Vehicle Trajectory Planning in the Problem of Traffic Flow Control at Signalized Intersections

A. S. Yumaganov, A. A. Agafonov

Vehicle trajectory planning

The trajectory planning algorithms optimize vehicle trajectories at signalized intersection, assuming that the information about phase timings of traffic signal are constant and known. In addition to information about signal state and timings, these algorithms use various information about the nearest vehicles and other objects of the transport infrastructure.

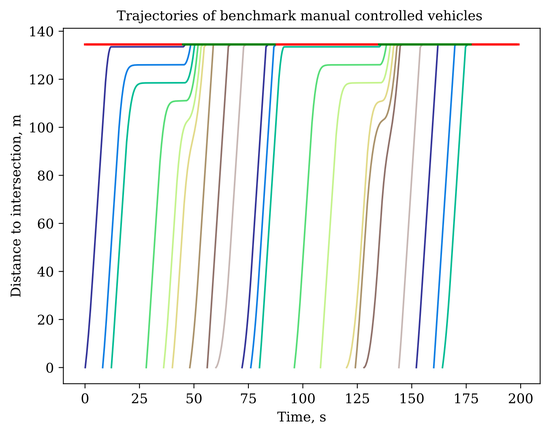
Despite the variety of trajectory planning algorithms developed, many of them can't deal with vehicle's lane-changes on multi-lane intersections. This paper presents a vehicle trajectory planning method at a signalized multi-lane intersection. The method presented in this paper is a modification of the previously known method, which is based on the Shooting Heuristic algorithm.

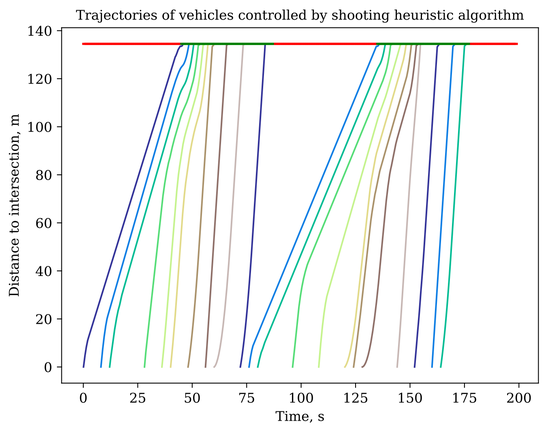
Shooting heuristic algorithm

The Shooting Heuristic algorithm contains two main processes: forward shooting process (FSP) and backward shooting process (BSP).

During the FSP, for the considered vehicle with a known initial state, a trajectory is formed, during the movement along which the vehicle accelerates to a speed v with a given acceleration $\bar{a}^f$. If the received trajectory does not intersect the trajectory of the vehicle in front and is at a safe distance from it, then FSP will return the received trajectory. If the resulting trajectory is not safe, then FSP will return a trajectory that is the result of its smooth merging with the so-called "shadow" trajectory of the vehicle in front. In the process of merging two trajectories, negative acceleration $\underline{a}^f$ is used.

If the trajectory obtained during the FSP crosses the intersection at a red traffic light, then the BSP is started for this trajectory. As a result of the BSP, the section of the trajectory containing the exit to the intersection is shifted to the beginning of the next green phase, and a trajectory is formed by smoothly merging the shifted section and the original trajectory. In the process of constructing such a trajectory, acceleration $\bar{a}^b$ and negative acceleration (deceleration) $\underline{a}^b$ are used. The resulting trajectory is the result of the Shooting Heuristic algorithm for the considered vehicle. The values of the parameters $(\vec{a}^f, \underline{a}^f, \vec{a}^b, \underline{a}^b, v)$ are limited by the capabilities of the vehicle and traffic rules.





Proposed method

Within the framework of the developed modification, for each road lane, a set of vehicle trajectories located on the corresponding lane is formed using the algorithm presented below.

```
Algorithm 1:  Lane-change sensitive modification of trajectory planning algorithm
input: Previous state, scenario information (intersections, edges, etc.)
output: New or/and updated trajectories of vehicles

for each intersection:
 for each edge:
  for each lane:

   disap_vehs = get_disap_vehs(lane, edge, prev_state) //get vehicles that disappeared from the given lane as a result of lane changing
   tg_car_disap_pos = 0

   if len(disap_vehs) > 0:
    tg_car_disap = get_prev_ahead_veh(disap_vehs, prev_state)
    //get the nearest vehicle id to the closest disappeared vehicle to the intersection in the previous simulation step
    tg_car_disap_pos = get_cur_pos(tg_car_disap) //get the position of the vehicle tg_car_disap in the current simulation step
   end if

   app_vehs = get_app_vehs(lane, prev_state)
   // get vehicles appeared on the lane
   tg_car_app_pos = 0

   if len(app_vehs) > 0:
    tg_car_app = get_clos_veh(app_vehs, lane)
    // get the closest appeared vehicle id to intersection
    tg_car_app_pos = get_cur_pos(tg_car_app)
    end if

   thresh = max(tg_car_app_pos, tg_car_disap_pos)
   vehs_on_lane = get_lane_vehs(lane)
   target_vehs = get_target_vehs(vehs_on_lane, thresh) // get sorted list of vehicles which position on lane not exceeding the obtained threshold value
   shadow_traj = get_prev_shadow_traj(thresh, prev_state, vehs_on_lane) // shadow trajectory of the closest vehicle (if exists), which current position
   exceed the obtained threshold value and which was located at the current lane on the previous simulation step
   for veh in target_vehs:
    traj = sh_heur(veh, shadow_traj) //shooting heuristic algorithm for vehicle veh taking into account the previously obtained shadow trajectory
    shadow_traj = get_shadow_traj(traj)// get shadow trajectory for traj
    result[intersection][edge][lane] = traj
   end for

  end for
 end for
end for
return result
```

Experimental results

The experimental study of the developed method were carried out in an open source traffic simulation system SUMO (Simulation of Urban MObility). SUMO allows to model transportation systems, including road vehicles, public transport, and pedestrians. The built-in API provides access to the simulated traffic and allow obtaining the values of the simulated objects parameters and manipulating their behavior "online".

Table presents the results of comparing the average waiting time at signalized intersections and average fuel consumption for two approaches: using the developed method for generating vehicle trajectories and without it. At the same time, the phase switching of traffic lights was carried out by the frequently used Uniform method, in which the phase change of the traffic light is carried out at predefined time intervals. Based on experimental studies, it can be concluded that the application of the developed algorithm can reduce the waiting time and average fuel consumption for vehicles at a signaled intersection with fixed phase change intervals.

Scenario	Total number of vehicles	Average waiting time, s		Average fuel consumption, ml	
		Proposed method + Uniform	***Uniform***	***Proposed method + Uniform***	***Uniform***
«1x1»	900	0.001	10.06	41.67	42.25
«1x1 v2»	1080	4.01	20.41	54.21	56.67
«cologne8»	2046	3.80	26.01	104.92	105.30

САМАРСКИЙ УНИВЕРСИТЕТ
SAMARA UNIVERSITY

SAMARA NATIONAL RESEARCH UNIVERSITY
34, Moskovskoye shosse, Samara, Russia (www.ssau.ru)