2 to outgoing lane –b_0 crosses X on the internal lane X_10_0. The entry link with index 10 is circled in green. The exit link is circled in yellow.

2.1 Earlier Model

The right-of-way model that was implemented in the initial release of SUMO is a strong simplification of real world behavior. When approaching an intersection a vehicle at first sets the information about its approach to the intersection. After this has been done for all vehicles, the intersection "decides" which vehicles are allowed to pass without braking and which vehicles have to yield. This is done using a right-of-way matrix. This matrix describes which connections cross each other and which one has the right of way in case of crossing connections. This concept is illustrated in the following using an example.

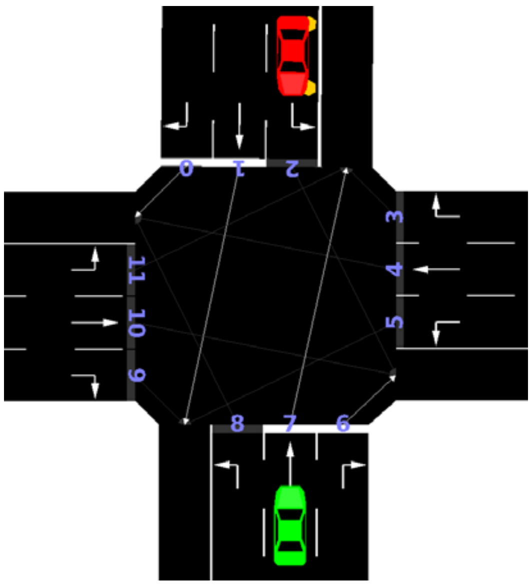


Fig. 2. Two vehicles (red and green) are approaching intersection X from **Fig. 1**. The red vehicle has to yield to the oncoming green vehicle on link 7 before it can make a left turn on link 2.

Fig. 2 shows an intersection which is approached by a red vehicle on link 2 and a green vehicle on link 7. Since the paths of both vehicles intersect and both wish to cross the intersection in overlapping time intervals, a right of way computation is performed. In **Fig. 3** the right-of-way matrix for this intersection is shown, emphasizing links 2 and 7. The matrix cell with row i and column j defines the right of way for a vehicle approaching link *i* (called the "ego"-vehicle) in regards to another vehicle approaching link *j* (called the "other"-vehicle). Depending on the colors white, yellow or red, the ego vehicle ignores, has priority over or yields to the other vehicle. In the example, the red vehicle (link 2) yields to the green vehicle (link 7) because of the red box in cell (2, 7) which agrees with the common rules of traffic for left-turning vehicles. This is mirrored by the yellow cell (7, 2) which indicates that the green vehicle has priority over the red vehicle.

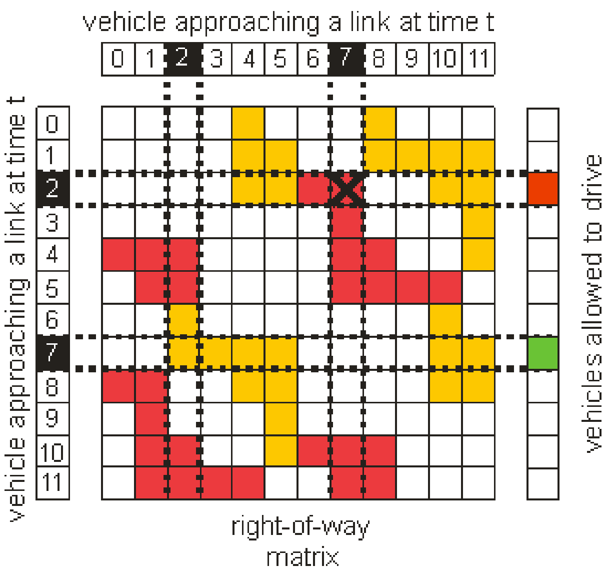

vehicle approaching a link at time t
0
1
2
3
4
5
6
7
8
9
10
11
vehicle approaching a link at time t
vehicles allowed to drive
right-of-way matrix

Fig. 3.: The right-of-way matrix of the intersection shown in **Fig. 2**. Row i corresponds to the crossing/priority relation for link i. Link 7 crosses links 2, 3, 4, 5, 10, 11 but has the right of way (yellow boxes). Link 2 crosses links 4, 5, 6, 7, 10, 11 but must yield to 6 and 7 as indicated by red boxes. Since a vehicle approaches on link 7 (in the relevant time interval) the vehicle on link 2 has to brake.

The matrix itself is static and computed during the network import/generation. In the earlier intersection model, traffic lights were implemented by removing the information about vehicles approaching links with a red signal. As the participation in determining which vehicles may drive is invalidated for these links by doing so, these vehicles are not allowed to pass the intersection and also do not hinder vehicles at other links.

Even though this model works well for simulation steps of one second, it caused problems when implementing sub-second time steps. Because the decision about letting a vehicle pass the intersection is performed in each time step, vehicles must not drive faster than their maximum braking ability multiplied with the step size time when being in front of the intersection. This is necessary to ensure that the vehicle can still brake if another vehicle with higher priority suddenly approaches. When decreasing the duration of simulation steps, this velocity is decreasing by the same factor, too, as depicted in **Fig. 4**.

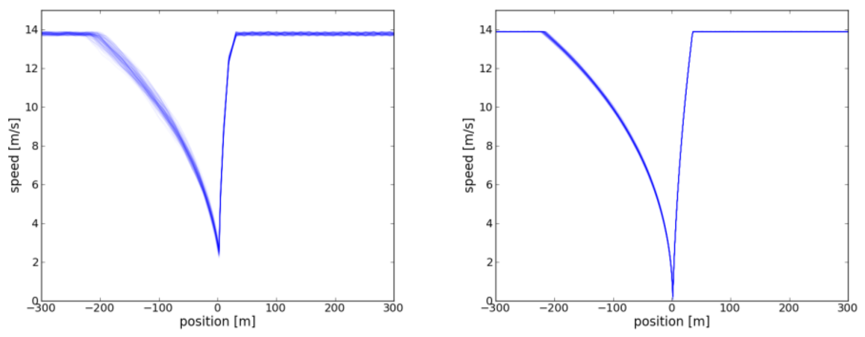


Fig. 4.: Vehicle speed when approaching an intersection in the old model; a) simulation steps of 1s, b) simulation steps of 0.1s.

This wrong behavior for lower step times was the motivation to change the intersection control algorithm.
Another important motivation was the need to model the interaction between vehicles which occupy the intersection simultaneously. This became necessary after the introduction of internal lanes on which vehicles may decelerate or even stop. This requires other vehicles to react in order to avoid collisions.
From an architectural standpoint, transferring the logic for passing an intersection from the intersection model into the driver model is assumed to be a development step into the right direction, allowing further work on driver behavior modeling.

2.2 Requirements for an improved Model

The goal for an improved intersection control model was to support all types of intersection typically found around the world and to allow for realistic simulation dynamics. The following intersection types are deemed necessary:

- Intersections without prioritization
  - right-before-left
  - all-way stop
- Prioritized intersections with
  - Different directions of the prioritized road (straight, turning),
  - Unprioritized roads with yield or stop signs,
- Intersections controlled by traffic lights.

Important aspects of realistic intersection dynamics are the following:

- No deadlocks,
- No collisions,
- Efficient use of the intersection,
- Realistic acceptance gaps,
- Approaching unprioritized links without stopping,
- Qualitative dynamics independent of the simulation step length.

The current intersection model meets all these goals.

2.3 Current Model

In this section we describe features which distinguish the current intersection model from the previous intersection model. These features were implemented over a time frame of more than 10 years to fulfill a growing list of requirements. The complete specification of all implemented of formulas and decision trees cannot be given due to lack of space. However, the described concepts should serve as a useful guide when reading the implementation sources of SUMO [5].

Approaching an Intersection.

The key to correct the deficiencies of the original model described in 2.1 was to not only consider the current time step, but to give the right of way based on information about oncoming vehicles including an extrapolation of their time of arrival at the intersection into the future. To do so, each vehicle informs the entry link about its approach. In contrary to the initial model, not only the approach as such is stored, but also the expected time of arrival at the intersection and the speed of this arrival. Using this information, the time within which a certain link over the intersection is occupied can be computed. Each entry link also stores information about its "foe links". This corresponds to the red boxes in one row of the right-of-way matrix shown in **Fig. 3**. When approaching an intersection (an entry link), a vehicle computes how long it will occupy the intersection and then checks against all approaching vehicles in all foe links of its entry link. If the requested time slot is separated from all approaching foe time slots by a suitable safety gap the vehicle is allowed to pass the entry link and thus enter the intersection. When the ego vehicle and the approaching foe vehicle have the same target lane, safety depends on the speed difference between both vehicles.The formula which determines whether a given situation is safe uses the same idea as that used in car following, namely that the follower vehicle *F* with velcocity v_F and deceleration d_F needs to be able to stop before the leader vehicle *L* (with velocity v_L and deceleration d_L) does:

$$v_L^2 / d_L > v_F^2 / d_F \qquad (1)$$

A vehicle informs the entry links to the next few intersections on its current route (up to a distance of about 3000 m) about its approach. Due to the advance knowledge of approaching foe vehicles, a vehicle approaching on an unprioritized link cannot be “surprised” by the sudden appearance of a foe. This allows decoupling the approach speed from the simulation step size. Instead, vehicles decelerate up to a fixed distance from the stopping line (default 4.5m, corresponding to the default vehicle deceleration of 4.5m/s^2). If braking is not necessary at this point they can safely accelerate and cross the intersection. Otherwise they stop until there is a suitable gap in traffic.

Fig. 5 shows that using this implementation assures similar behavior for different simulation step sizes. The velocity used for approaching the intersection is the vehicle's deceleration capability multiplied with 1 s. For the standard Krauß parameters it is equal to 16.2 km/h, what was found to be empirically valid when compared to measures obtained from test drives with DLR test vehicles. Within the current model, the vehicle's maximum deceleration ability is used for all intersections and all directions of driving across them. Because in reality, this speed is mainly dictated by the possibility to look into foe lanes for determining whether the intersection may be crossed, further extensions of the model, in means of differing between approach velocities promise to improve the model's quality. It should be also noted that the simulated time line of deceleration and acceleration is not yet matching the reality.

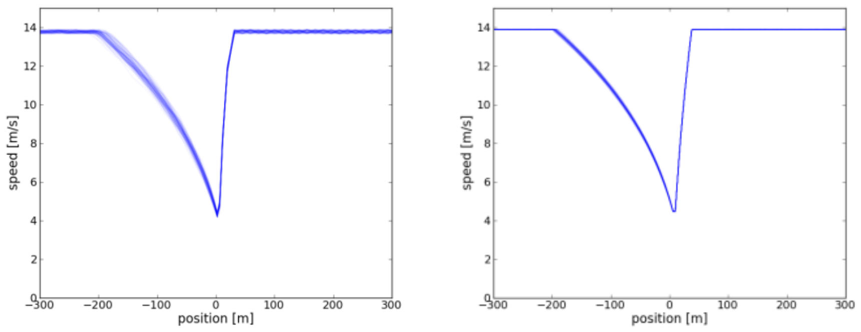


Fig. 5. Vehicle speed when approaching an intersection in the new model. a) simulation steps of 1 s, b) simulation steps of 0.1 s.

Dynamics within an Intersection

Once a vehicle enters an intersection by passing the entry link, this link is no longer informed. Since vehicles follow normal movement rules while on the intersection they may brake while on the intersection or even come to a stop. Therefore, other vehicles require an additional mechanism for keeping track of vehicles currently on the intersection in order to avoid collisions. The new implementation is designed to re-use the existing functionality of the car-following model for letting vehicles maintain safe distances while interacting within the intersection. Normally, this functionality is only active for vehicles which move on identical or subsequent lanes. At an intersection however, vehicles are on different lanes which cross somewhere on the intersection or merge into the same outgoing lane.

To be able to use the car following functions two things are required

1. A vehicle needs to know the lead vehicle;
2. There must be a well-defined distance between the follower and the lead vehicle.

The first point is accomplished by declaring the first vehicle of any two vehicles to enter the intersection as the leader. This is particularly important, because several vehicles may be driving within the space of the intersection at the same time and there must be a non-circular leader-follow relation among them to avoid deadlocks (technically speaking, all vehicles must be in an antisymmetric, transitive and irreflexive relation). The second point is accomplished by virtually superimposing both internal lanes up to the crossing point. If both internal lanes merge into the same outgoing lane, the crossing point is naturally the beginning of the outgoing lane.

We describe the position of the crossing point relative to the start of the respective internal lanes. Let *A* denote the lane on which the leader vehicle drives and let *B* denote the lane on which the follower vehicle drives. The position of the crossing point of *A* and *B* on lane *A*, $pos(C_{AB})$ is the driving distance from the start of lane *A* to the geometrical crossing point (disregarding the width of the lanes). Likewise, $pos(C_{BA})$ is the position of the crossing point on lane *B*. Furthermore, *pos(L)* is the position of the front of the leader vehicle *L* relative to the start of *A* and *pos(F)* is the position of the follower vehicle *F* relative to the start of *B*. We define the distance of *L* from the crossing point d_L and the distance of *F* from the crossing point d_F

$$d_L := pos(C_{AB}) - pos(L)$$

$$d_F := pos(C_{BA}) - pos(F)$$

The distances d_L and d_F are visualized in in **Fig. 6**. Using this notation, the virtual gap *g* between both vehicles is defined as

$$g := d_F - d_L - length(L) - minGap(F)$$

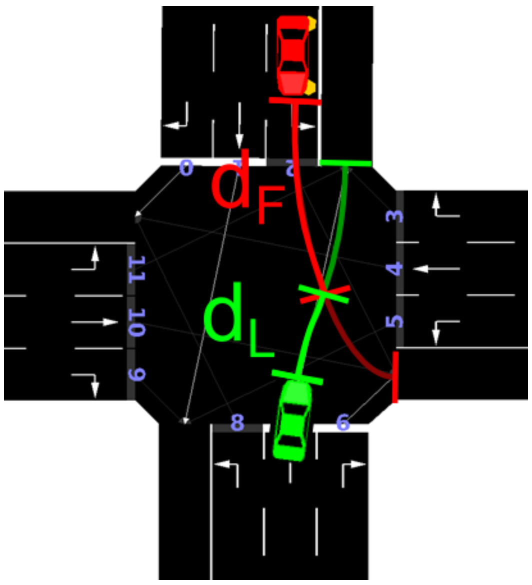


Fig. 6. Distance d_L of the leader vehicle (green) and d_R of the follower vehicle (red) to the crossing point of their future trajectories.

Where *length(L)* is the physical length of vehicle *L* and *minGap(F)* is the minimum gap that vehicle *F* intends to keep to its leader at all times. Note that *g* may be negative which causes vehicle *F* to stop.

In the current implementation each exit link maintains a list of “foe internal lanes”. These are the lanes which correspond to the yellow and red boxes in one row of the right-of-way matrix in **Fig. 3**. In other words, these are the internal lanes which intersect with the internal lane the approaching vehicle intends to use.

A vehicle that wishes to pass an exit link on its route asks this link for any additional vehicles to which it must adapt its speed. These vehicles are called link leaders. The link checks all of its foe internal lanes for occupancy computes the virtual gap and returns each found vehicle as a potential link leader to be followed.

Fig. 7 shows the same intersection as **Fig. 1** with three vehicles green (G), blue (B) and red (R). Vehicle R wishes to pass the exit link that belongs to link 11. Both vehicles G and B are on the same internal lane which is a foe internal lane for link 11. On the left side of **Fig. 7**, vehicles G and B are potential link leader for R. Since G and B have entered the intersection before R, they will both be followed. In this case only the speed adaption to B is relevant since B is already following G. On the right side of **Fig. 7** the situation is slightly different. Vehicle R has already entered the intersection before vehicle B and therefore, R only follows G.

In the current implementation each vehicle maintains a list of link leaders being followed for each exit link. This list is used when maintaining the antisymmetric link leader relation among vehicles (vehicle R only sets vehicle B as its link leader if B does not already have R as its link leader).

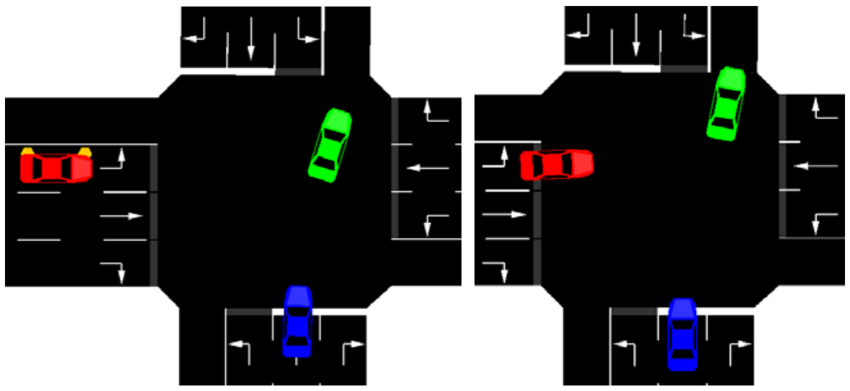


Fig. 7. Examples of link leader relations for the vehicles green (G), blue (B) and red (R). In the left figure G is the leader of B and both are the leaders of R. In the right figure G is the leader of R and R is the leader of B because R entered the intersection before B.

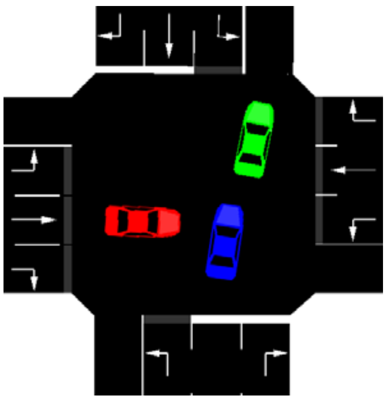


Fig. 8. The red vehicle may drive up to the point where its trajectory intersections with those of the green and blue vehicles.

The use of the car-following model gives appropriate results if the vehicles *L* and *F* approach the same outgoing lane. However, if the vehicles approach different outgoing lanes and their trajectories cross somewhere on the intersection, the car-following model only returns an upper bound on the collision-free speed. This can be easily understood when considering the situation shown in **Fig. 8**. Here the red vehicle (R) is on a west-east trajectory across the intersection If the green and blue vehicles (G, b) are stopped on the intersection, R may drive up to the crossing point without causing a collision even though the virtual gap *g* may be negative (the lengths of G and B which influence *g* are irrelevant here). To reflect this situation, the implementation lets vehicles drive with a speed that is as least as high as the maximum safe speed for stopping at the point where the trajectories cross. The computation of that speed is again left to the car-following model.

The link based model of detecting conflicting approach information coupled with the handling of link leaders allows for full vehicle dynamics on the intersection together with efficient use of the intersection as a natural extension of car following.

Additional Intersection Types

Another recent extension to the intersection model of SUMO was the ability to model additional intersection types. Intersections of the type “priority_stop” follow the same right of way rules as prioritized intersections but they require vehicles on minor roads to come to a complete stop before passing the intersection. This is accomplished by checking the “waitingTime” of the respective vehicle and forcing it to slow down unless that value is positive. The waitingTime of vehicles is among the common vehicle attributes being tracked anyway. It is incremented every time the vehicle’s speed is equal or below a fixed threshold of 0.1m/s and reset to 0 every time the vehicle’s speed is above that threshold.

Intersections of type “allway_stop” also force the vehicles to slow down unless they have a positive waitingTime. Additionally, the waitingTime of each vehicle is recorded as part of the approach information. The right of way between two conflicting vehicles is given to that vehicle which has a higher waitingTime.

Impatience

In the previous section it was described that the ego vehicle enters an intersection if its expected time frame of intersection occupancy is sufficiently distinct from the usage frames of all foe vehicles with higher priority. This is necessary to ensure a collision free intersection model. However, there is yet another degree of freedom which must be considered in this decision. Just as in reality, the ego vehicle may enter the intersection "aggressively", forcing foe vehicles to brake hard to maintain safety relations. Or it may refrain from entering the intersection until it can do so without disturbing other vehicles at all. Between these extremes lies a continuum of behaviours which all satisfy the requirements of a safe intersection model. We model this continuum using the term 'impatience' which is a real value from the interval [0, 1]. Vehicles with an impatience of 0 avoid actions which require other vehicles to slow down while vehicles with impatience 1 will enter an intersection even if other vehicles need to employ maximum deceleration to ensure safe driving distance. Generally speaking, a vehicle with impatience α will enter an intersection if it expects to leave it a time *t* and

$$t < (1 - \alpha) \times a_{min} + \alpha \times a_{max}$$

where a_{min} is the earliest time of the foe vehicles arrival at the intersection and a_{max} is the latest time of the foe vehicles arrival (using maximum deceleration). If the foe vehicle can come to a full stop before the intersection a_{max} is set to a constant value of *C* seconds to prevent α from becoming meaningless by having infinities in the equation. The current model uses *C*=30 because it was found to work well but there has not yet been a deeper evaluation of alternatives.

The choice of *α* for the simulation has important implications for the fluidity of traffic. If the value of *α* is too low, vehicles on an unprioritized road may be blocked from driving indefinitely while there is heavy traffic on the main road. If the value of *α* is near 1 traffic on the main road will be frequently disturbed in a manner that does not fit real world experience.

To avoid these problems, the value of *α* is dynamic in the simulation. The very name 'impatience' has been chosen because it implies something that grows over time. In the simulation the value of *α* is defined dynamically as:

$$\alpha := \mathrm{MAX}(0, \mathrm{MIN}(1, \alpha_C + \mathrm{waitingTime} / T))$$

where α_C is a vehicle type specific constant defaulting to 0 which can be configured in the range [-1,1] to model the base level of driver impatience. The extreme values of α_C result in constant values for α of 0 and 1 respectively. The value of T is a configurable simulation constant which governs the time after which stopped vehicles will be removed to clear deadlocks (defaulting to 300 seconds). The value of T can be taken to model that maximum waiting time that vehicles will typically tolerate. If this time is exceeded the model is in an erroneous state (typically because a deadlock has developed) and this state is cleared by removing vehicles and inserting them at a later point on their route. Using this definition of α, impatience of a vehicle (or rather its driver) grows while it is waiting to pass a link. This avoids major disturbances of traffic on the main road in most cases but allows some disturbance where necessary to avoid completely blocking the unprioritized road.

Avoiding Blocked Intersections

According to traffic laws around the world [6, 7] it is forbidden to enter an intersection if there is a danger of not being able to pass the intersection and consequently blocking cross traffic. As this rule requires some judgement about the probable behaviour of downstream traffic and compliance to traffic laws is not perfect, blocked intersections do still occur in reality.

With the introduction of internal lanes to the intersection model of SUMO, the issue of blocked intersections must be addressed in the simulation as well. For every vehicle approaching an intersection, there is a check whether the vehicle will be able to leave the intersection and thus may begin entering the intersection. If this check fails the vehicle stops before the intersection and repeats the check every simulation step. As a side effect of a failed check, the stopping vehicle no longer prevents vehicles with lower priority from crossing its path. This is important because otherwise a jam on the priority road would immediately extend to any roads that cross it.

Just as in reality it is not a trivial thing to decide in advance whether a given vehicle will be able to leave the intersection. In the following we describe the heuristic that is used to prevent vehicles from blocking an intersection. The ego vehicle has a certain space requirement *s* which consists of its physical length and the minimum standing gap to its predecessor vehicle, which must be met behind the intersection. The length of a vehicle and the size of the minimum gap are configurable attributes of a vehicle. The minimum gap is the distance which a vehicle must keep to its predecessor when both vehicles are stopped. If there is a stopped vehicle behind the intersection which leaves less than *s m*eters of space behind the intersection the check immediately fails. This idea is extended to cover cases where the leader vehicle is still moving.

For every simulation step there is a look-ahead horizon which the ego vehicles uses to plan future movements (mainly in regard to maintaining safe velocities and to follow its route). As a result of this horizon it is known which lanes the ego vehicle will drive on in the next simulation steps unless a lateral change of lanes occurs. Along these lanes a check is done until the first stopped vehicle is found or the lanes are exhausted. Along the way, the available space (lengths of the regular lanes) is accumulated as well as the space requirements of moving vehicles (length and minimum standing gap). The check succeeds only if the available space is sufficient to meet the combined space requirements of the ego vehicle and the moving vehicles up to the next stopped vehicle or the end of the look-ahead horizon. During the forward search along the future lanes of the ego vehicle two more things are treated just like stopped vehicles: closed links and vehicles which are about to stop. A closed link is one that prevents vehicle movements such as a red light or a minor road with incoming priority traffic. Whether a vehicle is about to stop is determined by checking that vehicle's tail lights and by checking whether the vehicle was flagged as about to stop in the previous simulation step. These flags are a technical solution for improving a driver’s ability to anticipate the behavior of fellow drivers and can be thought of as modeling a driver’s situational awareness.

It should be noted that the check by the ego vehicle is performed for all intersections along the look-ahead horizon. It may well be possible that the ego vehicle can safely enter the first intersection but has to slow down already because it may not enter the intersection after that. Another thing that must be considered is a minimum length for the look-ahead horizon. For reasons of avoiding collisions a vehicle only needs to look ahead as far as its current braking distance. However, to detect whether there is enough space ahead for leaving an intersection a larger look-ahead is sometimes needed. The current simulation model sets the minimum look-ahead distance to 5 times the length of the vehicle because this was found to work empirically. Taken together these heuristics prevents blocked intersections in most situations. The few remaining failure cases are characterized by the emergence of jam conditions immediately after a successful check.

3 Estimation of Link Entry/Exit Times

In the following, the estimation of times and speeds of arrival and leaving an intersection is discussed. **Fig. 9** shows the deviations of the estimated speeds and times over time for a major (high prioritized) link. These vehicles do not have to break. "deviation" denotes here the difference value obtained by subtracting the real from the estimated value in the following Figures. One may see that the times of arrival and leaving are both estimated too low and only increasingly move towards the correct value. This is due to the random "dawdling" behavior of SUMO's default car-following model, see [8]. If the dawdling is disabled, the estimation is correct from the very begin on (not shown, here). The deviations in time are due to the same reason. They are straight, as in each time step, the estimation is based on the perfect speed (50 km/h in the shown example) and the dawdling is performed by the model afterwards. It should be noted, that the estimation could be more correct, if the dawdling, regarding its stochastic nature, would be taken into account during the computation of the times/speeds.

The additional error in estimating the leave time is probably due to taking into account the distance to the leader in jam/when standing (SUMO's "minGap" attribute of a vehicle type), which was set to 2m in the shown example; 2 m divided by 13.89 m/s gives the offset shown here, which is about 0.14 s.

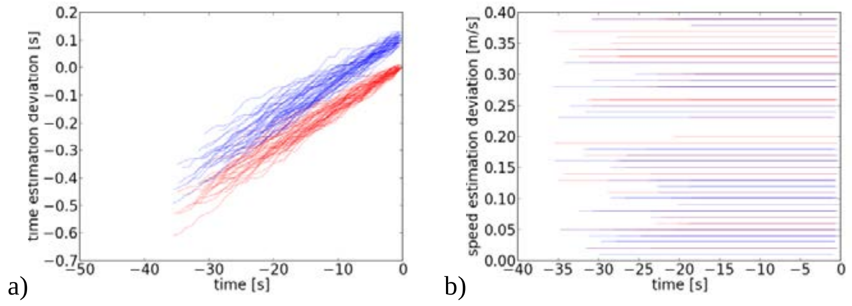
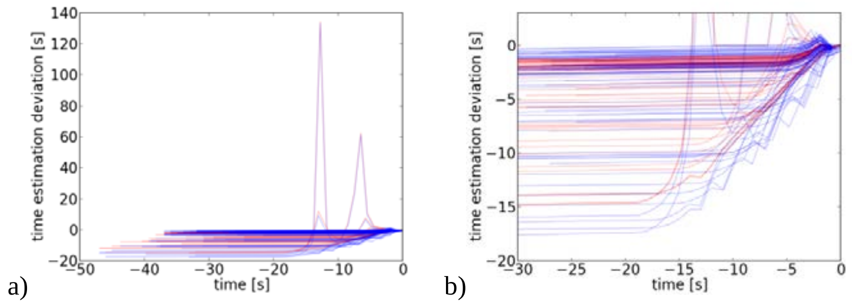
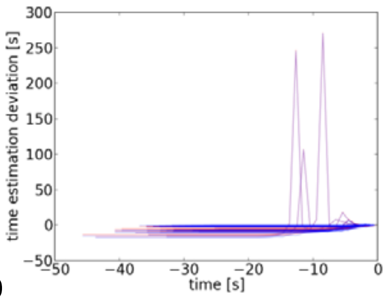


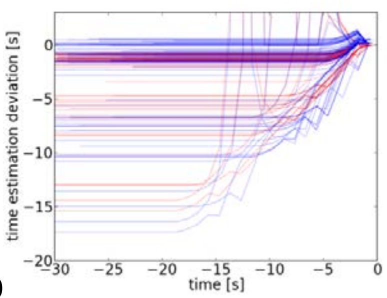
Fig. 9. Deviations of the estimated time (left) and speeds (right) from their final counterpart for arrival at (red) and leaving an intersection (blue). High prioritized vehicles

The difference in starting times is due to using a random position for the place the vehicle departs at. This was done for adding randomness into the possible co-occurrences of vehicles at high and at low prioritized roads. The behavior of vehicles on prioritized roads is straightforward and can be easily explained, see above. But the behavior of vehicles that have to react to crossing traffic are more complicated. Shown in **Fig. 10** the shape of time estimation development has three peculiarities. The first are large overestimations of the arrival and the leave time by about 260 s. The second can be seen better when focusing on the majority of traces, as done in **Fig. 10** b) and **Fig. 10** d). They show that the speed is – in addition to the continuous progress towards a correct value – oscillating with an amplitude of 2 s. The reason could be the dawdling, as discussed for vehicles approaching a major intersection. But, when looking at the same run with a dawdling value set to zero, as visualized in **Fig. 10** c) and **Fig. 10** d), some oscillations are still visible. The third peculiarity is an overestimation shortly before the link is reached.





c)



d)

Fig. 10. Deviations of the estimated time from their final counterpart for arrival at (red) and leaving an intersection (blue). Low prioritized vehicles. Top: with default dawdling, bottom: with no dawdling, left: the complete figure, right: focus on the majority of approaches

At the current time, these effects cannot be explained.

4 Summary

The currently implemented model for right-of-way rules at intersections was presented. Important features for the detailed simulation of intersection dynamics such as the computation of approach speeds, the computation of safe acceptance gaps, and the prevention of jammed intersections were described. Furthermore, it was shown that the model is suitable for simulations with configurable time steps. Preliminary validation results of the approaching behavior were presented. These will be part of a larger validation effort which is planned for all of the simulation models implemented in SUMO.

References

1. Behrisch, M., Bieker, L., Erdmann, J., Krajzewicz, D.: SUMO – Simulation of Urban MObility: An Overview. In: SIMUL 2011, The Third International Conference on Advances in System Simulation (2011)
2. DLR and contributors: SUMO homepage. http://sumo.sourceforge.net/ (2013)
3. Krajzewicz, D.: Traffic Simulation with SUMO – Simulation of Urban Mobility. In: Barceló, Jaume (Ed.): Fundamentals of Traffic Simulation, Series: International Series in Operations Research & Management Science, Vol. 145, Springer, ISBN 978-1-4419-6141-9 (2010)
4. Erdmann, J., Krajzewicz, D.: SUMO's Road Intersection Model. In Proceedings of SUMO 2013 (2013)
5. SUMO source code corresponding to this document. http://sumo-sim.org/trac.wsgi/browser/tags/v0_20_0/sumo/src
6. StVo §11
7. Calfornia Driver Handbook, p. 18
8. Krauß, S., Wagner, P., Gawron, C.: Metastable states in a microscopic model of traffic flow. Phys. Rev. E, American Physical Society, 1997, 55, pp. 5597-5602, (1997)