

## 도시 교통 시뮬레이션

김경아<sup>0</sup>, 김덕수, 윤성의  
한국과학기술원  
(ruddk527, bluekds, sungeui)@kaist.ac.kr

### Urban Mobility Simulation

Kyoungah Kim<sup>0</sup>, Duksu Kim, Sung-eui Yoon  
Korea Advanced Institute of Science and Technology (KAIST)

#### 요약

본 논문에서는 시내 상황을 반영한 교통 시뮬레이션을 위한 띠(ribbon)형 도로 네트워크를 형성하고, 에너지 함수를 이용하여 대용량의 차량 객체들의 움직임을 사실적으로 재현하는 실시간 시뮬레이션 방법을 제안한다. 본 연구에서 제안하는 방법으로 GIS (Geographic Information System)의 지도정보 혹은 임의 생성된 지도 모델을 이용하여 도로 네트워크 모델을 기존의 연구보다 적은 오차로 효율적으로 구성할 수 있다. 차량 객체들의 움직임을 사실적으로 재현하기 위하여, 본 논문은 새로운 객체-환경 상호작용 에너지 함수와 객체-객체 상호작용 에너지 함수를 제안한다. 본 논문에서 제안하는 상호작용 에너지와 효율적인 도로 네트워크 구성법으로 교차로 및 진입로 등과 같은 대부분의 교통 상황을 사실적으로 실시간 시뮬레이션 할 수 있다.

#### Abstract

We propose an intelligent ribbon road network for automatic vehicle simulation, and a real-time algorithm for large-scale, realistic traffic simulation based on artificial energy functions. Our method reconstructs a road network automatically from both GIS (Geographic Information System) real-world data and synthetic models. Such automatic road network helps us to easily simulate almost every possible scenario such as intersections, ramps, etc. In order to simulate agents' movement, we design car-environment interaction energy and car-car interaction energy functions. Car agents move along the road network according to the proposed energy functions while avoiding collisions with other car agents.

#### 1. 서론

교통량이 증가하고 도로 네트워크가 점점 복잡해짐에 따라 정확한 교통상황 예측이나 관리에 대한 요구가 점점 증가하고 있다. 효율적인 교통 시뮬레이션 시스템 개발을 통해 이러한 요구를 충족시키고, 더 나아가 교통 관련 정책에 직간접적 도움을 주려는 시도들이 진행되고 있다 [1]. 또한, 교통 시뮬레이션 기술은 구글(Google), Bing(Bing) 등과 같은 대형 인터넷 정보 서비스 업체들에서 제공하는 지도 API를 통해 쉽게 접근할 수 있는 GIS(Geographic Information System) 정보 또는 가상

도시 교통 모델을 이용하여 현실적인 영상을 생성하는 데에도 이용되고 있다 [2].

그러나 대부분의 연구들은 시내 교통 상황이 고려되지 않은 고속도로 시나리오에 국한되어 있다. 고속도로 시뮬레이션은 차량 객체들의 진로를 계산하기 위해 고려해야 하는 외부요소가 적고, 전체 시스템을 수학적으로 근사하기에도 용이하기 때문에 다양한 시뮬레이션 방법이 활발히 연구되었다. 이와 달리 시내 교통 시뮬레이션은 복잡한 도로 네트워크를 정의해주어야 하고, 도로 네트워크와 시뮬레이션 모듈이 상호작용 해야 한다

는 어려움이 있다. 본 연구에서는 시내 상황을 반영할 수 있는 도로 네트워크 구성 방법과 사실적인 교통 상황을 효율적으로 재현 할 수 있는 새로운 교통 시뮬레이션 기술을 제안한다.

본 논문에서 제안한 도로 네트워크 구성 기술은 이산 추출된 점 집합만으로  $C^1$  연속 도로를 기존 방법에 비해 보다 적은 오차로 재구성할 수 있다. 즉, 사용자가 원하는 도로 모델을 쉽게 만들어 낼 수 있다. 또한 객체와 환경간 상호작용과 객체와 객체간 상호작용을 대규모 객체 네트워크에서 효율적으로 계산하여 사실적인 교통 시뮬레이션이 가능하다.

## 2. 관련 연구

시내 상황을 고려한 교통 시뮬레이션은 크게 도로 네트워크를 정의해주는 과정과 차들을 움직여 주는 시뮬레이션 과정으로 나뉜다.

### 2.1. 도로 네트워크

Peter Willemsen 등 [3]은 도로를 연속한 띠들로 이루어진 띠 네트워크(ribbon network)로 표현하였고, 최근 D. Wilkie 등 [4]은 이를 바탕으로 기존의 시뮬레이션 모델을 쉽게 적용할 수 있는 시내 도로 네트워크 구성 방법을 제안하였다. D. Wilkie[4]는 GIS에서 제공되는 도로의 폴리라인 (polyline) 으로부터 그에 접하는  $C^1$  연속하는 곡선도로 (arc road) 로 이루어진 띠 네트워크를 생성해 냈다. 그러나 이 곡선도로는 이산 추출한 점 집합을 포함하지 않기 때문에 실제 도로의 모양과는 차이를 보인다. 즉, 사용자가 원하는 대로 도로 네트워크를 구성하기 어렵다는 단점이 있다. 본 연구에서는 곡선도로가 주어진 점들을 지나도록 변형하여 기존 방법보다 정확도가 높은 도로 네트워크를 생성하는 방법을 제안한다.

### 2.2. 교통 시뮬레이션

교통 시뮬레이션 기술은 크게 미시적(microscopic) 시뮬레이션과 거시적(macrosopic) 시뮬레이션으로 나눌 수 있다. 미시적 시뮬레이션은 각각의 차량 객체의 경로를 계산하는 방법으로, 현실적인 영상을 만들어낼 수 있다. 하지만, 차량 객체가 많아지거나 도로 모델이 커질 경우 실시간 시뮬레이션이 불가능하다. 반면, 거시적 시뮬레이션은 교통 전체의 흐름을 빠르게 계산해내는 것에 유리하다. 그러나 이 경우 실제 객체들은 각각이 다른 성향을 가지고 움직이는데 반해 물리적으로 이상적인 흐름

를 따라 차들이 움직인다고 가정했기 때문에 객체 각각의 현실적인 움직임을 재현하기 어렵다 [5].

대용량 시나리오 상의 사실적 시뮬레이션에 대한 계산 복잡도를 낮추기 위해서는 사실적 재현이 필요한 부분은 사실적으로 재현하고 상대적으로 필요하지 않은 부분은 간단히 재현해야 한다. 여기서 사실적 재현이 필요한 부분을 판단하기 위해서 사용자가 직접 지정할 수도 있고 객체간 상호작용이 활발히 일어나는 영역을 계산할 수도 있다. J. Sewall 등 [5]은 사용자가 지정한 영역을 미시적으로 시뮬레이션하고 나머지 부분은 거시적으로 시뮬레이션 하는 방법을 제안하였다. 그러나 이 방법은 사용자의 지정 영역만 자세히 보여주는 것이기 때문에 근본적인 해결 방안이 될 수 없다. 양질의 대규모 시뮬레이션을 위해서는, 사용자가 아닌 시스템이 객체간 상호작용을 측정하여 근사 가능한 영역을 판단할 수 있어야 한다.

본 연구에서는 대규모 네트워크 내부의 객체 간 상호작용 함수와 각 객체의 경로를 상호작용 함수로 정의하는 방법을 제안한다.

Michael T. Wolf 등 [6]은 각 차량 객체의 순간속력을 구하기 위하여 각 객체의 전위함수 (potential energy) 를 정의하였다. 이 에너지 함수는 주변의 객체 혹은 도로 분리선 과의 거리를 고려한 함수로, 에너지 함수를 최소화하는 방향, 즉 충돌 가능성이 적은 방향으로 객체가 움직이도록 설계되었다. 이 에너지 함수는 연속한 함수의 형태로 객체와 환경 간의 상호작용을 정의함으로써 자연스러운 객체의 움직임을 구현할 수 있지만, 대량의 객체가 상호작용할 수 있는 시스템으로 발전하지 못했다. 또한 제한적으로 정의되어서 다양한 교통 시나리오에 적용하는 데에 어려움이 있다. 본 연구에서는 이 전위함수를 기반으로, 차량 네트워크에서의 객체 상호작용 함수를 정의한다.

## 3. 도로 네트워크

본 장에서는 GIS 정보를 이용하여 정확한 도로 네트워크를 구성하는 방법을 제안한다.

본 연구에서 제안하는 도로 네트워크는 각 도로 구간 간의 연결 관계를 포함한 유향 그래프 형태로 정의된다. 차량 객체들이 도로 네트워크 위를 움직이게 하기 위해서, 각 도로 구간 간의 연결 관계를 그래프의 에지(edge)로 표현한다. 네트워크의 구성의 기본단위는 도로와 연결자로, 각각 그래프의 노드(node)와 에지로 대응된다. 도로는 차선의 개수와 위치 정보, 시작점과 끝점의 연결 정보를 고정된 값으로 가진다. 즉, 하나의 도로

단위 안에서 차선의 개수가 변할 수 없으며, 시작점과 끝점 외의 위치에서 다른 도로 단위로 연결 될 수 없다. 차선의 개수는 입력 값으로 주어지며, 도로 위치 정보는 점 집합이 입력 값으로 주어졌을 때 각 점을 지나는 C<sup>1</sup> 연속하는 호와 선분의 집합으로 정의한다(3.1절). 또한, 도로의 위치정보는 매개변수화 하여, 차량 객체 위치 보 간을 효율적으로 한다(3.3절). 각 도로 단위의 시작점과 끝점에는 한 개 이상의 도로가 연결될 수 있다. 연결자 로 도로간의 연결을 정의하며, 각 도로의 끝을 실제로 이어주는 가상의 도로(internal lane)와 활성화 상태 (activation state)를 통해 각종 교통상황 시나리오를 정의 한다(3.2절).

**기호 및 용어:** 도로의 집합은 기호 R로 정의되며, R에 속한 i번째 도로를 지칭하기 위해 R<sub>i</sub>를 사용한다. R<sub>i</sub>와 R<sub>j</sub>를 연결하는 연결자는 기호 R<sub>ij</sub>로 정의 한다.

### 3.1. 도로의 기하적 모델링

GIS에서 제공하는 도로 정보는 일반적으로 위성사진에서 추출한 폴리라인의 형태이다. 본 연구에서의 입력 정보의 형태 또한 폴리라인 혹은 점 집합{p<sub>i</sub>(i = 0, 1, ..., n)}를 대상으로 한다. 모델링에서의 오류를 없애기 위하여 임의의 연속한 세 점이 일직선 위에 있지 않는다고 가정한다. 세 개 이상의 점이 한 직선 위에 있을 경우, 모든 내부 점들은 양 끝점으로 이루어진 선분에 포함되기 때문에, 일직선 위인 연속한 세 점의 경우의 수를 없애도 모델링의 정확도는 떨어지지 않는다. 이 과정은 [4]에 소개되어 있다.

최종적으로 생성되는 곡선도로가 점 집합의 각 점을 지나도록 근사하기 위해서 각 점에서의 접선 벡터를 전방/후방차분법보다 적은 오차를 보이는 중앙차분법(central difference)으로 근사 한다. e<sub>i</sub>, t<sub>i</sub>, o<sub>i</sub>, n<sub>i</sub>를 각각 p<sub>i</sub>와 p<sub>i+1</sub>를 잇는 선분의 표준 벡터(e<sub>i</sub>), p<sub>i</sub>에서의 접선 벡터(t<sub>i</sub>), 방향성 벡터(o<sub>i</sub>), 법선 벡터(n<sub>i</sub>)로 정의한다. 방향성 벡터는 폴리라인의 진행 방향이 시계 방향인지 시계 반대 방향인지를 정의한다.

$$e_i = \frac{p_{i+1} - p_i}{|p_{i+1} - p_i|} \quad , i = 0, n-1$$

$$t_i = \begin{cases} e_i & , i = 0, n-1 \\ \frac{e_{i+1} - e_{i-1}}{|e_{i+1} - e_{i-1}|} & , otherwise \end{cases}$$

$$o_i = t_{i-1} \times t_i \quad n_i = o_i \times t_i$$

위에서 정의한 정보로 새롭게 정의된 도로 구획 띠(ribbon)는 호 또는 선분의 형태를 갖게 되며 아래와 같

이 중심각(θ<sub>i</sub>), 방향 벡터(d<sub>i</sub>), 반지름(길이, r<sub>i</sub>), 중심점(시작점, s<sub>i</sub>) 그리고 방향성 벡터(o<sub>i</sub>)로 정의된다.

$$L_i = [θ_i, d_i, r_i, s_i, o_i].$$

주어진 임의의 점 집합 {p<sub>i</sub>}에 대하여, C<sup>1</sup>연속하는 세그먼트(segment)의 집합 {L<sub>i</sub>}은 두 폴리라인의 진행방향에 따라 크게 두 가지 경우로 정의된다.

두 폴리라인의 방향이 다른 경우, 도로의 진행방향을 전환하기 위해 두 개의 호가 생성된다(그림 1). 두 호가 연속하게 하기 위해서는 두 중심점 사이의 거리가 반지름의 합과 같아야 하며, 다음과 같이 정의된다.

$$r_1 = \frac{|p|^2}{4p * n_i}$$

$$r_2 = \frac{|p|^2}{4r_1(n_i * n_{i+1} + 1) - 4p * n_{i+1}}$$

두 폴리라인의 방향이 서로 같은 경우, 호와 선분, 혹은 선분과 호가 그림2 와 같이 생성된다. 그림2의 연속한 선분과 호는 각각 길이 a와 반지름 r을 가지며 아래의 식을 만족한다.

$$p_i + r n_i + a t_i = p_{i+1} + r n_{i+1}$$

각 점에 대하여 위의 과정을 반복하면 그림3과 같이 C<sup>1</sup>연속한 호와 선분의 집합이 생성되고, 미리 정의된 차선의 개수와 너비에 따라 그림3과 같은 도로가 최종 생성된다. 매우 불규칙한 추출임에도 불구하고, 연속한 도로가 자연스럽게 생성된다.

### 3.2. 도로 네트워크의 정의

정의된 각각의 도로들은 연결자를 통해 도로 네트워크를 구성한다. 도로 R<sub>i</sub>으로부터 R<sub>j</sub>로의 연결자 R<sub>ij</sub>는 R<sub>i</sub>의 마지막 구간과 R<sub>j</sub>의 첫 번째 구간에 접하면서 동시에 C<sup>1</sup>연속 조건을 만족해야 한다. 방향성 벡터, 접선 벡터 등을 올바르게 정의하기 위해서는 최소한 세 개의 점이 주어질어야 하기 때문에, R<sub>i</sub>의 마지막 구간과 일직선인 두 점(p<sub>0</sub>, p<sub>1</sub>), R<sub>j</sub>의 첫 번째 구간과 일직선인 두 점(p<sub>2</sub>, p<sub>3</sub>)을 선택한 뒤 p<sub>1</sub>p<sub>2</sub>구간에 그림1 혹은 그림2와 같은 과정을 적용하여 그림4와 같이 연결자를 모델링 한다.

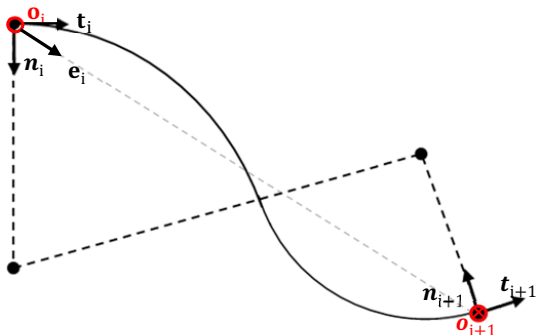


그림 1.  $o_i$  과  $o_{i+1}$  이 서로 다른 방향일 경우, 도로의 진행방향을 전환하기 위해 두 개의 호가 생성된다.

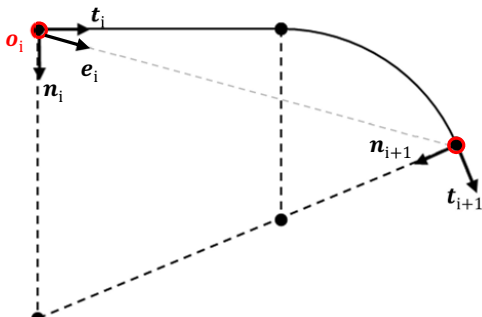


그림 2. 방향이 전환되지 않는 경우 호와 선분, 혹은 선분과 호가 위와 같이 생성된다.

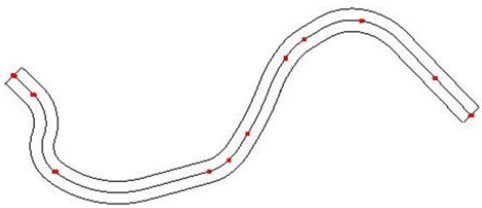


그림 3 점 집합 (붉은 점)으로부터 2차선으로 최종 생성된  $C^1$  연속한 도로.

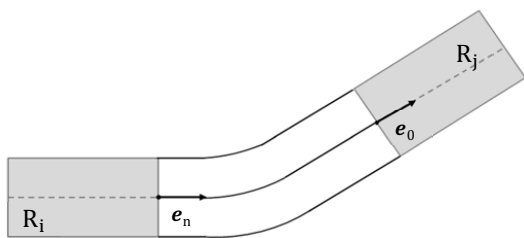


그림 4 연결자 생성의 예

**연결자 활성화 모듈:** 연결자의 활성화 상태를 조절함으로써, 다양한 교통 규칙(교차로, 진입도로 등)들을 쉽게 구현할 수 있다. 신호등이 있는 교차로 및 진입도로는 신호등의 규칙을 활성화 주기  $T_{ij}$ , 시작 시간  $s_i$ , 활성화 구간의 길이  $l_{ij}$  로 표현함으로써 구현할 수 있다. 즉  $k = 1, 2, 3, \dots$  에 대하여 시간 구간  $[kT_{ij} + s_{ij}, (k+1)T_{ij} + s_{ij}]$ 에서 연결자  $R_{ij}$ 가 활성화된다.

신호등이 없는 교차로 및 진입도로는 도로들 사이의 연관성을 정의함으로써 구현할 수 있다. 즉, 동시에 활성화가 될 수 없는 짝을 미리 정의하여 그 중 하나는 반드시 활성화되도록 한다.

시뮬레이션 과정에서 차량 객체들은 현재 소속된 도로와 연결된 도로들 중 활성화된 도로를 선택하여 움직이게 된다.

### 3.3. 도로의 매개변수화와 차량 위치 보간

객체의 직교좌표 위에서의 위치 보간을 쉽게 하기 위해서 도로의 각 구간을 매개변수화 한다. 각 구간  $L_k$ 의 구간좌표(Ribbon Coordinate)는 x축이 차량객체들의 진행 방향, 즉 방향벡터  $\mathbf{d}_k$ 로 정의되며  $s_k + r_k \theta_k$ 를 원점으로 한다. 직교좌표와의 관계는 방향벡터 방향으로 회전한 뒤 띠의 시작점으로 대칭이동한 행렬  $I_k$ 로 정의된다. 이하 섹션에서는 차량이 매개변수화 된 구간좌표에서 움직이고 있음을 가정한다. 실제 차량객체의 직교좌표 위에서의 위치는 매 프레임마다 행렬  $I_k$ 를 곱하여 보간한다.

## 4. 교통 시뮬레이션

본 장에서는 Michael T. Wolf 외 [6]이 디자인한 전위함수를 기반으로, 기존 에너지 함수를 유향그래프로 표현된 객체 네트워크에 적용시킬 수 있도록 재정의하여 일반적인 시나리오에 적용 가능한 도로 교통 시뮬레이션 방법을 제시한다.

### 4.1. 차량 객체 네트워크

차량 객체는 구간좌표와 직교좌표 각각에서의 위치  $(s_r, s)$ , 속력  $(V_r, V)$  정보와 차량 타입 및 크기를 변수로 가진다. 또한 각 차량 객체(운전자)의 성격을 반영하기 위하여 *희망* 속도  $(V_D)$ 를 가지며, 차선 변경 상태와 변경 방향을 변수로 가진다.

하나의 도로에 속한 차량 객체들은 유향 그래프 형태의 네트워크를 형성하며, 이 네트워크를 통해 객체간의 상호작용이 정의 된다. 계산의 효율성을 위해, 실제

차량 운행시 직접적인 영향을 주는 전방, 좌측 전방, 우측 전방에 있는 차량 객체만 연결하여 네트워크를 생성한다. 에지의 방향은 후방에서 전방 방향을 따르고, 따라서 각 차의 외차수(out-degree)와 내차수(in-degree)는 각각 항상 3 이하가 된다. 객체가 다른 도로로 옮겨갈 경우 해당 객체의 소속정보를 갱신한다 생성된 차량 네트워크는 매 프레임에 x축에 따라 객체를 정렬함으로써 갱신 할 수 있다.

임의의 객체  $c_i$  에 대하여 현재 차선 전/후방 차량 객체를  $F(c_i), F^{-1}(c_i)$ 라고 표현한다. 좌측 차선의 전/후방 차량 객체는  $L(c_i), R^{-1}(c_i)$ 로 표현되며 마찬가지로 우측 차선에 대해서도  $R(c_i), L^{-1}(c_i)$ 를 정의한다.

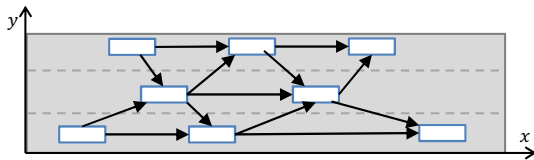


그림 5 차량 네트워크의 예

#### 4.2. 객체-도로 상호작용 에너지

Michael T. Wolf 외 (6)는 차량 객체와 도로 분리선 충돌 방지를 위한 에너지 함수를 다음과 같이 정의하였다.

$$U_{lane} = A_{lane} (t_L \exp\left(-\frac{(y - y_l)^2}{2\sigma^2}\right) + t_R \exp\left(-\frac{(y - y_r)^2}{2\sigma^2}\right))$$

$y$ 는 객체의  $y$ 값,  $y_l$ 와  $y_r$ 은 각각 좌측과 우측으로 인접한 도로 분리선의  $y$ 값이며  $\sigma$ 는 에너지가 증감하는 정도를 정의한다. 에너지 함수가 최저점에 있을 때 객체는 가장 안정적인 상태, 즉 가장  $y$ 방향 속력이 작은 상태를 유지한다. 즉, 표준 공간 위에서  $y$ 방향의 속력은  $-\partial U_{lane} / \partial y$  이다.

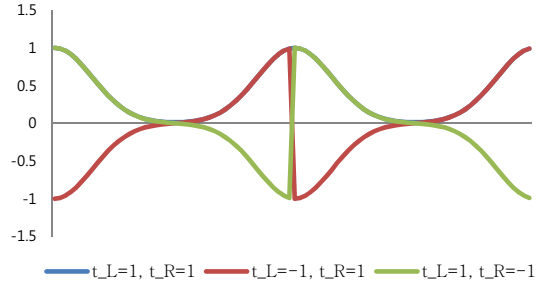
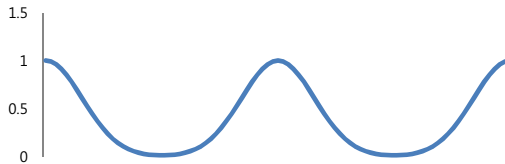


그림 5. 각각의 경우의 수에 대한 객체-도로 상호작용 에너지

$t_L$ 과  $t_R$ 은 차선변경상태를 나타내는 변수로, 좌측으로 차선을 변경하거나 우측으로 차선을 변경할 때 각각 -1의 값을 가지게 된다. 에너지 함수가 최고점이 되어 차선이 변경되는 즉시  $t_L$ 과  $t_R$ 을 1로 전환해주기 때문에 객체-도로 상호작용 에너지는 항상 연속상태를 유지한다.

#### 4.3. 객체-객체 상호작용 에너지

Wolf, Joel 와 Burdick [6]의 연구에서는 차량 간의 충돌 방지를 위한 에너지를 거리  $K$ 에 대하여 아래와 같이 정의하였다. 또한 상호 작용 대상 차량 객체가 전방 혹은 후방/측면 중 어느 곳에 있는지에 따라 영향력이 달라지기 때문에, 차량 객체  $c_i$ 로부터 차량 객체  $c_j$ 로의 링크에 대하여 거리를 아래와 같이 두 가지로 정의하였다. 거리  $K_1$ 은 후방/좌측/우측에 있는 객체와의 상호작용을 정의하기 위한 단순 거리이며, 거리  $K_2$ 는 전방에 있는 객체와의 상호작용을 정의하기 위하여 상대속도를 고려한 거리이다.  $v_i, q_i$ 는 객체  $c_i$ 의 현재 속력과 위치이다.

$$\begin{aligned} K_1(c_i, c_j) &= \|q_i - q_j\| \\ K_2(c_i, c_j) &= \delta(v_i, v_j) K_1(c_i, c_j) \\ \delta(v_i, v_j) &= \frac{d_0}{T_f v_i} e^{-\beta(v_i - v_j)} \\ U_{car}(K) &= \frac{A_{car} e^{-\alpha K}}{K} \end{aligned}$$

여기서  $A_{car}$ 는  $U_{car}$ 의 최대값,  $\alpha$ 는,  $U_{car}$ 의  $K$ 에 대한 민감도,  $\beta$ 는  $\delta(v_i, v_j)$ 의  $v_i - v_j$ 에 대한 민감도,  $T_f$ 는 차량객체간 최소도달시간(3초),  $d_0$ 는  $U_{car}$ 가 영향을 미칠 수 있는 최대거리 이다.

Wolf, Joel 와 Burdick [6]의 연구에서는 도로 위에

각 객체의 에너지 함수들을 누적 시킨 뒤 지역 최소값(local minima)으로 객체를 이동시켰으나, 본 연구에서는 높은 계산 복잡도를 낮추기 위해 객체 네트워크에서 정의된 에너지에 대한 객체-객체 상호작용 에너지 함수만을 계산한다.

임의의 객체  $c_i$ 에 대하여 현재 진행 중인 차선에서 받는 에너지 함수를  $U_F(c_i)$ , 좌측 차선에서 받는 에너지 함수를  $U_L(c_i)$ , 우측 차선에서 받는 에너지 함수를  $U_R(c_i)$ 라고 정의하며 아래와 같이 표현된다.

$$U_F(c_i) = U_{car}(K_2(c_i, F(c_i))) - U_{car}(K_1(F^{-1}(c_i), c_i))$$

$$U_L(c_i) = U_{car}(K_1(c_i, L(c_i))) - U_{car}(K_1(R^{-1}(c_i), c_i))$$

$$U_R(c_i) = U_{car}(K_1(c_i, R(c_i))) - U_{car}(K_1(L^{-1}(c_i), c_i))$$

측 전방에 있는 객체와의 충돌을 피하기 위해 x축 반대 방향으로 받는 힘과 후방에 있는 객체와의 충돌을 피하기 위한 x축 방향 힘의 합이 된다.

각 객체는  $U_F > \rho U_L$ 의 조건을 만족하면 좌측 차선으로 변경하기 위해 객체의 차선변경 변수를 갱신하며, 우측 차선 변경도 마찬가지로 정의된다. 여기서  $\rho$ 는 각 차량객체가 얼마나 민감하게 차선을 변경하는지를 정의한다. 다음 프레임에 변경된 차선변경 변수에 따라 네트워크가 갱신되어 차선 변경이 진행된다.

프레임 t+1에서의 최종 x축 방향의 속력  $V^{t+1}$ 은 아래와 같이 정의된다.

$$V_r^{t+1} = U_F + \gamma V_D + (1 - \gamma)V_r^t$$

위의 식에 따라 계산된 속력으로 각 객체를 움직여준다. 시뮬레이션 과정은 아래의 도표로 요약할 수 있다.

---

**알고리즘1** 교통 시뮬레이션 알고리즘

---

입력: n대의 차량 객체와 m개의 도로 단위

---

**WHILE:**

연결자들의 활성화 상태 결정

**FOR** 각 도로단위  $L_k$ :

차량 정렬 :  $O\left(\frac{n}{m} \log\left(\frac{n}{m}\right)\right)$

**FOR** 각 차량객체  $c_i$ :

$F, L, R, U_{lane}, U_F, U_R, U_L, V_r$  갱신

$t_L, t_R$  결정

$s_r = s_r + V_r$

**IF**  $s_r > 1$  **THEN** 활성 연결자로 이동,  $s_r = 0$

$s' = I_k s_r, V = (s' - s) / \Delta t$

---

## 5. 결과

본 논문의 시뮬레이션 방법의 계산 속도를 측정하기 위하여 단순한 시나리오에서의 프레임당 소요시간을 측정하였다. 또한 시내교통 시나리오에서 현실적인 움직임은 보여주는지 확인하기 위하여 임의의 교차로 위에서의 시뮬레이션 결과를 관찰하였다.

### 5.1. 계산 속도

2차선 도로에서 대량의 차들을 시뮬레이션 했을 때 한 프레임의 계산 속도를 측정하였다.

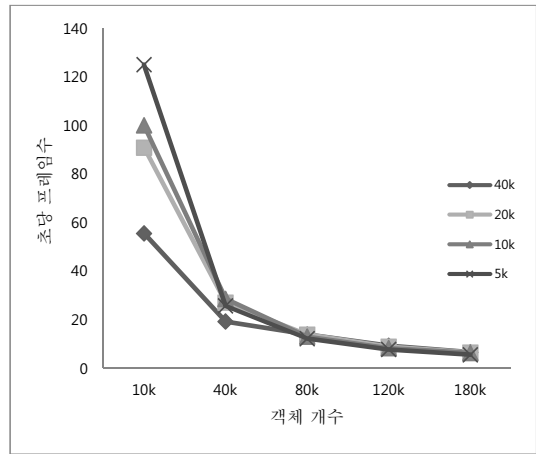


차트 1 2000km의 모델에서 전체 도로 네트워크의 구간이 5k, 10k, 20k, 40k인 경우에 객체 개수에 따른 초당 프레임 수

현재의 시스템은 크게 네트워크 업데이트 부분과 시뮬레이션 부분으로 나뉜다. 도로 구간 개수가 m이라 할 때, 네트워크 업데이트의 정렬 연산의 계산 복잡도는  $O\left(\frac{n}{m} \log\left(\frac{n}{m}\right)\right)$ 가 된다. 실제로 m에 따라 프레임당 소요시간 증가 폭이 달라짐을 볼 수 있다.

프레임 당 전체소요시간 중, 객체 네트워크 업데이트 시의 정렬 연산이 10,000대 규모의 네트워크에서 96.2%, 160,000대 규모의 네트워크에서 86.9%를 차지하였다. 정렬 연산을 배제할 경우 10,000대 규모에서 초당 200 프레임, 160,000대 규모에서 초당 9.6프레임의 속도를 보인다. 다시 말해 객체 네트워크를 선형 연산으로 업데이트할 경우 계산 복잡도를 크게 개선할 수 있다.

본 결과는 Intel® Core™2 Extreme CPU X9650 (3.00GHz)로 실험하였다. M. Treiber 등 [7]의 객체 기반 미시적 시뮬레이션 (agent-based microscopic simulation)은 Intel Core i7 980X Westmere (3.33GHz)에서 150,000대

규모에서 초당 약 12프레임, 190,000에서 약 8프레임의 속도를 보였다 [8]. 기존 방법과 비교했을 때 속도 면에서 더 우수하다고 할 수는 없지만, 다른 미시적 시물레이션과 달리 다중해상도로 확장이 가능하기 때문에 추후 더욱 개선될 가능성이 있다.

## 5.2. 시물레이션 시나리오

다음은 6차선도로가 만나는 교차로에서의 시나리오이다. 불규칙한 형태의 교차로에서도 그림6과 같이 연결자의 가상 도로를 자연스럽게 생성하여 차량 객체들이 부드럽게 움직인다. 또한 그림7과 그림8과 같이 상호작용 합수의 계산 결과에 따라 자연스럽게 차선을 변경하며 움직인다.

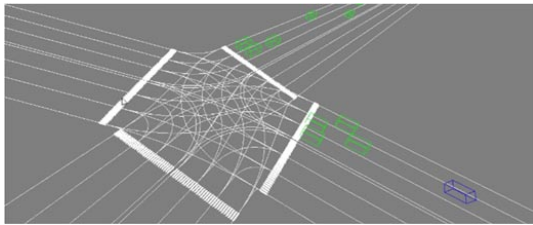


그림 6 교차로에서의 연결자 가상도로 생성 결과

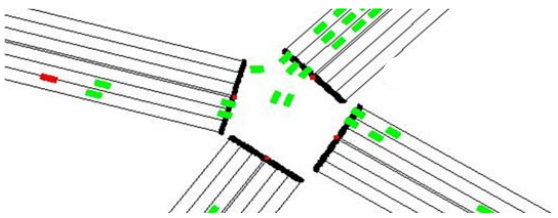


그림 7 붉은 색으로 표현된 객체가 상호작용 에너지가 더 안정적인 왼쪽으로 차선을 변경하고 있다.

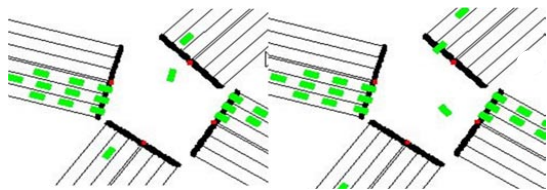


그림 8 연결자를 따라 좌회전하는 차량 객체의 예시

다음은 4차선 진입로의 시나리오이다. 도로의 폭이 변하는 부분에서 잦은 차선 변경, 혹은 속도 저하가 나타나는 것을 볼 수 있다.

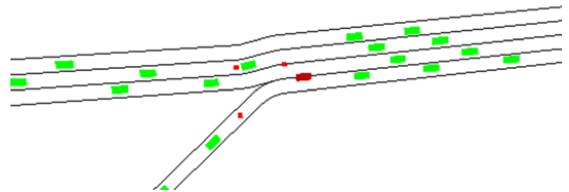


그림 9 진입 후 차선을 바꾸는 차량 (붉은 색)

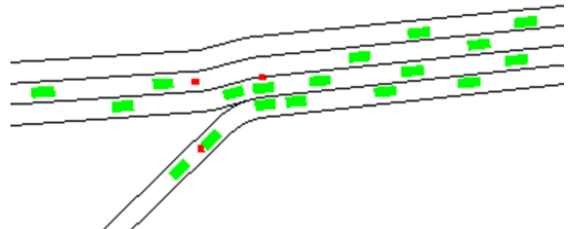


그림 10 진입로 근방에 속도가 저하되어 지체되는 모습

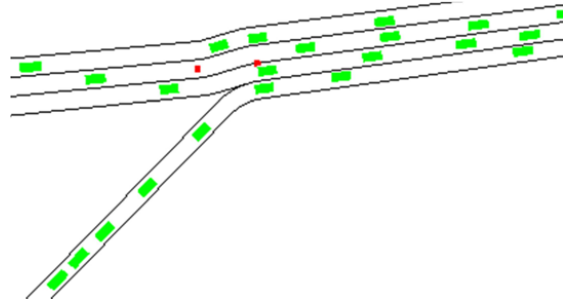


그림 11 지체가 지속되자 진입로 전체(하단)의 속도 저하 일어나는 모습

## 6. 결론 및 향후 연구

본 연구는 시내 교통상황을 간단한 규칙 조합만으로 구현할 수 있는 방법을 제안했다. 또한, GIS 정보를 이용하여 도로 모델을 구성하는 효율적인 방법을 제안했다. 본 연구가 제안한 방법은 모든 시내 교통 상황의 시나리오에 적용 가능하고, 도로 네트워크가 C<sup>1</sup>연속하기 때문에 별도의 과정 없이도 객체들이 자연스러운 경로를 따라 움직인다. 또한 본 논문에서 제시한 상호작용 에너지를 이용하여 객체간의 상호작용으로 인해 발생하는 다양한 교통상황을 사실적으로 재현할 수 있다. 결과에서 언급했듯이 차량의 네트워크를 얼마나 단순화하고 효율적인 업데이트 알고리즘을 만드느냐에 따라 계산 복잡도를 크게 낮출 수 있는 가능성이 열려 있다. 향후에는 차량 네트워크를 효율적으로 계산하고, 차량 네트워크의 다중해상도를 구현하여 계산 공간의 크기를 줄이는 연구를 진행할 계획이다.

## 감사의 글

다양한 조언으로 더욱 견실한 논문을 발표하게 도와주신 익명의 심사관분들께 감사의 말을 전하고 싶습니다. 또한, 연구진행에 많은 조언과 도움을 주신 대용량 그래픽스 연구실 동료들에게도 감사합니다. 본 연구는 한국콘텐츠진흥원의 문화콘텐츠산업기술지원사업의 지원을 받는 레고 블럭형 통합 VFX 시뮬레이션 기술 개발 사업의 일환으로 진행 되었습니다.

## 참고 문헌

1. *Traffic Scheduling Simulation and Assignment for Area-Wide Evacuation*. **Chiu, Y. C.** : Proceedings of the IEEE Intelligent Transportation Systems Conference(ITSC), 2004, Vols. 537-542.
2. maps.google.com. *Googla Maps*. [Online]
3. *Ribbon Networks for Modeling Navigable Paths of Autonomous Agents in Virtual Environment*. **Peter Willemse, Joseph K. Kerney, Hongling Wang.** : IEEE Transactions on Visualization and Computer Graphics, 2006, Vols. 12, No. 3.
4. *Transforming GIS data into Functional Road Models for Large-Scale Traffic Simulation*. **D. Wilkie, J. Sewall, M.C. Lin.** : IEEE Transactions on Visualizaiton and Computer Grahpics, 2010, Vols. 16, No. 5.
5. *Continuum Traffic Simulation*. **J. Sewall, D. Wilkie, P. Merrell, M. C. Lin.** s.l. : Eurographics, 2010, Vols. 29, No. 2.
6. *Artificial Potential Functions for Highway Driving with Collision Avoidance*. **Michael T. Wolf, Joel W. Burdick.** : IEEE International Conference on Robotics and Automation, 2008.
7. *Congested traffic states in empirical observations and microscopic simulations*. **M. Treiber, A. Hennecke, D. Helbing.** s.l. : Physical Review E 62, 2000, Vol. 2.
8. *Interactive Hybrid Simulation of Large-Scale Traffic*. **J. Sewall, D. Wilkie.** s.l. : ACM SIGGRAPH conference proceedings, 2011.

## <저자 소개>



김경아 (Kyoungah Kim)

- 2008년 ~ 현재 한국과학기술원 전산학과 학사
- 관심분야: 3D 애니메이션, 객체 인식



김덕수 (Duksu Kim)

- 2008년 성균관대학교 정보통신공학부 학사
- 2008년 ~ 현재 한국과학기술원 전산학과 석박사 통합 과정
- 관심분야: 근접질의, 병렬컴퓨팅



윤성의 (Sung-Eui Yoon)

- 1999년 서울대학교 컴퓨터공학과 학사
- 2001년 서울대학교 컴퓨터공학과 석사
- 2005년 University of North Carolina at Chapel Hill 박사
- 2006년 ~ 2007년 Lawrence Livermore National Laboratory Post-doctoral scholar
- 2007년 ~ 현재 한국과학기술원 (KAIST), IWON associate professor
- 관심분야: Scalable graphics/geometric algorithms, visualization, interactive rendering, collision detection, motion planning, image retrieval, cache-coherent algorithm and layouts, and other related geometric problem