

Дніпропетровський ліцей інформаційних технологій  
при Дніпропетровському національному університеті  
імені Олеся Гончара

Випускна робота  
на тему:  
**«Комп'ютерне моделювання у застосуванні до  
міського автомобільного руху»**

*Виконав:*  
*ліцеїст 11-А-2 курсу*  
*Федоряка Дмитро Сергійович*

*Науковий керівник:*  
*Ентін Йосиф Абрамович*

---

Дніпропетровськ  
2013

# Зміст

Вступ.....	3
Теоретична частина.....	4
Способи вирішення проблем транспортної системи.....	4
Значення моделювання в оптимізації міського трафіку.....	4
Способи моделювання транспортних потоків .....	4
Опис моделі та програми.....	6
Модель міста в програмі та її реалізація .....	6
Структура програми.....	7
Опис алгоритмів .....	7
Посібник користувача .....	11
Призначення та загальний вигляд програми .....	11
Робота з містом.....	11
Редагування міста.....	12
Моделювання .....	14
Технічні вимоги.....	15
Приклади використання програми .....	16
Висновки .....	19
Список використаної літератури.....	20
Додатки.....	21
Додаток 1. Приклади програмного коду.....	21

# Вступ

## Актуальність роботи

Організація автомобільного руху є серйозною проблемою великих міст. Транспортна система не витримує постійно зростаючого потоку, у зв'язку з чим виникають затори. Вони, у свою чергу, викликають забруднення довкілля, зайві витрати енергоресурсів та часу пасажирів.

Цю проблему можна розв'язати, розбудовуючи та реорганізуючи транспортну систему, для чого потрібні значні кошти та час. Тому важливо знати, якими будуть наслідки цих змін. При проектуванні нових міст також важливо передбачити слабкі місця їх транспортної системи.

Нині стрімко зростає кількість автомобілів у населення. Станом на 2012 рік у світі налічувалося близько 1,7 млрд. автомобілів, а впродовж 2011 року було продано 60 млн. автомобілів. Інтенсивність автомобільного руху у великих містах перевищує пропускну здатність транспортної системи. Так, у Лос-Анджелесі (США) налічується 6 мільйонів автомобілів, у Москві (Росія) – 4 мільйони, у Києві – 800 тисяч. Це призводить до виникнення заторів. Наприклад, у Москві середня швидкість руху автомобілів по місту складає 33 км/год, а в центральній частині міста – 18 км/год.

Звідси випливає, що проблеми оптимізації міського руху сьогодні є надзвичайно актуальними для великих міст. Оскільки моделювання є одним з основних інструментів для вирішення цієї проблеми, дана робота є актуальною.

## Мета роботи

Транспортну систему міста зручно аналізувати методом комп'ютерного моделювання. Дана робота присвячена розробці комп'ютерної моделі цієї системи та її реалізації у програмному продукті.

## Задачі роботи

1. Проаналізувати існуючі методи моделювання міського автомобільного руху.
2. Розробити математичну модель та алгоритми для моделювання міського трафіку.
3. Створити програмний продукт для моделювання.
4. Інтегрувати до нього інструменти для аналізу транспортної системи, створивши таким чином систему підтримки прийняття рішення для оптимізації міського трафіку.
5. Розробити технічну документацію до програмного продукту.

Модель – це штучний об'єкт, який відображує важливі для її розробника властивості іншого природного або штучного об'єкту (оригіналу). Комп'ютерне моделювання – це процес створення та вдосконалення програми, яка реалізує дану модель.

# **Теоретична частина**

## **Способи вирішення проблем транспортної системи**

Найпростішим способом розв'язання проблем виникнення заторів, на перший погляд, є розширення пропускної здатності існуючих доріг. Але це, як правило, неможливо здійснити, оскільки вздовж доріг розташовано будинки. До того ж, реконструкція доріг пов'язана з великими фінансовими витратами та тимчасовими незручностями, які погіршують транспортну ситуацію.

Інший спосіб боротьби з заторами – створення нових магістральних шляхів та об'їзних доріг. Це допомагає розвантажити центральні частини міст, де, як правило, дороги вузькі, від транзитних потоків, але вимагає великих витрат і пов'язане зі складною проблемою виділення територій під будівництво доріг.

Ще один спосіб боротьби із заторами – оптимізація трафіку за допомогою світлофорів, розмітки доріг, пере напрямлення транспортних потоків. Цей спосіб є найменш затратним, але потребує певних теоретичних розрахунків.

Крім цих способів, спрямованих безпосередньо на оптимізацію інфраструктури, існують інші, метою яких є зменшення транспортних потоків – створення мережі громадського транспорту та заохочення населення користуватися ним, заборона проїзду для певних видів транспорту тощо.

## **Значення моделювання в оптимізації міського трафіку**

При реорганізації транспортних систем зазвичай проводять інженерні розрахунки, які базуються на даних відслідковування динаміки транспортних потоків. Але вони не можуть у повному обсязі передбачувати наслідки, тому що транспортний потік буде підлаштовуватись під зміни транспортної системи.

Моделювання є необхідним для прийняття рішень щодо оптимізації транспортної системи через наступні її властивості.

- Компенсація зростання пропускної здатності при розвитку мережі збільшенням попиту на неї.
- Непередбачуваність поведінки кожного водія.
- Вплив випадкових факторів (ДТП, погодні умови, свята тощо).

## **Способи моделювання транспортних потоків**

Всі моделі транспортних потоків можна розбити на 3 класи [1]:

- **Моделі-аналоги, або макроскопічні.** Рух транспортних засобів уподібнюється певному фізичному потоку та описується рівняннями гідро- або газодинаміки.
- **Моделі слідування за лідером, або мікроскопічні.** Окремо розглядається рух кожного автомобіля. Враховується той факт, що водії узгоджують швидкість руху власної машини зі швидкістю руху автомобіля попереду.
- **Ймовірнісні моделі.** У них транспортний потік розглядається як стохастичний процес.

Модель, яка розглядається в даній роботі – мікроскопічна. Її перевага в тому, що вона найбільш точно описує поведінку транспортної системи в реальному часі (відлік часу в моделі співпадає з відліком часу в реальності). Крім того, вона є наочною.

# Опис моделі та програми

Під час виконання даної роботи створено програмний продукт **VirtualCity**. Для цього використано середовище **Microsoft Visual Studio 2010**, мова програмування **Visual Basic .NET**.

## Модель міста в програмі та її реалізація

У програмі використовується наступна математична модель міста. Воно відображується прямокутником певних розмірів. У деяких його точках розташовані перехрестя. Деякі з них сполучені дорогами. Дороги мають певну ширину - від нуля до шести смуг в обох напрямках (нуль смуг може бути лише в одному напрямку – тоді дана дорога одностороння). У місті є будинки, вони відображуються прямокутниками зі сторонами, паралельними осям координат.

Головним об'єктом дослідження в програмі є автомобіль. Він відображується прямокутником та прив'язується до міста координатами середини сторони з того боку, куди він рухається (далі називатимемо це «передньою точкою») та кутом повороту. Крім того, він може бути прив'язаний до дороги, на якій знаходиться, номером смуги та відстанню від початкового перехрестя. Автомобілі в програмі виїжджають з одних будинків та рухаються за наперед прорахованим маршрутом до інших.

Для кожного будинку відома його місткість (максимальна кількість автомобілів, що може в ньому знаходитись), кількість машин у ньому в даний момент часу та популярність (параметр, який визначає середню частоту в'їзду машин до даного будинку).

Описана модель реалізована в програмі за допомогою концепції об'єктно-орієнтованого програмування. Об'єктна система програми складається із шести класів:

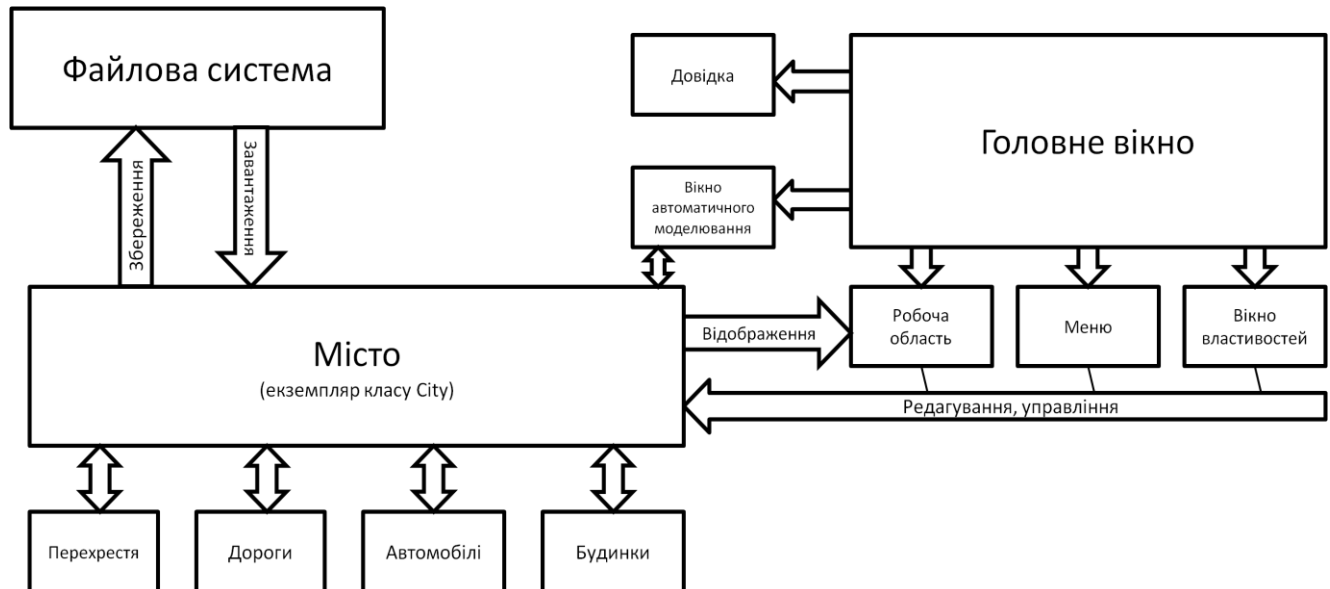
- **City** – місто. В екземплярі класу зберігаються параметри міста та всі його компоненти. Методи класу забезпечують редагування, відображення та моделювання міста.
- **Road** – дорога. Клас забезпечує відображення доріг.
- **Intersection** – перехрестя. В екземплярі класу зберігаються координати перехрестя. Методи забезпечують проїзд машин через перехрестя.
- **TrafficControl** – світлофор. Клас забезпечує проїзд машин через перехрестя зі світлофором.
- **Car** – автомобіль. Клас забезпечує моделювання руху та відображення автомобілів.

- **Building**— будинок. Клас забезпечує відображення будинків та їх взаємодію з містом.

При роботі з містом створюється екземпляр класу **City**, з яким взаємодіють всі елементи інтерфейсу програми.

## Структура програми

На наведеній блок-схемі зображена загальна структура програми.



## Опис алгоритмів

### Моделювання

Моделювання в програмі відбувається дискретно із кроком по часу **dt** (в програмі - 50 мс). Це реалізується за допомогою компоненту **Timer**. На кожному кроці відбуваються наступні дії:

**1.Рух автомобілів.** Для кожного автомобіля прораховується, на яку відстань він має зміститись, рухаючись із максимальною швидкістю. Потім перевіряється, чи не наїжджає він при цьому на автомобіль, що рухається попереду. Якщо так, то ця відстань зменшується. Потім координата машини на дорозі збільшується на цю відстань.

Крім того, виконуються наступні додаткові перевірки.

- Якщо попереду даної машини рухається машина з меншою швидкістю та сусідній ряд є вільним, вона перестроюється до нього.
- Якщо машина під'їжджає до кінця дороги, надсилається запит на проїзд до відповідного перехрестя.

- Якщо відомо, що місце, куди прямує машина, знаходиться на даній дорозі, і вона наблизилась до цього місця, то машина відкріплюється від дороги та рухається прямолінійно до відповідного будинку; після прибуття до нього вона зникає, а кількість автомобілів у цьому будинку збільшується на одиницю. Якщо кількість автомобілів у будинку-цілі дорівнює його місткості, машина продовжує рух до найближчого будинку, де є вільні місця.

Окремо моделюється рух автомобілів, що виїжджають на дорогу. Вони рухаються прямолінійно до точки виїзду на дорогу. При підїзді до цієї точки, якщо там немає іншої машини, вони виїжджають на дорогу.

**2. Проїзд автомобілів на перехрестях.** Автомобілі зі списку **MovingCars** на всіх перехрестях зміщуються на відстань  $dt * V_{\Pi}$ , де  $V_{\Pi}$  – швидкість проїзду перехрестя. Потім перевіряється, чи є довжина шляху, пройденого через перехрестя, більшою від розрахованої довжини траєкторії проїзду перехрестя. Якщо так, машина відкріплюється від перехрестя та прикріплюється до дороги, на яку має виїхати, при цьому їй встановлюється координата, що відповідає початку нової дороги.

**3. Випадкова генерація автомобілів у будинках.** Для кожного будинку генерується випадкове число  $Z$  від 0 до 1 та порівнюється з величиною  $dt * V$ , де  $dt$  – крок моделювання (с),  $V$  – ймовірність виїзду автомобіля протягом секунди. Ймовірність пропорційна наповненості будинку:  $V = V_0 * (N/X)$ , де  $V_0$  – константа (у програмі – 0.2; це значення підібрано експериментально з тих міркувань, що при такому значенні коефіцієнта одержуємо картину трафіку, досить близьку до реальної),  $N$  – кількість машин у будинку,  $X$  – місткість будинку. Якщо  $Z < dt * V$ , у будинку є хоча б одна машина та дорога, на яку має виїжджати машина, вільна, створюється новий автомобіль, який виїжджає з даного будинку (кількість машин у кожному будинку в початковий момент задана). Для створеного автомобіля знаходиться будинок, до якого він має рухатись та розраховується маршрут. При виїзді машини з будинку кількість автомобілів у ньому зменшується на одиницю.

### **Визначення цілі та оптимального маршруту автомобілів**

Для кожної машини при її генерації визначається ціль – будинок, до якого вона має рухатись. Це робиться так, щоб для будинків із більшою популярністю була більша ймовірність того, що його буде обрано. Нехай  $P_1, P_2, \dots, P_n$  – популярності будинків. Позначимо  $S(i) = P_1 + P_2 + \dots + P_i$ . Програма генерує випадкове число  $X$  в інтервалі від 0 до  $S(n)$ . Ціллю обирається будинок з номером  $Y$  – такий, що  $S(Y-1) \leq X < S(Y)$ .

Після вибору цілі для машини визначається маршрут її руху, а саме послідовність перехресть, які вона проїжджає. Сукупність доріг розглядається як орієнтований граф, дуги якого – дороги (дорога з двостороннім рухом відображується двома дугами з різними напрямками, з одностороннім – однією дугою), а вершини –



перехрестя та точки в'їзду й виїзду даної машини. Вага ребер визначається як час проїзду дороги – відношення її довжини до середньої швидкості руху машин даною дорогою або до швидкості даної машини, якщо на дорозі машин немає або мало. Найкоротший шлях в даному графі розраховується за алгоритмом Дейкстри. Таким чином, коли рух автомобілів нормальний, обирається маршрут найменшої довжини з усіх допустимих, а коли в місті є затори, обирається маршрут найменшого часу.

### Пріїзд автомобілів через перехрестя

Кожне перехрестя має два списки: 1) **WaitingCars**, де зберігаються номери машин, що стоять безпосередньо перед перехрестям та очікують проїзду; 2) **MovingCars** – де зберігаються номери машин, що рухаються перехрестям. Коли машина під'їжджає до перехрестя, до нього посилається запит на проїзд. При цьому розраховується початкова та кінцева точки траєкторії, за ними – кут та довжина шляху по перехрестю (він є прямолінійним), автомобіль видаляється з дороги та додається до списку **WaitingCars**. Раз на 500 мс для всіх машин цього списку визначається, чи можуть вони почати рух. Для цього їх шлях має не перетинати шлях машин, що вже рухаються перехрестям. Якщо на перехресті є світлофор, то перевіряється, чи дозволений у даний момент часу проїзд даного автомобіля. Якщо дане перехрестя – з головною дорогою, то спочатку визначається можливість проїзду для машин на головні дорозі. Якщо дозвіл на проїзд отримано, номер машини переводиться до списку **MovingCars**. Моделювання руху машин зі списку **MovingCars** описано вище.

### Взаємодія автомобілів та будинків

Кожному будинку зіставляється точка **P** на певній дорозі, яка є основою перпендикуляру, опущеного з центра будинку на неї. Дорога обирається таким чином, щоб довжина перпендикуляру була мінімальною.

Коли машина виїжджає з будинку, вона спочатку рухається прямолінійно від його центру до точки **P**. Коли вона досягає її, вона прикріплюється до даної дороги. Коли машина досягає точки, що відповідає будинку-цілі, вона відкріплюється від дороги та рухається до центру будинку. Після досягнення центру машина видаляється.

### Візуалізація руху автомобілів

Автомобілі в програмі, крім розташування на дорозі, характеризуються координатами передньої точки та напрямком руху. При цьому напрямок руху може миттєво змінюватись. Щоб у такому випадку машина не розверталась миттєво, кут повороту прямокутника машини перераховується при кожному кроці пересування машини таким чином, ніби її тягнуть, прикладаючи силу до передньої точки. Це створює ефект повороту при русі через перехрестя, виїзді на дорогу або з дороги.

### **Визначення заторів**

Для кожної машини на кожному кроці моделювання визначається її миттєва швидкість, як відношення фактичного зміщення до часу. Потім для кожної дороги, на якій є машини, визначається середнє арифметичне цих швидкостей. Якщо на певній дорозі довго тримається низьке значення цього параметру (менше 10 м/с) та велика кількість машин (більше десяти), виводиться інформація про затор: дорога виділяється рамкою, колір якої змінюється від жовтого до червоного в залежності від середньої швидкості руху.

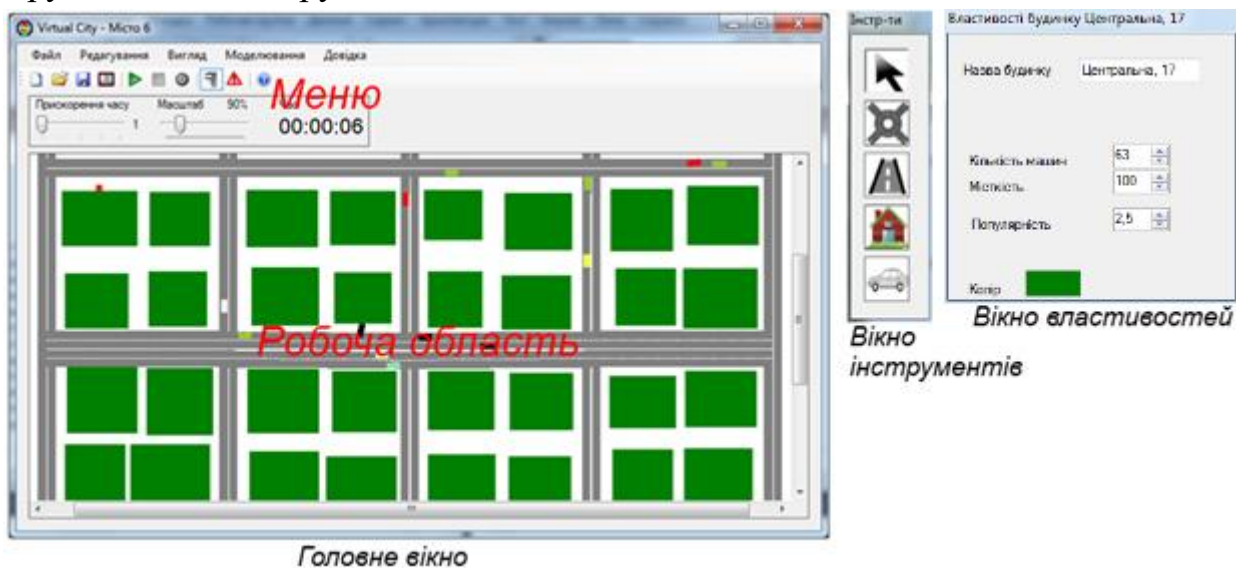
# Посібник користувача

## Призначення та загальний вигляд програми

**VirtualCity** – це програмний продукт, призначений для комп'ютерного моделювання міського автомобільного руху. Він містить набір інструментів для створення моделі міста, моделювання міського трафіку та систему підтримки прийняття рішень для його оптимізації.


### Загальний вигляд програми



Головне вікно програми складається з меню та робочої області. Також програма має рухомі вікна інструментів та властивостей.




## Робота з містом

Програма **VirtualCity** зберігає всі дані про досліджуване місто у текстових файлах з розширенням **\*.vcity**, з яких можна пізніше завантажувати ці дані. Файл з даними про місто у програмі називається «Місто».

**Створення міста.** Виберіть пункт меню **Файл|Нове місто**, або натисніть кнопку , або натисніть **Ctrl+N**. Тоді з'явиться діалогове вікно. Введіть до нього назву міста, його ширину та висоту й натисніть **ОК**. Після цього ви зможете редагувати представлення створеного міста в робочій області.





**Завантаження/збереження міста.** Для завантаження міста виберіть пункт меню **Файл|Завантажити**, або натисніть кнопку , або натисніть **Ctrl+O**. Після цього виберіть файл та натисніть **ОК**. Для збереження міста до файлу виберіть пункт меню **Файл|Зберегти** або натисніть кнопку , або натисніть **Ctrl+S**. Після цього виберіть місце збереження файлу, введіть його ім'я та натисніть **ОК**.


**Експорт зображення міста.** Ви можете зберегти поточний вигляд міста до графічного файлу. Для цього виберіть пункт меню **Файл|Експорт зображення** або натисніть кнопку .

**Масштабування та прокрутка.** За допомогою пункту меню «Масштаб» можна задати потрібний масштаб міста. Якщо при цьому все місто не вміщується до робочої області, ви можете прокручувати його за допомогою смуг прокрутки у нижній та правій частинах робочої області.

## Редагування міста

**Додавання нових об'єктів.** В концепції **VirtualCity** місто складається з перехресть, доріг, автомобілів та будинків.

Спочатку виберіть інструмент **Перехрестя** () та за допомогою лівої кнопки миші (ЛКМ) відмітьте перехрестя в необхідних місцях. Потім виберіть інструмент **Дорога** () та з'єднайте деякі перехрестя, натискаючи на них по черзі. Потім виберіть інструмент **Будинок** () та відмітьте будинки. Для цього наведіть курсор на ліву верхню точку будинку, затисніть ЛКМ, протягніть його до нижньої правої точки будинку та відпустіть. Для того, щоб додати автомобілі, виберіть інструмент **Автомобіль** () та натисніть ЛКМ в потрібних місцях доріг.

**Редагування об'єктів.** Для редагування міста користуйтеся інструментом вибору (). Для вибору об'єкту натискайте на ньому ЛКМ.

Щоб перемістити перехрестя, затисніть на ньому ЛКМ, протягніть до необхідної точки та відпустіть. Якщо при цьому дороги будуть проходити крізь дома чи перетинатися, переміщення не відбудеться.

Для того, щоб змінити розмір будинку, наведіть курсор на його межу, затисніть ЛКМ та перемістіть цю межу.

Для редагування параметрів міста (назва, ширина, висота та колір фону) натисніть ЛКМ на порожній ділянці міста та задайте відповідні параметри у вікні властивостей.

**Налаштування об'єктів міста.** За допомогою вікна властивостей можна задавати параметри об'єктів міста. Для цього виберіть необхідний об'єкт та введіть відповідні параметри. Для автомобілів можна встановлювати максимальну швидкість, для будинків – кількість машин у них, місткість, популярність та колір, для доріг – кількість рядів в кожному напрямку(від нуля до шести). Також для всіх об'єктів можна задавати назву.

**Властивості будинків.** У вікні властивостей для кожного будинку задаються такі параметри, як місткість, кількість машин та популярність.

Місткість будинку – це максимальна кількість машин, які можуть знаходитись у ньому. Кількість машин задається в інтервалі від нуля до місткості. Вона змінюється у процесі моделювання. В режимі моделювання не можна змінювати ці два параметри. Відношення кількості автомобілів до місткості визначає частоту виїзду автомобілів.

Популярність – це коефіцієнт, який визначає інтенсивність потоку машин до даного будинку відносно інших будинків. У скільки разів популярність одного будинку більша за популярність іншого, у стільки разів більше машин (в середньому) в'їжджатиме до нього за одиницю часу.

**Візуалізація властивостей будинків.** На будинках при увімкненій опції **Вигляд|Показувати відмітки** відображаються властивості будинків. На кожному будинку зображується коло та квадрат. Колір кола вказує на міру заповненості будинку (відношення кількості машин в ньому до місткості). Колір квадрата вказує на популярність. Кольори змінюються від зеленого (мінімальне значення) до червоного (максимальне значення).

**Способи регулювання перехресть.** В програмі існують три види перехресть – нерегульоване, з головною дорогою та зі світлофором. Їх можна задавати у вікні властивостей. На нерегульованому перехресті машини перетинають перехрестя в тому порядку, в якому під'їжджають, на перехресті з головною дорогою пріоритет надається машинам з головної дороги. Для того, щоб задати головну дорогу, потрібно вказати дві дороги, що підходять до даного перехрестя і утворюють головну дорогу. На перехресті зі світлофором у певні моменти часу машини можуть рухатись лише з заданих доріг на інші задані дороги. Щоб створити програму світлофора, необхідно задати його цикл  $C$  у секундах. Потім необхідно задати набір режимів. Режим має чотири параметри: час початку  $t1$ , час закінчення  $t2$  ( $0 \leq t1, t2 \leq C$ ), дорогу-в'їзд  $r1$  та дорогу-виїзд  $r2$ . Такий режим означає, що через дане перехрестя машини з дороги  $r1$  на дорогу  $r2$  можуть рухатись лише в проміжки часу міста  $[C \cdot n + t1; C \cdot n + t2]$ , де  $n=0,1,2,\dots$ . Щоб задати режим, натисніть кнопку «Додати» у вікні властивостей та введіть необхідні параметри. Щоб видалити режим, натисніть «Видалити». На перехресті з чотирма дорогами можна встановити стандартну програму, яка по черзі пропускатиме машини з під'їздів, які утворюють одну дорогу. Для цього натисніть кнопку «Авто».




**Рекомендації щодо налаштування міста.** Програма розрахована на моделювання ситуацій, наближених до реальності. Тому рекомендується створювати такі моделі міст, які могли б існувати в реальності. Не робіть перехрестя, з яких виходить більше ніж п'ять доріг, не задавайте дуже велику ймовірність виїзду машин із будинків, не робіть програми світлофорів, які не пропускають машини в якихось напрямках.


Також особливості розрахунку маршрутів вимагають, щоб у місті не було перехресть, з яких виходить лише одна дорога.

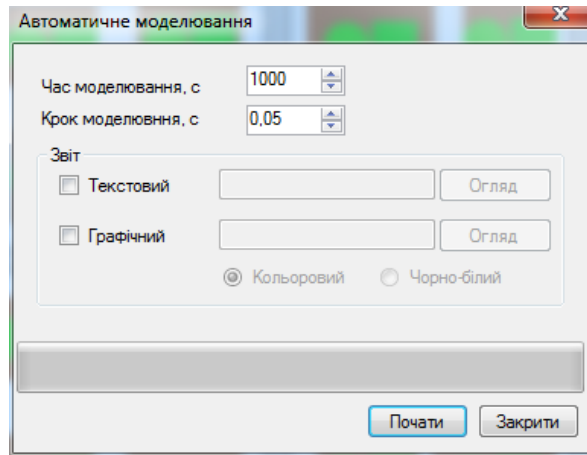
**Додаткові можливості редагування.** В пункті меню «Редагування» наявні наступні інструменти:

- **Місто.**
  - **Створити сітку кварталів** – створити систему із перехресть і доріг, яка задає класичний вигляд міста із квадратними кварталами.
  - **Очистити** – видалити всі об'єкти.
  - **Вирівняти по сітці** – перемістити перехрестя та вершини будинків до найближчих вузлів уявної сітки.
- **Автомобілі.**
  - **Видалити всі автомобілі.**
- **Будинки.**
  - **Задати кількість автомобілів** – задати для всіх будинків однакову місткість та кількість автомобілів.
  - **Задати один колір.**
  - **Задати випадкові кольори.**

## Моделювання

**Моделювання в реальному часі.** Для того, щоб розпочати моделювання, виберіть пункт меню **Моделювання|Почати**, натисніть кнопку  або клавішу F5. Для того, щоб припинити моделювання, виберіть пункт меню **Моделювання|Зупинити**, натисніть кнопку  або клавішу F6. Моделювання відбуватиметься в реальному часі, але для його прискорення можна змінювати прискорення руху внутрішнього годинника програми за допомогою пункту меню «**Прискорення часу**» від 1 до 5. Під час моделювання не рекомендується редагувати місто. Якщо під час моделювання на деяких дорогах з'являтимуться затори, ці дороги будуть обведені рамкою, колір якої змінюватиметься від жовтого до червоного в залежності від «важкості» затору. Щоб увімкнути/вимкнути цю опцію, використовуйте пункт меню **Вигляд|Показувати відмітки** або кнопку . Під час моделювання в головному меню відображається поточний час міста у форматі ГГ:ХХ:СС.

**Автоматичне моделювання.** В програмі передбачена можливість автоматичного моделювання. Для цього виберіть пункт меню **Моделювання|Автоматичне моделювання**, натисніть кнопку  або клавішу F7. Після цього з'явиться діалогове вікно:



В ньому необхідно задати час моделювання та крок моделювання (від 0,05с до 1с). Після натискання кнопки **«Почати»** програма буде моделювати рух автомобілів за вказаний час. Статус моделювання буде відображати заповнення смуги. Після завершення моделювання буде відображений стан міста на момент завершення.

**Звіти автоматичного моделювання.** Програма може створювати звіти, в яких відображаються параметри, що характеризують місто під час моделювання (для будинків – кількість автомобілів, що в'їхали та виїхали; для доріг – кількість автомобілів, що проїхали та середня швидкість проїзду). Текстовий звіт – це текстовий файл, де ці параметри представлені у вигляді таблиці. Графічний звіт – зображення міста (будинки, дороги та параметри, написані на них). Крім того, швидкість проїзду машин відображена за допомогою кольорів доріг (на кольоровому звіті – відтінками червоного й жовтого, на чорно-білому – відтінками сірого). Ці звіти допомагають проаналізувати стан транспортної системи даного міста протягом тривалого часу.

Для того, щоб створити звіт, помітьте відповідні опції у вікні моделювання перед моделюванням та вкажіть місце збереження звіту. По закінченню моделювання можна відкрити ці звіти за допомогою кнопки **«Відкрити»**.

## Технічні вимоги

### Апаратні вимоги

- Процесор з частотою 2 GHz або вище;
- RAM 2GB або вище;
- Відеоадаптер 512 MB або вище;
- Монітор, клавіатура, миша;
- 5 MB вільного місця на жорсткому диску.

### Програмні вимоги

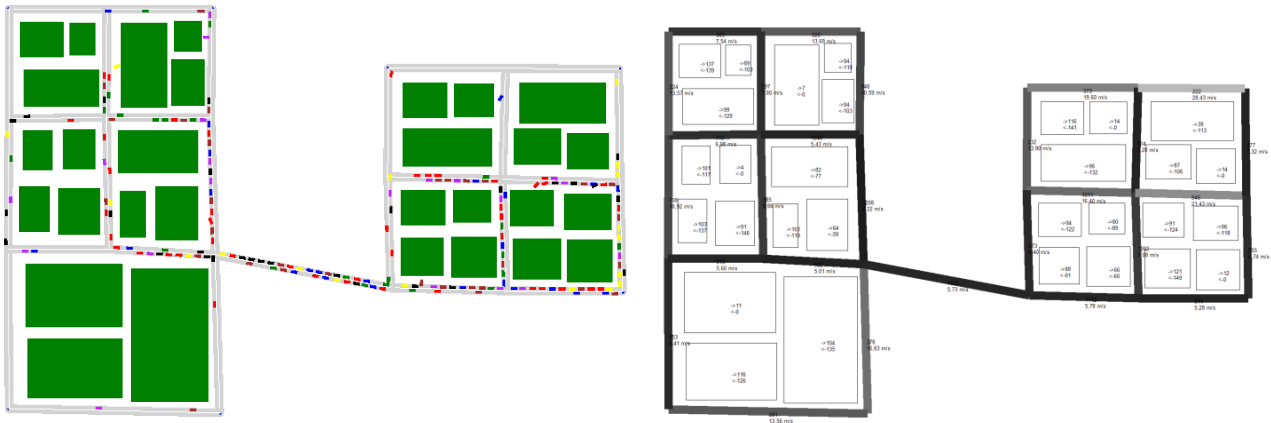
- ОС Windows XP, Vista, 7, 8;
- .NET Framework 4.5.

# Приклади використання програми

Нижче наводиться ряд прикладів, що демонструють використання засобів моделювання та аналізу створеної програми для оптимізації транспортних систем. Вони проілюстровані зображеннями міст та графічними звітами автоматичного моделювання.

## Приклад 1

Маємо місто, що складається із двох частин, поєднаних однією дорогою. Її пропускна здатність менша, ніж автомобільний потік між частинами міста, тому на ній та на прилеглих до неї дорогах виникає затор.



Розширимо цю дорогу до двох смуг в обидва боки. При цьому середня швидкість руху автомобілів по ній та сусідніх дорогах збільшується.



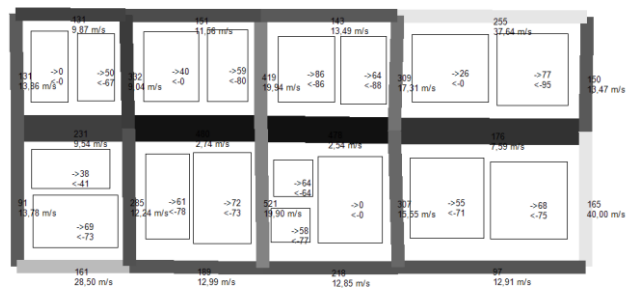
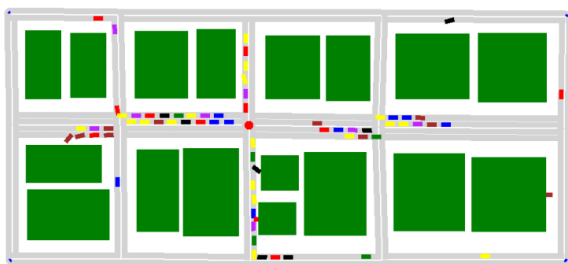
Замість того, щоб розширювати дорогу, можна побудувати нові дороги між частинами міста, знімаючи таким чином частину навантаження з першої дороги.



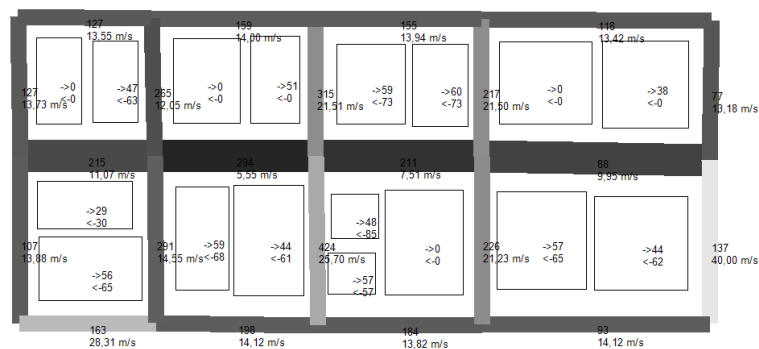


## Приклад 2

Маємо проспект, на якому встановлені три світлофори (на перехрестях, що мають по чотири дороги). Цикл їх роботи складає 100с. При цьому машини вимушені довго стояти при червоному світлі, коли перпендикулярна дорога вільна. Це призводить до невиправданих заторів.

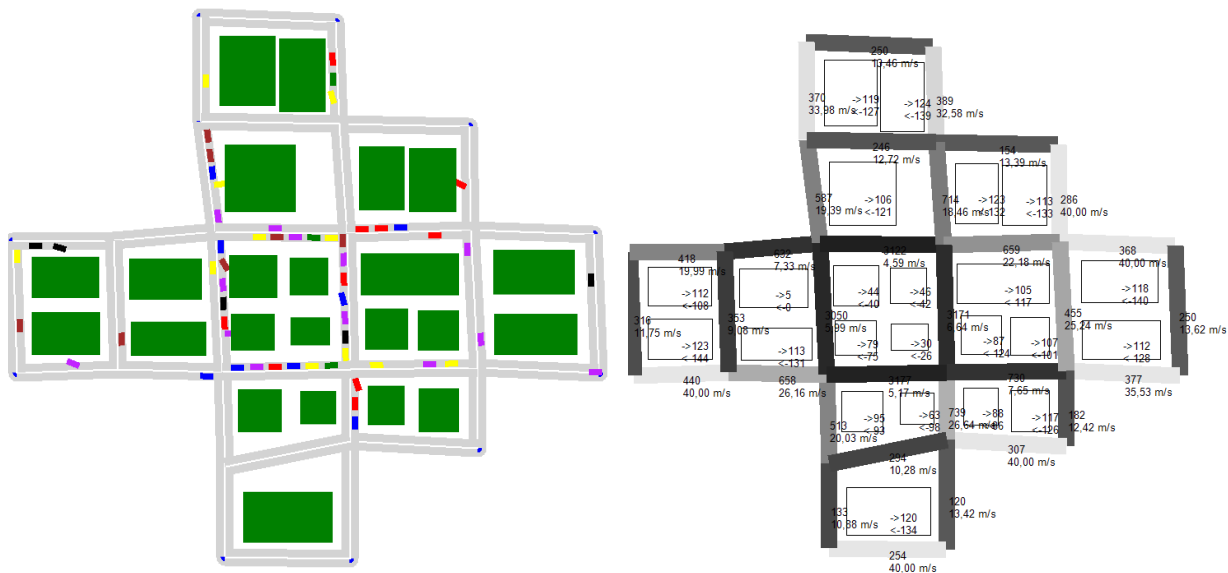


Зменшимо цикл роботи світлофора до 30 с. При цьому зростає середня швидкість на всіх дорогах.

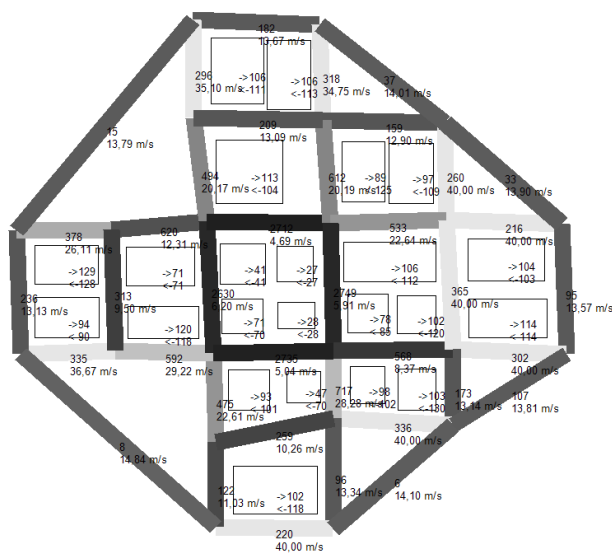


## Приклад 3

Маємо місто із досить активним рухом та низькою пропускнуою здатністю доріг (це характерно для старих частин багатьох сучасних міст).



Побудуємо об'їзні дороги. Вони пропускають частину автомобілів, які раніше вимушені були рухатись через центр. При цьому ситуація в центрі міста поліпшується.



# Висновки

Нижче наведено головні результати моєї роботи.

1. Проаналізовано існуючі методи моделювання міського трафіку, розроблено власну мережну модель міської транспортної системи та алгоритми комп'ютерного моделювання для неї.

2. Створено програмний продукт для комп'ютерного моделювання міського автомобільного руху. Він містить набір інструментів для створення моделі міста (редактор), моделювання міського трафіку та систему підтримки прийняття рішень для його оптимізації. Розроблено технічну документацію для підтримки роботи з продуктом.

3. Під час виконання роботи я поглибив свої знання мови програмування **Visual Basic .NET**, концепції об'єктно-орієнтованого програмування, теорії транспортних потоків, теорії графів та теорії алгоритмів.

Створена програма може бути використана місцевими органами самоврядування великих міст при проектуванні, розбудові та оптимізації транспортної системи міст або транспортними організаціями при прокладанні маршрутів. Також вона може використовуватись для теоретичних досліджень.

Я бачу наступні перспективи розвитку теми роботи.

- Розгляд більш складних моделей (врахування міського транспорту, пішоходів, зміни інтенсивності та напрямку руху в залежності від часу доби, погодних умов тощо).
- Створення можливості автоматичної оптимізації транспортної системи міста, що моделюється.

# Список використаної літератури

- 1.Семенов В.В. Математическое моделирование динамики транспортных потоков мегаполиса - Москва, Институт прикладной математики им. М.В. Келдыша – 2004. – 44с.
2. Тымкив К. Хорошо стоим. Чем обернулся рост количества автомобилей в украинских мегаполисах. //Кореспондент. – 2012. – №48. – С.38-42.
- 3.Чарльз Петцольд. Программирование для Microsoft Windows на Microsoft Visual Basic .NET – М.: «Русская редакция» – 2003. – В 2-х томах.
- 4.Окулов С.М. Программирование в алгоритмах – М.: БИНОМ – 2002 – 341с.
- 5.Климов А. Занимательное программирование на Visual Basic .NET – С-Пб.: «БХВ-Петербург» – 2005. – 528 с.
- 6.Дэн Кларк.Объектно-ориентированное программирование в Visual Basic .NET. – 2003. – С-Пб.: «Питер» – 352с.
7. Уизем Дж. Линейные и нелинейные волны. М.: Мир, 1977 – 624 с.

# Додатки

## Додаток 1. Приклади програмного коду

Метод класу **City**, який моделює рух в місті за малий проміжок часу **dt**.

```
Public Sub ModelStep(ByVal dt As Single)
    'Рухаємо всі машини
    For i = 1 To CarsCounter
        If Cars(i).Exists Then Cars(i).Move(dt, Me)
    Next
    'Рухаємо машини на перехрестях
    For i = 1 To ISCounter
        If IntScs(i).Exists Then IntScs(i).Move(dt, Me)
    Next
    'Випадковий виїзд із будинків
    For i = 1 To HsCounter
        If Rnd() < Building.MaxOutPossibility * (Houses(i).CarsNum / Houses(i).MaxCarsNum) * _
            dt Then Houses(i).CreateCar(Me)
    Next
    'Збільшуємо час міста
    Time += dt
End Sub
```

Функція класу **Car**, яка визначає прямокутник, що зображує машину.

```
Public Function GetRect(ByRef CT As City) As PointF()
    Dim pnts(4) As PointF
    'Перпендикуляр до напрямку орієнтації машини
    Dim ang1 As Single = TrueRot - PI / 2
    Dim p3 As PointF = Pos 'Передня точка машини
    'Задня точка машини
    Dim p4 As PointF = New PointF(p3.X - Length * Cos(TrueRot), p3.Y - Length * Sin(TrueRot))
    'Визначаємо координати вершин прямокутника
    pnts(0) = New PointF(p3.X - 0.5 * Width * Cos(ang1), p3.Y - 0.5 * Width * Sin(ang1))
    pnts(1) = New PointF(p3.X + 0.5 * Width * Cos(ang1), p3.Y + 0.5 * Width * Sin(ang1))
    pnts(3) = New PointF(p4.X - 0.5 * Width * Cos(ang1), p4.Y - 0.5 * Width * Sin(ang1))
    pnts(2) = New PointF(p4.X + 0.5 * Width * Cos(ang1), p4.Y + 0.5 * Width * Sin(ang1))
    pnts(4) = pnts(0)
    Return pnts
End Function
```

Метод класу **Car**, який «протаскує» машину на відстань **dist** під кутом **MovRot** за передню точку.

```
'Pos - положення передньої точки, MovRot - напрям руху, TrueRot - орієнтація
Public Sub Drag(ByVal dist As Single)
    'Зміщуємо передню точку
    Pos = New PointF(Pos.X + dist * Cos(MovRot), Pos.Y + dist * Sin(MovRot))
    'Визначаємо нову орієнтацію машини
    Dim l As Single = Me.Length
    Dim ang As Single = TrueRot - MovRot
    Dim x As Single = Sqrt(l * l + dist * dist + 2 * l * dist * Cos(ang))
    TrueRot -= Asin(dist / x * Sin(ang))
End Sub
```

Метод класу **Car**, який моделює зміщення машини за **Time** секунд.

```
Public Sub Move(ByVal Time As Single, ByRef CT As City)
```

```

'Відстань, яку має пройти машина
Dim dist As Single = Speed * Time

'Якщо машина виїжджає із будинку на дорогу
If Not AtRoad And GoToRoad Then
    GR_Pos += dist
    Drag(dist)
    If GR_Pos > GR_Len Then      'Якщо машина виїхала
        Me.GoToRoad = False
        Me.AtRoad = True
        Me.MovRot = CT.Roads(Me.Road).Rotat
        If Me.Line > 0 Then MovRot += PI
        CT.Houses(GR_House).GR_IsCarGoingOut = False
        Me.GR_House = 0
    End If
    Exit Sub
End If

'Якщо машина в'їжджає з дороги до будинку
If GoFromRoad Then
    GR_Pos += dist
    Drag(dist)
    If GR_Pos >= GR_Len Then
        CT.Houses(GR_House).Analyse_InComeCounter += 1
        Me.Exists = False
    End If

    Exit Sub
End If

'Якщо машина перестроюється
If ChangingLine Then
    CL_Pos += dist
    Drag(dist)
    If CL_Pos >= CL_Len Then
        ChangingLine = False
        Me.MovRot = CT.Roads(Me.Road).Rotat
        If Me.Line < 0 Then Me.MovRot += PI
        Me.TrueRot = Me.MovRot
    End If
    Exit Sub
End If

'З'їзд із дороги
If LastRoadOfRoute And PosAtLine + dist >= CT.Houses(TargetBuilding).Targ_PosAtLine Then
    dist = CT.Houses(TargetBuilding).Targ_PosAtLine - PosAtLine
    Me.GoFromRoad = True
    Me.RouteDetermined = False
    Me.MovRot = Geometry.AngleBy2Pt(Me.Pos, CT.Houses(TargetBuilding).Targ_InPoint)
    Me.GR_Len = Geometry.Dist(Me.Pos, CT.Houses(TargetBuilding).Targ_InPoint)
    Me.GR_Pos = 0
    Me.GR_House = TargetBuilding
    PosAtLine += dist
    Pos = GetPos(CT)
    Exit Sub
End If

'Якщо попереду є машина, зменшуємо швидкість
Dim IsCarForward As Boolean = False
Dim SafeDist As Single = City.SafeDist
For i = 1 To CT.CarsCounter
    If CT.Cars(i).Exists And Not (i = Me.Number) And CT.Cars(i).Road = Me.Road _
    And CT.Cars(i).Line = Me.Line Then
        Dim CrBack As Single = CT.Cars(i).PosAtLine - CT.Cars(i).Length
        If CrBack > PosAtLine And CrBack - SafeDist < PosAtLine Then

```

```

        dist = 0
        IsCarForward = True
    ElseIf CrBack > PosAtLine And CrBack < PosAtLine + dist Then
        dist = CrBack - SafeDist - PosAtLine
        IsCarForward = True
    End If
End If
Next

'Спроба перестроювання
Dim LineChOk As Boolean = False
If IsCarForward And Me.PosAtLine < CT.Roads(Me.Road).Length - IntZone(CT) - Me.Length Then
    If Me.Line > 0 Then
        If CT.Roads(Me.Road).n1 > Me.Line Then
            TryStartChangeLine(Me.Line + 1, CT, LineChOk)
            If Me.Line > 1 Then TryStartChangeLine(Me.Line - 1, CT, LineChOk)
        End If
    Else
        If CT.Roads(Me.Road).n2 > Abs(Me.Line) Then
            TryStartChangeLine(Me.Line - 1, CT, LineChOk)
            If Me.Line < -1 Then TryStartChangeLine(Me.Line + 1, CT, LineChOk)
        End If
    End If
End If
If LineChOk Then Exit Sub

'Якщо машина доїхала до перехрестя
If PosAtLine + dist > CT.Roads(Road).Length - IntZone(CT) Then
    dist = CT.Roads(Road).Length - IntZone(CT) - PosAtLine
    IST_NewRoad = GetNextRoad(CT)
    Dim NextIntersection As Integer = NextInsc(CT)
    CT.IntScs(NextIntersection).RequestPass(Number, IST_NewRoad, CT)
End If

'Зміщуємо машину
PosAtLine += dist
Drag(dist)
Pos = GetPos(CT)
End Sub

```