

Steering Autonomous Driving Agents Through Intersections in Virtual Urban Environments

Hongling Wang, Joseph K. Kearney, James Cremer
Dept. of Computer Science
University of Iowa
Iowa City, IA, USA

Peter Willemsen
School of Computing
University of Utah
Salt Lake City, UT, USA

Abstract

This paper explores how autonomous driving agents drive through intersections in virtual urban environments. The knowledge about where a vehicle runs on an intersection is embedded in environment database. Control of how and when a vehicle runs through an intersection is provided by vehicle behaviors. Vehicle control for traversing an intersection is divided into cruising behavior, following behavior, and intersection behavior. The component behaviors are integrated together to steer an autonomous driving agent through intersections. Before a virtual vehicle enters an intersection, the driving agent should make a decision that the vehicle either goes forward to pass the intersection or stops before the intersection. The chosen action is then applied to vehicle control. Because the ambient traffic and the status of traffic control signals on the intersection are dynamic, the decision should also be made on each time step.

1. Introduction

Intersections are an important part of a virtual urban environment. In a virtual city, we build roads and intersections. We put autonomous driving agents in the virtual city and have them drive on roads and in-

tersections. Intersections have special traffic control signals such as traffic lights and stop signs. Driving agents should be aware of the traffic from different directions before and when they traverse an intersection. They should also respond to the traffic control signals.

Our approach to the problem of steering virtual driving agents through intersections relies on a tight coupling between the virtual environment database and the behaviors that control the vehicles. The database encodes the physical and logical structure of the intersection. This includes both geometric information to guide vehicles from incoming lanes to outgoing lanes and the rules of road governing right of way. This information is used by vehicle behaviors to control passage through the intersection.

An intersection defines a surface area with a well-defined boundary along which incident roads connect to it. An agent enters an intersection from a specific lane on an incident road and exits the intersection on a specific lane on another incident road. To guide agents across an intersection, we overlay the intersection with corridors that splice together the lanes of incoming and outgoing roads. Corridors cross and merge with one another. This information is embedded in the environment database by attaching notes that specify the relationships between corridors [6]:

$$\begin{array}{ll} \text{corridor } a_1 & \text{CROSS} \quad (\text{corridor } b_1, b_2, b_3, \dots) \\ & \text{MERGE} \quad (\text{corridor } c_1, c_2, c_3, \dots) \end{array} \quad (1)$$

which means corridor a_1 crosses corridors b_1, b_2, b_3 ,

⁰2004 International Conference on Modeling, Simulation and Visualization Methods (MSV'04)
Copyright:CSREA Press

... and merges with corridors c_1, c_2, c_3, \dots . Agents traversing a corridor must be alert for potential collisions with other agents traversing the crossing and merging corridors on the same intersection. The traffic control signals that regulate traffic flow are also embedded in the environment database,

$$\text{light } l_1 \text{ CONTROL (corridor } d_1, d_2, d_3, \dots) \quad (2)$$

which means light l_1 controls the traffic on corridors $d_1, d_2, d_3 \dots$. Formal right-of-ways rules prioritize movement of agents through intersections on the basis of arrival time, incoming and outgoing lanes, and the state of signals that regulate traffic flow. Generally accepted social conventions help to avoid problems where the formal rules are ambiguous or incomplete. In this paper, we focus on a simple four-way intersection shown in figure 1 which is controlled by traffic lights for all incoming lanes. There are neither specific turning lanes on the incident roads nor specific traffic lights for turning left or right. Intersections of this type are common in many cities and present a challenge for autonomous vehicle control.

Steering behaviors for traversing intersections are built on top of basic competences to track corridors, maintain a desired speed, maintain safe following distance between a vehicle and its leader vehicle, and follow the right-of-way rules and generally accepted social conventions. The basic vehicle behaviors for traversing intersections include following behavior, cruising behavior, and intersection behavior. Following behavior controls a vehicle to keep a safe distance behind its leader. Cruising behavior controls a vehicle to run at its desired speed. Intersection behavior controls a vehicle to follow the right-of-way rules and generally accepted social conventions when a vehicle traverses an intersection. In this paper, we use a component-wise method to control the steering behaviors of an autonomous driving agent traversing through an intersection. First, we talk about the related work. Then, we describe the pursuit point control method used to steer virtual vehicles. We also describe the ribbon coordinate system defined on navigable surfaces modeled as ribbons. Next, control of component behaviors is introduced. After the com-

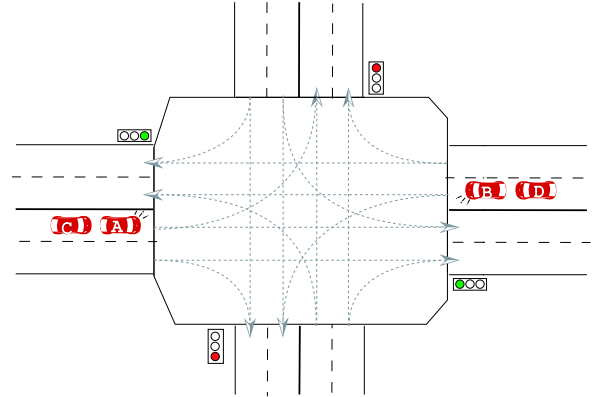


Figure 1: A four-way intersection.

ponent behaviors are presented, we integrate them together. Finally, we give a conclusion.

2. Related work

Our work builds on two related bodies of research: behavior control of autonomous driving agents and transportation research, while our contribution is mainly on the former one. Cremer et al [2] presented a model for autonomous driving behavior. They used HCSM (Hierarchical Concurrent State Machine) to define behaviors. Driving behavior is divided into several sub-behaviors. The general driving behavior is modeled as an HCSM while sub-behaviors are modeled as child HCSMs. The general driving behavior HCSM combines the output of the child HCSMs. A *most conservative rule* was introduced to resolute competing behaviors. The emphasis of this work is on HCSM framework. In [3], the HCSM model is used to generate interesting scenarios from ambient traffic composed of microscopically simulated autonomous vehicles. Sukthankar[4] introduced methods for controlling the tactical behavior of vehicles in highway traffic. The emphasis of this work is on tactical situation awareness and action selection.

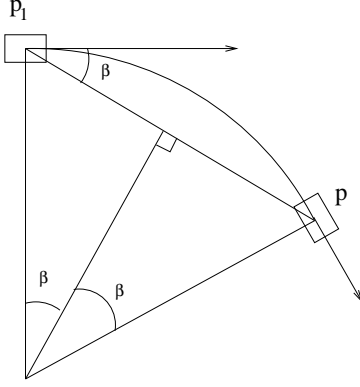


Figure 2: Pursuit point tracking.

3. Driving a vehicle through an intersection

In our driving simulator Hank, vehicles are controlled through two parameters: acceleration and steering angle. Acceleration is used to control the speed of a vehicle. Steering angle is used to control the direction of movement. However, steering angle is not set directly. Instead, it is indirectly controlled by selecting a pursuit point on the road in front of the vehicle. We use pursuit point tracking to control steering. On each time step, a vehicle is assumed to travel on a planar circular track uniquely determined by the current position and orientation of the vehicle, and the selected pursuit point[7]. The circular track is shown in figure 2, where p_1 is the current position and p is the pursuit point. The curvature of the circular track is computed and used directly to control the direction of motion of the vehicle.

Both a lane on a road and a corridor on an intersection are seen as a ribbon [5]. A corridor on an intersection, an incoming lane on an incident road on one end of the corridor, and an outgoing lane on another incident road on another end of the corridor form a continuous ribbon. A ribbon establishes a curvilinear coordinate system in which a 3-dimensional point is expressed in coordinates of distance along the central axis of the ribbon, D , offset on the ribbon surface from the central axis, O , and loft above or below the ribbon, L . The coordinates (D,O,L)

form a 3-dimensional right-handed ribbon coordinate system. With the ribbon coordinate system, an incoming lane, a corridor and an outgoing lane together form a continuous frame of reference for navigation. Steering behaviors control motion trajectories by aiming for a succession of pursuit points located on the reference ribbon some distance ahead of a vehicle's current position. By properly managing the pursuit point's distance and offset coordinates, a vehicle is controlled to run very near the centerline of the reference ribbon. In this way, a vehicle can be controlled to enter an intersection from the incoming lane and leave the intersection for the outgoing lane.

4. Cruising and following behavior

The basic goal of cruising behavior is to control a vehicle to run at its desired speed. A simple proportional controller for cruising looks like,

$$a_c = \begin{cases} \min(a_P, k_p^c * (v_d - v)) & \text{if } v < v_d \\ 0 & \text{if } v = v_d \\ \max(a_N, k_p^c * (v_d - v)) & \text{if } v > v_d \end{cases} \quad (3)$$

where a_c is the acceleration for cruising, v is the current speed, v_d is the desired speed, a_P is the maximum positive acceleration, a_N is the maximum negative acceleration, k_p^c is a proportional parameter.

A vehicle will run at its desired speed under the control of cruising behavior. However, if the vehicle has a slower leader, it may collide with its leader if it continues to run at its desired speed. Following behavior is responsible for controlling a vehicle to keep a desired distance behind its leader. Usually, a vehicle looks ahead a range of distance to search for its leader. If there is no vehicle in its way within this range of distance, the vehicle is treated as having no leader. If some vehicles are in the way of the vehicle within this range of distance, the lead vehicle which is closest to the vehicle is defined to be the leader of the vehicle. If a vehicle has a leader, a PD controller [1] is used to compute an acceleration for the vehicle,

$$a_f = \max(a_N, k_p^f * \Delta s - k_v^f * \Delta v) \quad (4)$$

where a_f is the computed acceleration, Δs is equal to the actual following distance minus the desired following distance, Δv is equal to the follower's speed minus its leader's speed, a_N is the maximum negative acceleration, k_p^f is the proportional parameter, k_v^f is the derivative parameter and k_v^f is equal to $2.0 * \sqrt{k_p^f}$ for critical damping [1].

In general, the range of distance used to look for leaders should be related to the speed of the vehicle. It is reasonable to define the range of distance as the speed of the vehicle times a constant,

$$s_l = \max(s_0, v * k_l) \quad (5)$$

where s_l is the range of distance, v is the current speed, k_l is a constant, and s_0 is the least range of distance.

It is reasonable to define the desired following distance as the speed of the vehicle times a constant,

$$s_d = \max(s_1, v * k_d) \quad (6)$$

where s_d is the desired following distance, v is the current speed, k_d is a constant, and s_1 is the minimum desired following distance.

The value of a_f computed in formula 4 might be positive, 0, or negative. Following behavior aims to slow a vehicle down (when a_f is negative) or keep the vehicle from speeding up (when a_f is 0). Positive acceleration doesn't make sense for following behavior. If a_f is positive, the vehicle is regarded as still very far away from its closest front vehicle and therefore treated as having no leader.

Leader search in following behavior is based on the ribbon composed of the incoming lane that a vehicle traverses before it enters an intersection, the corridor that the vehicle traverses on the intersection, and the outgoing lane that the vehicle traverses after the vehicle leaves the intersection. Therefore, the vehicles that are traversing this ribbon will not collide with each other under control of following behavior. As a result, two and more vehicles can traverse the same corridor at the same time. However, vehicles may come from different directions. We should also control vehicles not to collide with vehicles coming

from other directions. This is the role of intersection behavior.

5. Intersection Behavior

Intersection behavior regulates access to intersections in conformance with conventional rules of the road. It acts as a gate to the intersection, preventing vehicles from entering the intersection until it is their turn to cross. This includes adherence to traffic lights and yielding to other vehicles in or approaching the intersection that have right of way. If the vehicle is denied access to the intersection, the behavior will slow the vehicle to a stop and keep it stopped until it is safe to cross. Once the vehicle enters the intersection, it must rely on other behaviors to navigate through the intersection. Of course, human drivers will often stage turns by staking out a position in the intersection. Our more conservative behaviors will wait at the edge of the intersection.

5.1. Decision of action selection

First of all, a driving agent decides if it will respond to the traffic control signals on an intersection. The driving agent checks if the vehicle is so far away from the intersection that it is not necessary to respond to the traffic control signals, the traffic on the intersection, and the traffic coming toward the intersection from other directions. This is the screening test performed before we consider decision making strategies under different situations. If the vehicle is still very far away from the intersection, the decision is to go forward. Otherwise, the agent chooses an action in accordance with the right-of-way rules on the intersection.

If a driving agent is deprived of the right of way on an intersection by traffic control signals, for example, the traffic light is changed from green to yellow, its action selection decision depends on distance between the vehicle and the intersection. If the vehicle is so close to the intersection that it is not able to stop before entering the intersection, the vehicle just ig-

nores the traffic control signals and the decision is to go. Otherwise, the decision is to stop.

In some situations the driving agent can go forward according to the traffic control signals, it must still yield right of way to the crossing and merging traffic. In this case, it makes a decision based on the current state of the vehicle, the traffic on the intersection, and the traffic coming to the intersection from other directions. We estimate the time t_1 when the vehicle will enter the intersection and the time t_2 when the vehicle will leave the intersection based on its current position and speed. If the time between t_1 and t_2 represents a gap during which no vehicle having the right of way will traverse the crossing or merging corridors, the vehicle will go forward. Otherwise, it will stop and yield the right of way to other vehicles. For each approaching vehicle that has right of way over the yielding vehicle, we estimate the time t_1^i when it will enter the intersection and the time t_2^i when it will leave the intersection. If the time window $[t_1, t_2]$ overlaps with any time window $[t_1^i, t_2^i]$, the yielding vehicle has no gap and its decision is to stop. Otherwise, the yielding vehicle has found a gap and its decision is to go forward.

On an intersection, the traffic and the state of traffic control signals are dynamic. Therefore the decision of action selection should also be made dynamically. On each time step, the decision is reevaluated. A driving agent may alternate between choosing stopping actions and going forward actions.

In the former part of this section, we mention we need a screening test to determine if it is necessary to respond to the traffic control signals. Later, we mention a driving agent sometimes has to determine if the vehicle is able to stop within a fixed distance. However, we did not give solution for these problems. They are the focus of the next section. When we talk about stopping a vehicle, we designate where to stop it. We define the position of the geometric center of a vehicle as the vehicle's position. The desired stopping position for a stopping action is located half of the vehicle length away from the boundary of the intersection so that the whole body of the stopped vehicle is kept outside of the intersection.

5.2. Stopping an autonomous driving agent at a desired position

A non-positive acceleration should be applied to vehicle control in order to stop a running vehicle on a desired position. A negative acceleration can stop a running vehicle. A zero acceleration can keep a stopped vehicle stopped. We use the following stopping controller to compute an acceleration,

$$a_i = -\frac{v_c^2}{2 * s} \quad (7)$$

where a_i is negative acceleration, v_c is the current speed, and s is the distance between the current position and the desired stopping position. This controller gives an invariant negative acceleration. If a_i is continuously applied to vehicle control, a vehicle will just stop at the desired stopping position when its speed declines linearly to 0. The value of a_i tells whether a vehicle is too close to an intersection so that a successful stopping action is impossible and whether a vehicle is still too far away from an intersection so that a stopping action is not necessary. If the vehicle is not too close to and is not too far away from the intersection, the acceleration value a_i can successfully stop a running vehicle at the desired stopping position. We define a desired negative acceleration a_d for each vehicle. Its value reflects the negative acceleration that a driver prefers for stopping purpose. Then, the acceleration computation has the following cases,

- If a_i is within a small desired range around the desired negative acceleration, i.e., $a_i \in [a_d - \Delta a, a_d + \Delta a]$, the vehicle can be successfully stopped at the desired stopping position. Otherwise,
- If a_i is out of the desired range in positive direction, i.e., $a_i > a_d + \Delta a$, then, we believe the desired stopping position is still far away. It is too early to respond to the traffic control signals. Otherwise,
- If a_i is out of the desired range in negative direction, i.e., $a_i < a_d - \Delta a$, then, the vehicle

can not stop soon enough at its desired deceleration rate. However, if $a_i \geq a_N$, where a_N is the maximum negative acceleration, the vehicle can still be stopped at the desired stopping position. Otherwise,

- If a_i is out of the desired range in negative direction, i.e., $a_i < a_d - \Delta a$, and $a_i < a_N$, the vehicle is too close to the desired stopping position so that it is impossible to stop the vehicle at the desired stopping position.

In conclusion, the role of the intersection behavior is to gate entry into intersections as dictated by the state of traffic control devices and the right-of-way rules. The next section examines the integration of the intersection behavior with other component behaviors.

6. Integrate component behaviors

Cruising behavior computes an acceleration by formula 3. The cruising behavior is always active. Its acceleration contribution can be either positive, zero, or negative. Following behavior computes an acceleration by formula 4 if the vehicle has a leader. The following behavior is active if the vehicle has a leader and the computed acceleration is non-positive. It is not active if the vehicle has no leader or the computed acceleration is positive. Its acceleration contribution is always non-positive if it has one. As for intersection behavior, if the driving agent chooses to go forward, intersection behavior restrains from acceleration contribution and therefore it is not active. If an action of stopping is chosen, intersection behavior computes a negative acceleration by formula 7 or gives a zero acceleration if the vehicle has been stopped at the desired stopping position and the intersection behavior is active. The acceleration contribution from intersection behavior is always non-positive if it has one. The acceleration contributions from different component behaviors are competitive. Cremer et al. [3] introduced a *most conservative rule* that chooses the minimum acceleration produced by competing behaviors. According to this

rule, the three component behaviors can be integrated together in the following manner,

$$a = \min(a_c, a_f, a_i) \quad (8)$$

where a_c is the acceleration contribution of cruising behavior, a_f is the acceleration contribution of following behavior, a_i is the acceleration contribution of intersection behavior, and a is the acceleration after the integration of component behaviors. If following behavior is not active, we take a_f away from formula 8. If intersection behavior is not active, we take a_i away from formula 8.

Experiments show that behaviors work well after they are integrated together in the above manner except that some vehicles may fall into starvation or deadlock situations. A virtual vehicle may fall into starvation if it continuously yields right of way to other vehicles. A virtual vehicle may fall into deadlock when it yields right of way to other vehicles while it is in the way of other vehicles. In this kind of situations, the virtual vehicle could not make progress while keeping other vehicles from making progress. We may meet these two situations in real world. A driver in real world may enter the intersection a little bit and get a kind of preoccupied advantage. In virtual world, we use the same method to solve the starvation and deadlock problems.

7. Conclusion

This paper presents a simple framework to control the steering behaviors of autonomous driving agents for driving through intersections in virtual urban environments. Emphasis is placed on intersection behavior and its integration with other component behaviors. The role of steering autonomous driving agent through an intersection is divided onto two parts: environment database which embeds the structure of intersections and behaviors which steer autonomous driving agents according to traffic rules.

Experiments show that our virtual vehicles run naturally on intersections. They responds to traffic control signals and ambient traffic according to

right-of-way rules. Vehicles coming from different directions can traverse an intersection at the same time if the corridors they are traversing do not merge with or cross each other. Vehicles coming from the same direction can traverse the same corridor at the same time while they keep a same distance from each other. Even when the traffic is very dense, the vehicles still run well on intersections.

8. Acknowledgements

This material is based on work supported through National Science Foundation grants CDA-9623614, INT-9724746, EIA-0130864, and IS-0002535.

References

- [1] J. J. Craig. *Introduction to Robotics: Mechanics and Control*. Addison-Wesley, 1989.
- [2] J. Cremer, J. Kearney, and Y. Papelis. Hcsm: A framework for behavior and scenario control in virtual environments. *ACM Transactions on Modeling and Computer Animation*, (3), 7 1995.
- [3] J. Cremer, J. Kearney, and P. Willemsen. Directable behavior models for virtual driving scenarios. *Transactions of the society for computer simulation international*, (2), 6 1997.
- [4] R. Sukthankar. *Situation Awareness for Tactical Driving*. PhD thesis, Robotics Institute, Carnegie Mellon University, 1997.
- [5] P. Willemsen, J. Kearney, and H. Wang. Ribbon networks for modeling navigable paths of autonomous agents in virtual urban environments. In *Proceedings of IEEE Virtual Reality Conference*, pages 79–86, March 2003.
- [6] P. J. Willemsen. *Behavior and scenario modeling for real-time virtual environment*. PhD thesis, The University of Iowa, 2000.
- [7] J. Wit, C. Crane, D. Armstrong, and J. Duffy. Autonomous ground vehicle path tracking. In *Proceedings of the 32nd Southeastern Symposium on System Theory*, March 2000.