КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ТАРАСА ШЕВЧЕНКА

ФАКУЛЬТЕТ РАДІОФІЗИКИ, ЕЛЕКТРОНІКИ ТА КОМП'ЮТЕРНИХ СИСТЕМ

Кафедра радіотехніки та радіоелектронних систем

До захисту допущено: «На правах рукопису»
Завідувач кафедри _____ Ігор АНІСІМОВ
« __ » червня 2023 р.

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему:

«РОЗРОБКА РОЗУМНОГО АВТОМОБІЛЯ В 5G, ІНТЕЛЕКТУАЛЬНИХ ТЕЛЕКОМУНІКАЦІЯХ V2X З ВИКОРИСТАННЯМ C++»

Виконав:
студент 4-го курсу
денної форми навчання
спеціальності 172 - Телекомунікації та радіотехніка
ОП «Інформаційна безпека телекомунікаційних систем і мереж»
Ільчук Олександр Миколайович _____

Науковий керівник:
к.т.н., ас. Аль Шурайфі МУШТАК _____

Рецензент:
к.т.н., доц. Рєзніков Михайло Ігорович _____

Засвідчую, що у цій бакалаврській роботі
немає запозичень з праць інших авторів без
відповідних посилань
Студент _____

Робота допущена до захисту в ЕК рішенням кафедри радіотехніки та радіоелектронних систем від «22» червня 2023 р., протокол № 21.

Завідувач кафедри радіотехніки та радіоелектронних систем,
доктор фіз.-мат. наук, професор
Анісімов Ігор Олексійович _____

Київ – 2023

АНОТАЦІЯ

У цьому дослідженні вивчається роль і застосування технології V2X в інтелектуальних транспортних системах, з особливим акцентом на моделюванні трафіку і мереж в сфері телекомунікацій. У ній розглядається важливість даних про трафік, архітектура мережі V2X і актуальність безпеки і зв'язку в автомобільному середовищі, а також потреба в ефективних стратегіях розподілу ресурсів в мережі NR-5G.

Використовуючи симулятори SUMO та ns-3 для моделювання трафіку та мережі відповідно, було створено реальний сценарій трафіку в Голосіївському районі Києва та проаналізовано його за різних мережевих умов. Моделювання проводилося у двох різних режимах: з вимкненим та увімкненим режимом User Datagram Protocol UDP, що дозволило оцінити продуктивність V2X-зв'язку за цих умов.

Метою цієї дипломної роботи є розгляд фундаментальних механізмів технології V2X, вивчення принципів моделювання трафіку та мереж у V2X, а також практична реалізація цих концепцій з метою поглиблення розуміння та сприяння створенню більш ефективних, безпечних та результативних транспортних систем.

Завдання, поставлені перед цією роботою, включали детальний розгляд технології V2X та її ролі в інтелектуальних транспортних системах, огляд моделювання трафіку і мереж, вибір і застосування симуляторів SUMO і ns-3 для моделювання трафіку і мереж, а також практичний аналіз симуляції на основі реального сценарію трафіку.

Ключові слова: Vehicle-to-Everything (V2X), інтелектуальні транспортні системи (ITS), телекомунікації, дані про дорожній рух, архітектура мережі, безпека і зв'язок, розподіл ресурсів, мережа NR-5G, SUMO, ns-3, User Datagram Protocol (UDP), моделювання трафіку, моделювання мережі.

ЗМІСТ

ВСТУП

Зі стрімким розвитком технологій та все більшою інтеграцією інформаційно-комунікаційних технологій в автомобільну промисловість, зв'язок між транспортними засобами став критично важливою сферою досліджень. Мільйони робочих годин втрачаються на дорогах через затори, а в деяких випадках важкі аварії значно ускладнюють життя. Згідно з доповіддю Всесвітньої організації охорони здоров'я (ВООЗ) [1], у 2013 році в результаті дорожньо-транспортних пригод загинуло 1,25 мільйона людей. Затори на дорогах також призводять до величезних втрат часу та пального, що призводить до величезних економічних втрат [2]. Ці основні проблеми можуть бути добре вирішені, якщо водії або інтелектуальні транспортні засоби будуть своєчасно оновлюватися з урахуванням поточних умов дорожнього руху. Як життєво важливий компонент інтелектуальних транспортних систем (ITC), V2X має потенціал для значного підвищення безпеки дорожнього руху, ефективності руху та енергозбереження. Отже, дослідження технології V2X, зокрема її моделювання та імітаційне моделювання, є своєчасним та актуальним.

Об'єктом даної роботи є технологія зв'язку 5G V2X. Зокрема, вона зосереджена на моделюванні V2X-зв'язку з використанням мереж 5G - нової парадигми, яка, як очікується, забезпечить суттєве покращення пропускної здатності, затримки та надійності порівняно з попередніми поколіннями.

Предметом роботи є моделювання та практичне застосування моделей 5G V2X. Створюючи точні симуляції, ми можемо прогнозувати продуктивність систем V2X в різних сценаріях, розуміти потенційні виклики та розробляти рішення для їх пом'якшення. Крім того, буде представлено

практичне застосування цих моделей, яке продемонструє, як їх можна використовувати для вирішення реальних проблем.

РОЗДІЛ І: Зона покриття V2X.

Вступ

У швидко мінливому ландшафті 21-го століття перетин автомобільних технологій і систем зв'язку стоїть на передовій інновацій. Цей зв'язок, відомий як зв'язок Vehicle-to-Everything (V2X), обіцяє революціонізувати не тільки наш підхід до транспорту, але й те, як ми сприймаємо безпеку, ефективність та зв'язок на наших дорогах. У першому розділі цієї дипломної роботи досліджуються ключові компоненти, які сприяють створенню системи зв'язку V2X в контексті управління даними про дорожній рух, мережевої архітектури, безпеки руху та застосування технології 5G.

Ціль дослідження: дослідження роботи технології V2X.

Актуальність: з огляду на стрімкий технологічний прогрес і потенційний вплив на суспільну безпеку, екологічну стійкість та ефективність транспорту, це дослідження є важливим кроком до розуміння комплексної зміни автомобільних парадигм.

1.1 Інтелектуальні транспортні системи (ITS) та їх значення в сучасному суспільстві

Технології інтелектуальних транспортних систем (ITS; intelligent transportation systems) представляють собою інноваційну конвергенцію передових бездротових, електронних і автоматизованих технологій. Ці системи прагнуть безперешкодно інтегрувати різні типи транспортних засобів - від транзитних і вантажних автомобілів до особистих транспортних засобів - з користувачами і транспортною інфраструктурою, такою як дороги і транзитні системи.

Автоматизовані та бортові технології включають точне стикування автобусів, автоматизовані напрямні та системи уникнення зіткнень. Багато технологій ITS можуть допомогти оптимізувати поїздки (прокладання маршрутів), зменшити непотрібні кілометри, збільшити використання інших видів транспорту, скоротити час, проведений у заторах, зменшити залежність від іноземної нафти та покращити якість повітря. Коли ці технології вміло застосовуються в управлінні системами і проектуванні транспортних засобів, вони відкривають шлях до значного скорочення споживання палива. Цього можна досягти за рахунок:

- Оптимізації планування маршруту і мінімізації часу в дорозі

- Забезпечення більш плавного прискорення і уповільнення, тим самим зменшуючи кількість зупинок і переїздів

- зменшення заторів на дорогах

- Впровадження стратегічного ціноутворення та управління попитом

- Підвищення привабливості громадського транспорту

- Адаптація трансмісії транспортного засобу до різних дорожніх умов та рельєфу місцевості

- полегшення руху невеликих взводів близько розташованих транспортних засобів (тобто, більш безпечні транспортні засоби можуть дозволити зменшити вагу без шкоди для безпеки пасажирів) [3].

Важливим елементом ITS є зв'язок, який в основному поділяється на дві категорії: V2X (транспортний засіб до всього) і V2N (транспортний засіб до мережі). V2X забезпечує прямий зв'язок між транспортними засобами та інфраструктурними системами. На відміну від цього, V2N передбачає передачу сигналів через телекомунікаційні мережі, причому технологія 5G відіграє ключову роль у цьому процесі. Зв'язок V2X має вирішальне значення для уникнення зіткнень і автоматизованого водіння, тоді як V2N перевершує його в передачі значних обсягів даних на великі відстані[4].

Виділений зв'язок малого радіусу дії (DSRC; dedicated short range communication) - це технологія V2X, спеціально розроблена для додатків з безпеки на транспорті. Синергетичне поєднання V2X і V2N, яке часто називають гібридним зв'язком, має значний потенціал для оптимізації зв'язку в ITC. Цей гібридний підхід вже впроваджується в транспортних засобах та інфраструктурних системах по всьому світу. Нещодавно з'явилися пропозиції використовувати передові технології стільникового зв'язку, включаючи 5G, для прямого зв'язку V2X, тим самим ще більше підвищуючи ефективність і безпеку транспортних систем.

1.2 Дані про дорожній рух і V2X

DSRC - це добре відпрацьована технологія що характеризується своєю стійкістю і перевіреною надійністю, часто викликає дискусії через свою довговічність. Проте дуже важливо розуміти, що стабільність, яка є ключовим атрибутом DSRC, не є ознакою застарілості. Скоріше, це гарантія її потенційної актуальності в майбутньому, враховуючи її ключову роль в транспортних засобах і транспортних системах.

Почнемо з того, що LTE - це технологія передачі голосу і даних. Слідом за 3G і 4G ми починаємо зустрічати 5G, який має високу пропускну здатність, надвисокі швидкості, значні можливості підключення і низьку затримку. Однак для цього потрібна спеціальна інфраструктура, розбудова якої потребуватиме величезних витрат і часу. Справа в тому, що стільникові мережі використовують базові станції як пристрої трансляції між поколіннями, а прямий V2X такого пристрою трансляції не має. Таким чином, місця з підключенням 5G обмежені на ранній стадії розгортання [5].

Різні рекламні оголошення ставлять під сумнів можливість використання технології стільникового зв'язку для V2X або викликають плутанину. Впровадження технології для мобільних телефонів, які

замінюються в середньому кожні два-три роки, відрізняється від прийнятої технології для життєвого циклу транспортних засобів, який становить приблизно 12-15 років у США, Європі та Японії. Усталена, надійна і перевірена технологія необхідна для автомобілів, де безпека має першорядне значення.

У 14-й версії 3GPP визначена технологія прямого зв'язку V2X, що базується на технології LTE. Однак, оскільки американський стандарт SAE J3161/1, який визначає мінімальні вимоги до передачі базових повідомлень безпеки між транспортними засобами, ще не визначений. Реліз 14 технології LTE V2X не був повністю протестований в різних сценаріях і аспектах, таких як масштабованість в перевантажених сценаріях для зв'язку безпеки V2X. Реліз 16 визначає нову технологію радіодоступу для прямого V2X на основі технології 5G, так зване нове радіо. Розробка стандарту версії 16 була завершена в червні 2020 року. Випуск 17 удосконалить технології 5G і RV2X. 5G New Radio (NR) V2X потребуватиме подальших стандартів для аналізу, оцінки та безпеки транспортних засобів, перш ніж його можна буде використовувати на дорозі[5].

Для безпечного водіння, які технічні вимоги потрібні для зв'язку V2X? Одна з труднощів з V2X полягає в тому, що ви не знаєте, якому автомобілю потрібно буде надсилати конкретну інформацію, і хто повинен буде отримувати цю інформацію. З цієї причини необхідно впровадити технологію, яка дозволить кожному автомобілю передавати своє місцезнаходження і швидкість через регулярні проміжки часу, а також дозволить автомобілям, що знаходяться поруч, надійно отримувати цю інформацію.

Розглянемо, чому DSRC є більш надійною технологією, ніж LTE V2X. DSRC використовує технологію CSMA/CA. Це рандомізований метод, при якому транспортний засіб спочатку перевіряє, чи не передають інші транспортні засоби, перш ніж передавати дані. За допомогою цього

механізму, як правило, лише один автомобіль займає канал одночасно. Таким чином, всі навколишні транспортні засоби мають можливість приймати передачу. Поки кожен автомобіль періодично здійснює ці випадкові передачі, зв'язок досягається. DSRC - це зріла технологія, яка відповідає вимогам V2X. З іншого боку, в LTE V2X PC-5 часові ресурси розділені на 1 мілісекундні підкадри. Частотні ресурси розділені на 10 підканалів, як показано на рис 1.1. Ресурсна одиниця, що використовується транспортними засобами для зв'язку. Кожна машина перевіряє стан використання ресурсу за останню секунду, а потім випадковим чином вибирає один з ресурсів, який знаходиться в нижній 20-ці за енергоспоживанням для зв'язку. Обраний кожною машиною ресурс використовується кожні 100 мілісекунд. Після того, як ресурс обрано, передавач не перевіряє, чи передає хтось інший, перш ніж почати передачу. Принцип запобігання зіткненням на основі DSRC проілюстрована на рис. 1.1. Використання ресурсу відоме як напівпостійне планування. LTE V2X відрізняється від DSRC тим, що дозволяє одночасний зв'язок між декількома автомобілями з використанням різних підканалів, кожен з яких багаторазово використовує свій обраний ресурс[6]. Через ці відмінності це може призвести до кількох проблем.

Першою проблемою є постійна проблема зіткнення пакетів. Якщо кілька машин вибирають один і той же ресурс, комунікації, що використовують цей ресурс, будуть неодноразово стикатися.

Другою проблемою є постійна проблема напівдуплексного зв'язку. Коли кілька машин передають одночасно, навіть якщо використовуються різні підканали, вони зможуть чути один одного під час передачі. Чим більше підканалів встановлено, тим більша ймовірність того, що ця проблема виникне.

Третя проблема - це постійна проблема ближнього і дальнього радіусу дії. У ситуації, зазначеній вище, відповідно до позиційного співвідношення приймаючого автомобіля, сила сигналу від автомобіля, що знаходиться

поруч, буде занадто сильною, а це означає, що існує ймовірність того, що він не зможе почути передачу від автомобіля, який знаходиться далі.

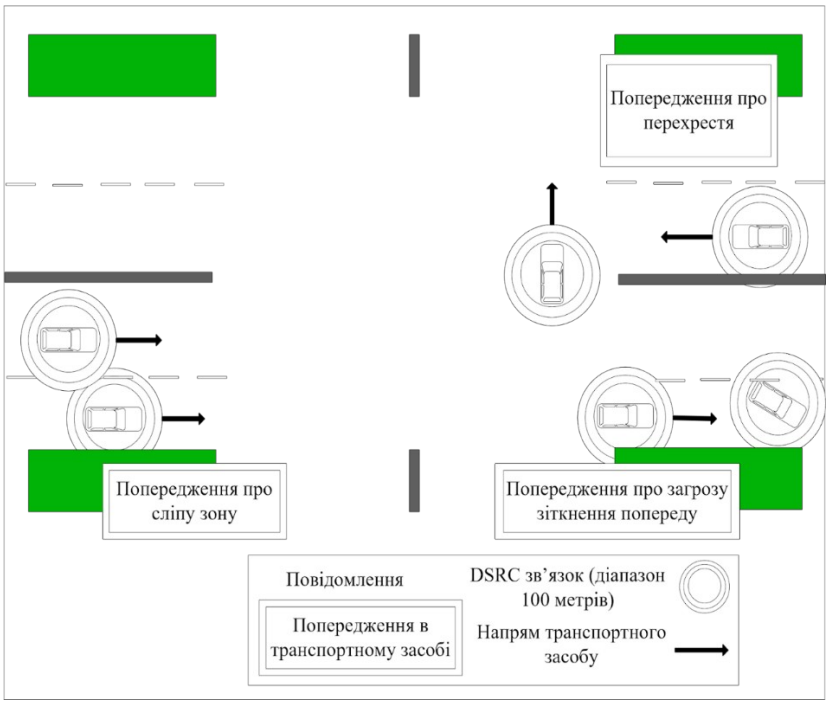


Рис 1.1: Транспортні засоби, що надсилають повідомлення про безпеку, а також відображають попередження водієві

Через ці проблеми LTE V2X може бути ненадійним через постійну втрату пакетів. Для боротьби з цим пакети зазвичай дублюються і відправляються двічі. В результаті LTE V2X вимагає вдвічі більшої пропускної здатності, ніж DSRC або NGV. Тож коли трафік перевантажений, існує ризик, що це ще більше загострить проблему. Для того, щоб LTE V2X PC-5 став зрілою технологією зв'язку V2X, їй потрібно буде довести, що зв'язок у складних умовах може бути встановлений.

1.3 Архітектура мережі

Мережева архітектура V2X в основному використовує інтерфейс PC5. Цей інтерфейс забезпечує зв'язок "один до багатьох" і полегшує

використання LTE-Uu, радіоінтерфейсу між пристроями користувачів і eNodeB [9], [10]. Передача може відбуватися в одноадресному режимі або у форматі мультимедійного мовлення/багатоадресного передавання (MBMS). UE може незалежно використовувати ці два режими роботи для передачі та прийому даних. Крім того, повідомлення V2X можуть надсилатися через лінію зв'язку LTE-Uu і прийматися пристроями користувачів. Існують два режими роботи через PC5 і LTE-Uu, які детально описані нижче:

- *Зв'язок на основі PC5*: Цей режим передбачає передачу повідомлень V2X через PC5, а пристрої користувачів отримують повідомлення через PC5 і MBMS. У цьому процесі дорожній стаціонарний пристрій служить в якості пристрою користувача для отримання повідомлень V2X, які він потім надсилає на сервер додатків V2X. Оброблені повідомлення від сервера додатків V2X потім поширюються серед пристроїв різних користувачів через MBMS. Цей режим дозволяє стільниковій мережі передавати інформацію на ширший діапазон і є важливим для полегшення роботи передових додатків допомоги при керуванні автомобілем.
- *Гібридний зв'язок V2X на основі LTE-Uu і PC5*: цей режим підтримує зв'язок між RSU через PC5 як для передачі, так і для прийому повідомлень V2X. Сервери додатків V2X можуть зв'язуватися з RSU через стільникову мережу. Наприклад, LTE-Uu можна використовувати для передачі V2X-повідомлень за межами прямого діапазону зв'язку PC5. Завдяки такому гібридному використанню LTE-Uu і V2X-зв'язку на основі PC5, трансляція MBMS низхідної лінії передачі даних може бути незначною [9].

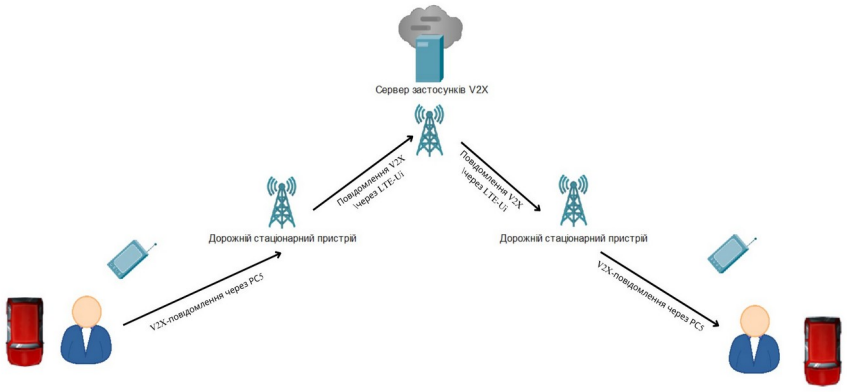


Рис. 1.2: Передача та прийом V2X-повідомлень по LTE-Uu з використанням PC5

1.4 Безпека та зв'язок (безпека та транспортні засоби)

Часто ведуться дискусії про продуктивність зв'язку, такі як частота помилок в пакетах і дальність зв'язку, але є й інші більш важливі показники. Можна написати чотири ключові слова які є важливими для основних повідомлень про безпеку V2X. Ключовими словами є

- Кожен.
- Всюди
- Різні покоління.
- Масштабованість.

Базова безпека V2X вимагає надійного зв'язку з усіма. Якщо деякі транспортні засоби або інфраструктури не можуть зв'язатися один з одним, послуги базової безпеки V2X не будуть працювати. Для того, щоб забезпечити зв'язок з усіма транспортними засобами та інфраструктурами, ключовим фактором для базової безпеки V2X є фундаментальна технологія.

Базова безпека V2X вимагає надійного зв'язку всюди. Транспортні засоби подорожують в різні місця за межами міських і житлових районів, де розгорнуто не так багато інфраструктур. Наприклад, в горах або через тунелі.

Базова безпека V2X вимагає надійного зв'язку з різними поколіннями. Середній термін експлуатації транспортного засобу становить від 12 до 15 років [4]. Тож якщо нові технології будуть замінюватися одна за одною, нові транспортні засоби та старі транспортні засоби не зможуть спілкуватися між собою і послуги базової безпеки V2X не працюватимуть. Для того, щоб забезпечити зв'язок між старими та новими транспортними засобами, для технології зв'язку V2X необхідна зворотна сумісність.

Крім того, базова безпека V2X вимагає надійного зв'язку з можливістю масштабування, щоб мати можливість надійно зв'язуватися навіть в умовах перевантаженості. Зв'язок не повинен втрачатися при занадто великому трафіку. Зв'язок для кожного транспортного засобу повинен бути надійним, навіть в умовах перевантаженості.

Найголовніше, технологія зв'язку повинна бути зрілою для забезпечення базової безпеки V2X. Зрілість означає, що технологія була перевірена протягом багатьох років, протестована в дорожньому русі та інших різних сценаріях. Комунікації ITS мають дві основні категорії: прямий зв'язок для транспортних засобів та інфраструктурних систем, які називаються V2X і V2N, де сигнали передаються через телекомунікаційні мережі. V2N використовується для з'єднання транспортних засобів і застосовується вже багато років. Прямий зв'язок V2X підходить для уникнення зіткнень та автоматизованого водіння. А мережевий V2N добре підходить для передачі великих обсягів даних на великі відстані[6].

1.5 Розподіл ресурсів

Розподіл ресурсів є важливим аспектом будь-якого середовища стільникової мережі. Він відіграє важливу роль у підтримці дружнього доступу для кінцевих користувачів, ділових партнерів і клієнтів додатків на

базі стільникового зв'язку. Розподіл ресурсів має великі переваги для середовища стільникових мереж.

Розподіл ресурсів у мережі 5G NR передбачає виділення користувачам частотно-часових ресурсів - завдання, на яке впливають численні фактори, такі як вимоги користувача до швидкості передачі даних, стан каналу і доступні ресурси. Ці фактори відіграють вирішальну роль у забезпеченні ефективної підтримки зв'язку V2X за технологією 5G NR.

Існує два основних підходи до розподілу ресурсів в мережах 5G NR:

- *Централізований розподіл ресурсів*: У цьому сценарії базова станція (БС) централізовано розподіляє ресурси між користувачами. Хоча цей метод легко реалізувати, він може бути неефективним, оскільки не враховує специфічні умови каналу для кожного користувача.
- *Розподілений розподіл ресурсів*: У цьому підході, навпаки, користувачі самі керують розподілом ресурсів. Хоча цей метод складніший у виконанні, він може підвищити ефективність за рахунок врахування унікальних умов каналу для кожного користувача.

У багатьох практичних сценаріях використовується гібридний підхід, коли базова станція централізовано розподіляє ресурси в одних випадках, а користувачі керують власними ресурсами в інших. Цей метод пропонує баланс між простотою централізованого підходу та ефективністю розподіленого методу.

Розглянемо докладніше фактори, що враховуються при розподілі ресурсів в 5G NR:

- *Вимоги користувача до швидкості передачі даних*: Це стосується обсягу даних, які користувачеві необхідно відправити або отримати протягом певного періоду. Щоб гарантувати надійний зв'язок V2X, базова станція повинна забезпечити кожному користувачеві достатній обсяг ресурсів для виконання цих вимог.

- *Умови каналу*: Ці умови відрізняються у різних користувачів через відмінності в розташуванні користувачів та фізичному середовищі. Базова станція повинна враховувати ці умови при розподілі ресурсів для оптимізації зв'язку V2X.
- *Доступні ресурси*: Ресурси в будь-якій системі обмежені. Тому базова станція повинна ефективно розподіляти ресурси, щоб всі користувачі могли передавати і отримувати необхідні їм дані.

5G (NR) має структуру протоколу, яку можна розділити на дві окремі частини: стек протоколів площини користувача і стек протоколів площини управління. Роль стеку протоколів користувацького рівня полягає в транспортуванні даних користувача, в той час як архітектура керуючого рівня відповідає за встановлення з'єднань, забезпечує мобільність і комплексну безпеку.

Стек протоколів для користувацького рівня в 5G NR зображено на Рис 1.3, де показано, що передача між gNB і UE на інтерфейсі фізичного рівня (повітряний інтерфейс) відбувається у вигляді радіокадрів.

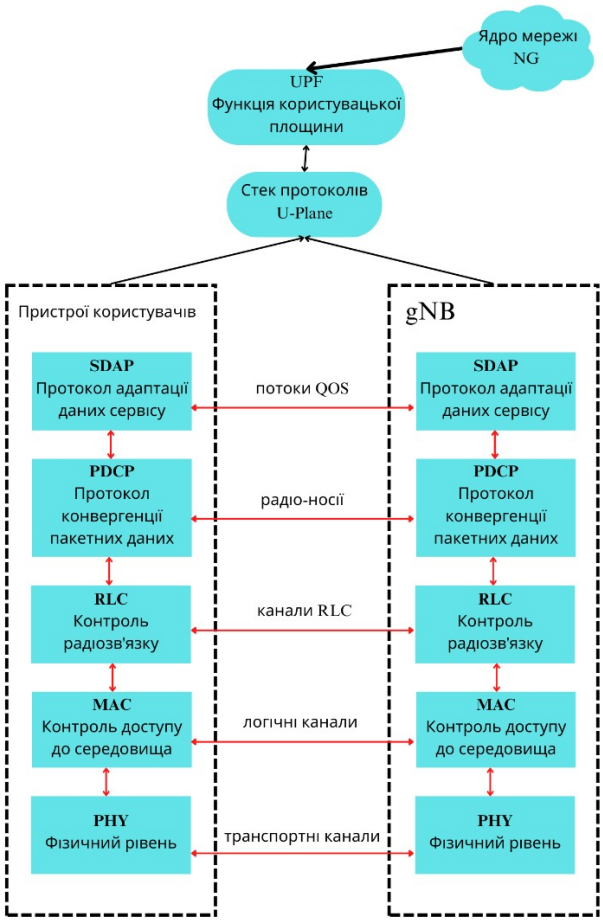


Рисунок 1.3: Архітектура стеку протоколів користувацького рівня 5G NR [7]

З точки зору часової області, структура кадру 5G NR складається з радіопередач, які класифікуються на радіокадри, підкадри, слоти і міні-слоти Рис. 1.4. Радіокадр триває 10 мс і включає 10 підкадрів, кожен з яких триває 1 мс. Кожен підкадр може складатися з одного або декількох суміжних слотів, а кожен слот складається з 14 символів OFDM (ортогонального мультиплексування з частотним розділенням каналів). Потенційна можливість передачі через частину слоту називається міні-слотом. Як показано на Рис.1.4, структура радіокадру підтримує різні нумерології передачі ($\mu = 0, 1, 2, 3, 4$) відповідно до Табл. 1.1.

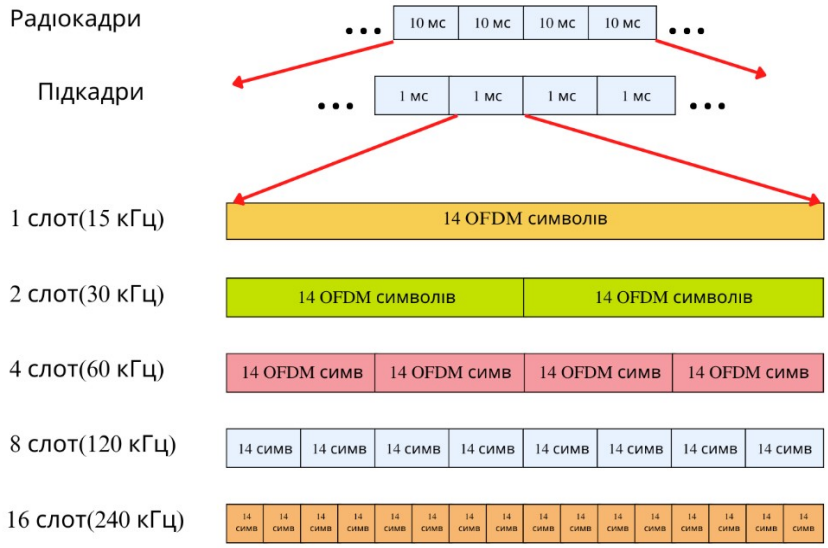


Рис. 1.4: Структура кадру 5G NR [7]

Оскільки тривалість OFDM-сигналу обернено пропорційна відстані між його піднесучими, тривалість слоту залежить від обраної нумерації (Таблиця 1). З точки зору частотної області, OFDM-символ включає 12 піднесучих, і кожна піднесуча може бути розташована відповідно до масштабованої нумерології, детально описаної в Табл 1.1.

Таблиця 1.1 5G NR - підтримувані нумерології передачі [8]

μ	0	1	2	3	4
Ширина субносія $\Delta f = 2^{\mu}$x15 кГц	15 кГц	30 кГц	60 кГц	120 кГц	240 кГц
Ширина смуги блоку ресурсів Δf μ x 12 субносіїв	180 кГц	360 кГц	720 кГц	1.44 МГц	2.88 МГц
Тривалість кадру	10 мс	10 мс	10 мс	10 мс	10 мс
Слотів за кадр	10	20	40	80	160
Тривалість слоту	1 мс	500 мкс	250 мкс	125 мкс	62.5 мкс
OFDM символів на слот	14	14	14	14	14
Тривалість символу	71.43 мкс	35.71 мкс	17.86 мкс	8.93 мкс	4.46 мкс

Циклічний префікс	Звичайний	Звичайний	Звичайний, розширений	Звичайний	Звичайний

Елемент ресурсу, найменший часово-частотний ресурс на одній піднесучій одного OFDM-символу, позначається як $(k,l)_{p,\mu}$. Тут "k" означає індекс піднесучої в частотній області, "l" - позицію OFDM-символу в часовій області, "p" - порт антени, а "μ" - конфігурацію інтервалів між піднесучими, як визначено в таблиці 1. Ресурсний блок, також відомий як фізичний ресурсний блок (PRB), - це блок з $N^{RB}_{sc} = 12$ піднесучих, на яких планується передача. Фізичний рівень 5G NR використовує частотно-часовий ресурс (фізичний ресурсний блок) для передачі.

Ресурсна сітка складається з $N^{розмір,\mu}_{сітка,x}$ піднесучих і Nпідкадрів,μ _{symb} кількості OFDM-символів (Табл 1.2 і 1.3).

Таблиця 1.2 - Кількість OFDM-символів на слот, підкадрів на слот та OFDM-символів на підкадр для нормального циклічного префікса[8]

μ	0	1	2	3	4
	14	14	14	14	14
	1	2	4	8	16
	14	28	56	112	224

Таблиця 1.3 - Кількість OFDM-символів у слоті, підкадрів у слоті та OFDM-символів у підкадрі для розширеного циклічного префікса[8]

μ			
2	12	4	48

1.6 Висновок

Автомобільна промисловість переживає період значної трансформації, що значною мірою зумовлена технологічним прогресом. Такі ключові сфери, як безпека транспортних засобів, управління даними про дорожній рух, мережева безпека та розподіл ресурсів у рамках нової технології 5G, відіграють вирішальну роль у формуванні майбутнього мобільності.

У вступі підкреслили постійно зростаючу необхідність впровадження передових технологічних досягнень в автомобільну промисловість. Впровадження цих технологій не лише забезпечує підвищення безпеки та ефективності, але й відкриває шлях до нових захоплюючих можливостей, таких як автономне водіння.

Згодом заглибилися у важливість даних про дорожній рух та роль мережевої архітектури V2X в сучасних автомобільних екосистемах. Обговорили, як обмін інформацією між транспортними засобами, інфраструктурою та іншими учасниками дорожнього руху може значно підвищити безпеку, управління дорожнім рухом та ефективність реагування на надзвичайні ситуації.

Крім того, проаналізували роль зв'язку V2X у підвищенні безпеки транспортних засобів. Тут підкреслили необхідність того, щоб зв'язок був узгодженим і надійним між усіма суб'єктами, незалежно від місця розташування, покоління транспортних засобів або рівня перевантаженості мережі. Крім того, підкреслили необхідність зворотної сумісності і масштабованості технології зв'язку V2X для забезпечення її довгострокової життєздатності та ефективності.

Нарешті, розглянули розподіл ресурсів у мережі 5G. Ця технологія зв'язку наступного покоління, що здатна революціонізувати V2X-зв'язок, вимагає ефективних стратегій розподілу ресурсів. Обговорили нюанси розподілу ресурсів в рамках NR-5G, підкресливши необхідність збалансованого підходу, який враховує вимоги користувачів до швидкості передачі даних, стан каналу і обмежену доступність ресурсів.

РОЗДІЛ ІІ. Моделювання дорожнього руху в розумних транспортних системах

Вступ

Розробка інтелектуальної транспортної системи (ITS) є складним процесом у зв'язку з великою кількістю взаємозв'язаних компонентів та розподіленою структурою системи. Така складність часто призводить до нерозв'язаних проблем та неясностей під час аналізу. Симуляція може служити надійним інструментом для віртуального моделювання та перевірки, забезпечуючи більшу впевненість в процесі проектування та реалізації. Симулятори відіграють критичну роль в таких інженерних областях, як динаміка транспортного засобу і мережеві комунікації, які є ключовими елементами ITS. Тому в наступному розділі ми розглянемо інструменти для моделювання роботи V2X.

2.1 Загальний огляд моделювання технології V2X

Протягом тривалого періоду, безліч дослідницьких ініціатив надали відповіді на ключові технологічні потреби у сфері зв'язку для транспортних засобів. Сьогодні на ринку доступні вартісно ефективні системи позиціювання та радіоелектронне обладнання, процес їх стандартизації активно триває. Завдання з автомобільного зв'язку переходять від наукового дослідження до етапу готовності до впровадження, причому основна увага приділяється використанню уже існуючого обладнання, яке доступне в широкому масштабі.

Польові експлуатаційні випробування – це реальні тестові програми великого масштабу, які мають за мету докладну оцінку ефективності, якості, стабільності, та придатності технологій, розроблених з метою створення

більш інтелектуальних, безпечних, екологічно чистих та комфортних транспортних систем[9].

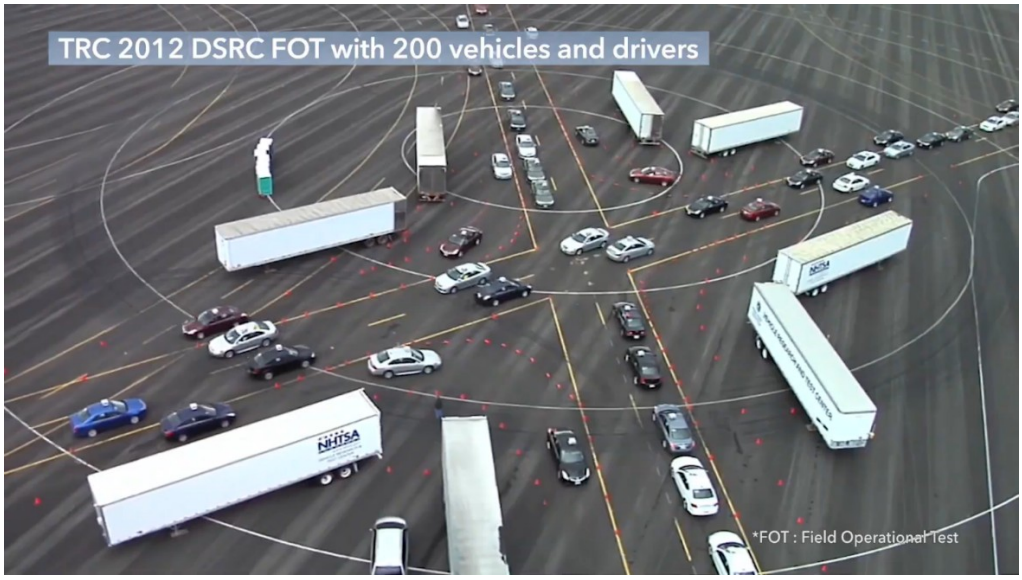


Рис. 2.1: Польові експлуатаційні випробування[10]

Через високу вартість експлуатаційних тестових площадок, комп'ютерне моделювання стає єдиною практичною альтернативою для аналізу продуктивності. Такі симуляції можуть враховувати і непрямі ефекти, наприклад, перевантаження мережі у годину пік. Моделювання є важливим елементом у процесі польових експлуатаційних випробувань і має проводитися в ході польових експлуатаційних випробувань[9].

На даний момент для моделювання автомобільного зв'язку використовуються різні типи симуляторів. Реалістичне моделювання V2X зв'язку включає в себе мінімум два різних симулятори: симуляція дорожнього руху та симуляція мережі.

2.1.1. Симуляція дорожнього руху

Спеціалісти у сфері транспорту часто звертаються до транспортних симуляторів для відтворення різних аспектів транспортних потоків та дослідження їх впливу на стабільність руху та інші фактори. Вони також

використовуються як джерела інформації про модель мобільності для мережевих симуляторів, коли йдеться про моделювання V2X.

Модель мобільності – це математичне представлення або симуляція, яка описує схему руху мобільних користувачів і те, як їхнє місцезнаходження, швидкість і прискорення змінюються з часом. Модель мобільності описується структурою, яка включає топологічні карти, такі як смуги руху, дороги, вулиці, перешкоди в моделі мобільності та комунікації, швидкості автомобілів, точки тяжіння і відштовхування, засновані на щільності руху, пов'язані з тим, як може змінюватися час моделювання, розподіл транспортних засобів на дорогах і інтелектуальна модель водіння.

Відповідно до різних рівнів деталізації інформації, необхідної для моделювання, моделі можна умовно розділити на дві категорії: макроскопічні моделі та мікроскопічні моделі[11].

Макроскопічна модель базується на розподілі транспортного потоку та початкових транспортних засобів, що відображає загальні характеристики транспортних засобів на дорожній мережі. До загальновживаних параметрів відносяться транспортний потік, середня швидкість, щільність розподілу транспортних засобів тощо.

Мікроскопічна модель використовується для моделювання кожного транспортного засобу як відносно незалежного об'єкта дорожньої мережі, який може відображати взаємозв'язок руху та взаємний вплив між транспортними засобами. Мікроскопічні моделі трафіку розробляються з акцентом на окремі транспортні засоби, що беруть участь у дорожньому русі. Цей підхід допомагає аналізувати вплив на систему транспорту за умови моделювання різних ступенів взаємодії транспортних засобів у контексті загальної кількості автомобілів.

Генератори моделей мобільності транспортних засобів необхідні для підвищення рівня реалістичності симуляцій V2X. Вони генерують реалістичні траси руху транспортних засобів, які використовуються як вхідні

дані для мережевого симулятора. На виході траси детально описується місцезнаходження кожного транспортного засобу в кожний момент часу протягом всього часу моделювання і їх профілі мобільності.

До найбільш поширених транспортних симуляторів, що використовуються в V2X моделюванні, відносяться:

- SUMO (Simulation of Urban MObility) - це популярне програмне забезпечення для мікроскопічного моделювання дорожнього руху, яке включає в себе можливості V2X. Воно дозволяє оцінити вплив технологій V2X на транспортний потік, безпеку та ефективність, і може бути інтегроване з такими інструментами, як Veins для моделювання комунікацій V2X.
- VanetMobiSim - це універсальний симулятор автомобільної спеціальної мережі (VANET), призначений для моделювання зв'язку та мобільності V2X. Він дозволяє дослідникам аналізувати продуктивність протоколів V2X та алгоритмів маршрутизації в реалістичних транспортних сценаріях.
- MOVE - це середовище моделювання, орієнтоване на кооперативні транспортні системи, включаючи V2X-зв'язок. Він пропонує підтримку реалістичних моделей мобільності, моделювання каналів і сценаріїв руху, що дозволяє дослідникам оцінювати продуктивність V2X в різних середовищах.
- STRAW (Simulation Tools for Realistic Analysis of VANETs) - це спеціалізований інструмент моделювання для VANET, включаючи зв'язок V2X. Він забезпечує реалістичне середовище для вивчення впливу різних протоколів зв'язку і сценаріїв трафіку на продуктивність V2X.
- FreeSim - це симулятор для моделювання трафіку з відкритим вихідним кодом, який підтримує можливості V2X. Він пропонує гнучкість у моделюванні протоколів зв'язку V2X, моделей мобільності та сценаріїв

трафіку, що дозволяє дослідникам моделювати великомасштабні мережі V2X[12].

Таблиця 2.1 - Порівняння параметрів різних симуляторів дорожнього руху[

Параметр	VanetMobiSim	SUMO	MOVE	STRAW	FreeSim
Ліцензія	Вільна	Вільна	Вільна	Вільна	Вільна
Графічний інтерфейс	Має	Має	Має	Має	Має
Мапи реального світу	Має	Має	Має	Має	Має
Користувацькі карти	Має	Має	Має	-	Не має
Макроскопічне моделювання	Не має	Не має	Не має	Не має	Має
Мікроскопічне моделювання	Має	Має	Має	Має	Має
Вивід	Ns-3, GloMoSim, QualNet, XML	XML, TraCI interface	Ns-3, GloMoSim, QualNet, XML	SWANS	-

2.1.2. Симуляція мережевого зв’язку

Мережевий симулятор - це програмне забезпечення, яке прогнозує поведінку комп'ютерних мереж. оскільки комунікаційні мережі стали занадто складними для традиційних аналітичних методів, щоб забезпечити точне розуміння поведінки системи, використовуються мережеві симулятори. більшість симуляторів використовують симуляцію дискретних подій - моделювання системи, в якій змінні стану змінюються в дискретні моменти часу. Симуляція мережевого зв'язку використовується для аналізу параметрів мережі шляхом побудови математичних моделей та імітації реальних умов

дорожньої мережі і поведінки передачі даних. Дослідники можуть контролювати параметри симуляції та адаптувати фреймворк відповідно до конкретних дослідницьких потреб, що допомагає виявити системні проблеми та модифікувати алгоритми і функції. На якість бездротового зв'язку між транспортними засобами впливають різні фактори, такі як будівельні перешкоди, сусідні транспортні засоби та багатопроменеві ефекти. Моделювання каналу бездротового зв'язку між транспортними засобами та дорогою має важливе значення[14,15].

Ось огляд деяких популярних мережевих симуляторів, що використовуються для V2X та ITS:

- ns-3 (Network Simulator 3) - це широко використовуваний дискретно-подієвий інструмент для моделювання мереж. Він підтримує дротові та бездротові мережі, пропонуючи широкий спектр компонентів і протоколів для моделювання.
- OMNET++ - це мережевий симулятор на основі подій, який забезпечує загальну підтримку різних мережевих систем. Він має ядро на основі C++ та додаткові фреймворки для спеціалізованих областей застосування. OMNET++ пропонує власну мову сценаріїв для опису топології, підтримує розподілене та паралельне виконання, а також надає інструменти аналізу продуктивності для візуальної оцінки.
- SWANS (Scalable Wireless Ad hoc Network Simulator) - це симулятор бездротових і сенсорних мереж. Він використовує високопродуктивний механізм моделювання JiST і робить акцент на масштабованості. SWANS використовує компонентну архітектуру і дозволяє виконувати мережеві додатки на Java через змодельовані мережі, що робить його придатним для великомасштабних сценаріїв.
- NetSim - це комерційний симулятор, який широко використовується для проектування та оцінки комп'ютерних та комунікаційних мереж. Він пропонує зручні інструменти для моделювання та емуляції,

забезпечуючи анімацію пакетів і підтримку вбудованого аналізу. NetSim був прийнятий в різних професійних середовищах, включаючи військові організації, комунальні служби та виробників мережевого обладнання.

- QualNet - це комерційний мережевий симулятор, який дозволяє користувачам тестувати і планувати великі реалістичні мережі за допомогою симуляції. Він пропонує операційно точні контексти, широкі можливості конфігурації, а також інструменти для відстеження та аналізу подій трафіку. QualNet часто використовується для проектування, розробки та навчання мереж.

Таблиця 2.2 - Порівняння параметрів різних симуляторів мережі

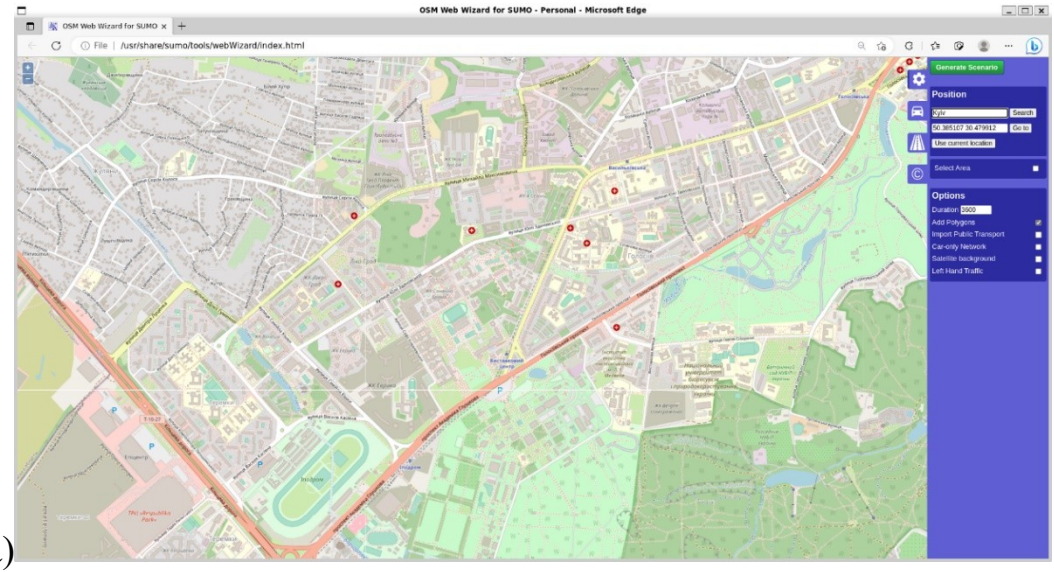
Параметр	ns-3	OMNET++	SWANS	NetSim	QualNet
Ліцензія	Вільна	Вільна для навчання	Вільна	Комерційна	Комерційна
Графічний інтерфейс	Обмежений	Має	Має	Має	Має
Інтерфейс	C++/ Python	C++/NED	Java	C/Java/ .NET	Parsec (C Based)
Паралелізм	Не має	MPI/PVM	Не має	-	SMP/ Beowulf
Операційна система	Linux, FreeBSD, Mac OS X, Windows	Linux, Mac OS X, Windows	Linux, Mac OS X, Windows	Windows	Linux, Mac OS X, Windows, Unix
Підтримка мобільності	Має	Має	Має	Має	Має

Для подальшої роботи для симуляції дорожнього руху будемо використовувати SUMO, а для симуляції мережі зв’язку – ns-3. Розглянемо обидві програми більш детально.

2.2. Огляд програми SUMO та принцип роботи

"Симуляція міської мобільності" (SUMO) - це комплексний пакет для моделювання дорожнього руху, який розробляється Інститутом транспортних систем Німецького аерокосмічного центру. Цей проект був започаткований у 2001 році, а його перший реліз, як платформи з відкритим вихідним кодом, відбувся у 2002 році[17].

Інструментарій SUMO виходить за рамки простого моделювання дорожнього руху, надаючи додаткові утиліти для створення та імпорту структур дорожньої мережі та управління генерацією, перетворенням та імпортом попиту на транспортні послуги. Завдяки своїм універсальним можливостям моделювання, SUMO вміло відтворює різні аспекти дорожнього руху, включаючи системи громадського транспорту, різні типи світлофорів і динаміку багатосмугового рух[16].

а) 

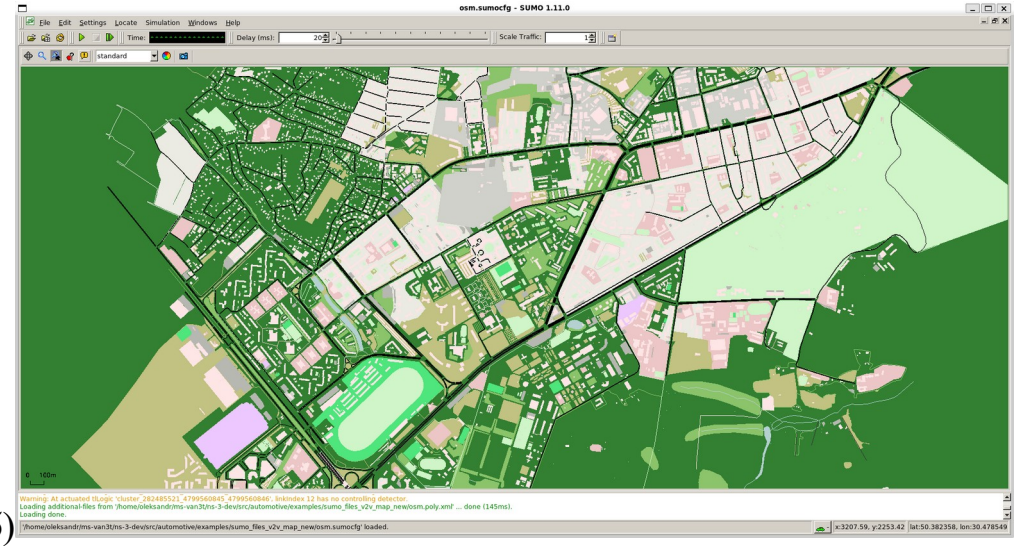
б) 

Рис. 2.2: Приклад перетворення OpenStreetMap; а) оригінал OpenStreetMap, б) мережа, імпортована в SUMO

SUMO - це більше, ніж простий симулятор дорожнього руху; це комплексний набір інструментів, які використовуються для налаштування та виконання симуляцій дорожнього руху. Система імітації дорожнього руху "сумо" вимагає представлення попиту на транспортні послуги та дорожньої мережі в унікальному форматі, який повинен бути створений або імпортований з різних ресурсів.

Дорожні мережі для SUMO можуть бути створені за допомогою інструменту, відомого як "netgen", або шляхом імпорту цифрової дорожньої карти. Імпортер дорожніх мереж "netconvert" має можливість імпортувати мережі з інших програм для моделювання дорожнього руху, таких як VISUM, Vissim або MATsim. Він також може працювати з іншими широко використовуваними форматами, такими як шейп-файли або Open Street Map. Підтримка мереж TIGER була припинена через відсутність відповідних додатків. Однак мережі TIGER також доступні у вигляді шейп-файлів і були інтегровані в базу даних OSM. Окрім цих форматів, "netconvert" також може працювати з менш відомими форматами, такими як формат мережі RoboCup або openDRIVE.

У SUMO використовується мікроскопічна модель мобільності і кожен транспортний засіб представлений детально і чітко. Унікальний ідентифікатор, час відправлення та вказаний маршрут по мережі - це мінімальні вимоги до кожного транспортного засобу. За бажанням, опис може бути більш повним. Можна вказати такі характеристики, як характеристики відправлення та прибуття, наприклад, конкретну смугу руху, швидкість або місцезнаходження.

У SUMO визначення транспортних засобів створюються з використанням таких джерел даних, як "матриці відправлення/призначення" (O/D матриці), які особливо корисні для великомасштабних сценаріїв. Додаток "od2trips" в SUMO перетворює ці матриці в окремі поїздки транспортних засобів, призначаючи початкову і кінцеву дорогу і час відправлення.

Маршрути розраховуються за допомогою розподілу трафіку, який застосовує різні функції витрат. Такі програми, як "jtrrouter" та "dfrouter", використовуються для розрахунку маршрутів через мережу на основі відсотків поворотів на перехрестях та даних детектора петель відповідно.

Моделювання в SUMO є дискретним у часі та безперервним у просторі, а положення кожного транспортного засобу визначається його смугою руху та відстанню від початку смуги. Швидкість транспортних засобів обчислюється за допомогою моделі слідування за автомобілем. Існує дві версії симуляції дорожнього руху: версія командного рядка для пакетного моделювання та графічна версія для візуалізації з використанням openGL[18,19].

2.3. Огляд програми ns-3 та принцип роботи

ns-3 - це програмний інструмент для моделювання мереж, що використовується переважно в академічних дослідженнях та освіті. Симулятор розроблений на C++ і пропонує прив'язки для Python, що робить

значну частину інтерфейсу прикладного програмування (API) C++ доступною для Python. Тому користувачі можуть писати симуляції ns-3 будь-якою з цих мов. Такий підхід полегшує налагодження та робить симуляції більш практичними у порівнянні з системами вищого рівня абстракції, такими як ns-2, його попередник[13].

Рис. 2.3: Приклад роботи програми ns-3

Симуляції в ns-3 є більш достовірними, оскільки вони виконують код C++ безпосередньо, без необхідності використання складного рівня інтерпретації. Таке наближення коду симуляції до реальної реалізації робить результати симуляції більш реалістичними. Симулятор працює, обробляючи відсортований список майбутніх подій у порядку зростання часу їх настання.

ns-3 надає декілька типів моделей, що дозволяє користувачам визначати різні компоненти мережі. Для виконання симуляції користувач повинен створити всі необхідні елементи мережі, беручи до уваги такі категорії, як вузли (представляють фізичні мережеві системи, такі як смартфони або комутатори), пристрої (зображують частини мережевого вузла, що беруть участь у мережевому зв'язку), канали (представляють фізичні середовища для передачі даних), протоколи (визначають операції, що

виконуються над кожним мережевим пакетом), заголовки (управління даними, що містяться в заголовках пакетів відповідно до мережевих стандартів) і пакети (використовуються для представлення даних, що передаються через мережу, в тому числі інформації в заголовку і корисного навантаження).

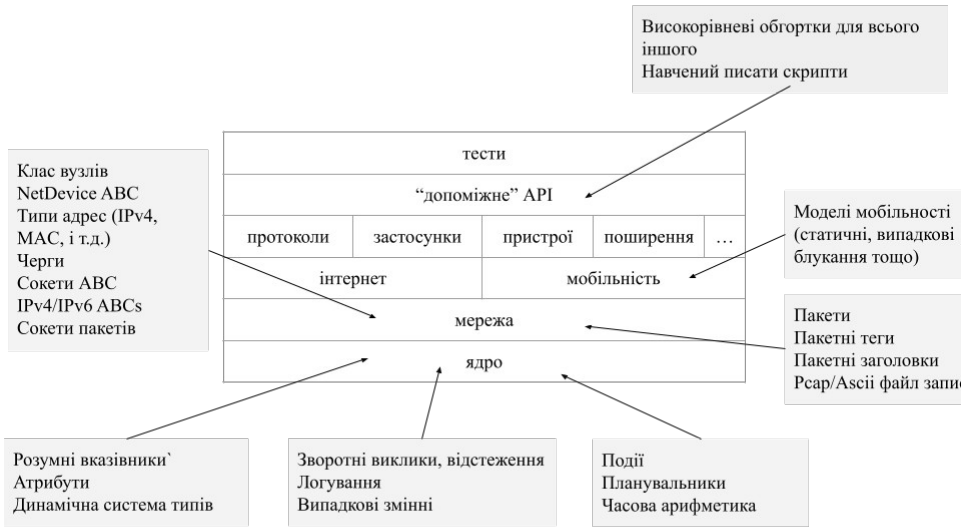


Рис. 2.4: Програмна структура ns-3

На додаток до цих компонентів, ns-3 пропонує інші об'єкти, які допомагають у моделюванні, наприклад, ті, що надають випадкові числа або зберігають додаткову необхідну інформацію. Симулятор функціонує, керуючи відсортованим списком подій. Для цього необхідно створити топологію мережі, яку потрібно змоделювати, використовуючи вищезгадані частини мережі. Для запуску симуляції необхідно оголосити початкові події, що може призвести до створення більшої кількості подій.

Після запуску симуляції симулятор ітераційно переглядає список майбутніх подій, підлаштовуючи час симуляції під час настання події та викликаючи відповідний обробник для її опрацювання. Будь-які події, викликані цим обробником під час симуляції, додаються до списку майбутніх подій симулятора і обробляються в такий самий спосіб. Симуляція

завершується після заданого часу симуляції або коли немає більше подій, що очікують настання[20].

2.4. Висновок

Таким чином, у цьому розділі ми представили всебічне дослідження моделювання V2X. Складна взаємодія між транспортним і мережевим середовищами в V2X вимагає одночасного використання двох симуляторів: симулятора дорожнього руху і симулятора мережі.

Наш огляд симуляторів дорожнього руху виявив їхню важливу роль у моделюванні та прогнозуванні поведінки транспортних засобів і пішоходів у різних дорожніх сценаріях. Ці інструменти дозволяють створювати складні та реалістичні моделі транспортних потоків, що робить їх незамінними для розуміння динаміки V2X.

Аналогічно, наш аналіз мережевих симуляторів підкреслив їхню здатність імітувати різні мережеві умови, конфігурації та протоколи. Такі симулятори надають можливість оцінити, як дані передаються, приймаються і обробляються в мережах V2X, що є важливим аналогом симуляції трафіку.

Для продовження цього дослідження було обрано SUMO та ns-3 як симулятори трафіку та мережі відповідно. SUMO, симулятор дорожнього руху з відкритим вихідним кодом, був обраний завдяки своїм широким можливостям моделювання дорожнього руху, розширюваності та налагодженій інтеграції з мережевими симуляторами. Він має можливість створювати реалістичні сценарії руху, підтримує широкий спектр типів транспортних засобів і видів транспорту, а також може бути масштабований для моделювання як малих, так і великих мереж.

З іншого боку, ns-3 був обраний в якості мережевого симулятора через його детальне і точне моделювання мережевих протоколів, відкритий характер і широку підтримку симуляції сучасних технологій бездротового

зв'язку. Крім того, його сумісність з SUMO була ще однією переконливою причиною для цього вибору, оскільки це полегшило б узгоджене моделювання V2X.

Також було досліджено функціонування програм SUMO та ns-3 , окреслили їхні принципи роботи та потенціал спільного використання для більш інтегрованого та реалістичного моделювання V2X. Це дозволить краще дослідити вплив мобільності транспортних засобів на продуктивність систем зв'язку V2X.

На закінчення, обрана комбінація SUMO для моделювання трафіку і ns-3 для моделювання мереж, враховуючи їх сильні сторони і універсальність, пропонує ідеальну платформу для проведення поглибленого аналізу і оцінки сценаріїв зв'язку V2X. Володіючи цими потужними інструментами, ми добре підготовлені до моделювання та аналізу продуктивності різних протоколів і додатків у сценаріях V2X, що є темою наступного розділу.

РОЗДІЛ ІІІ. Практичне застосування V2X моделювання: Аналіз трафіку та мереж за допомогою SUMO та ns-3

Вступ

У цьому розділі ми переходимо від теорії до практики, використовуючи симулятори SUMO та ns-3 для моделювання V2X-комунікацій у Голосіївському районі Києва.

По-перше, ми використаємо SUMO для створення реалістичного сценарію дорожнього руху на основі реальної карти цього району, щоб відтворити різноманітні умови руху.

Потім ми використаємо ns-3, щоб запустити симуляцію двічі, спочатку з вимкненим, а потім увімкненим режимом UDP. Це забезпечить порівняльне розуміння того, як різні мережеві протоколи впливають на продуктивність V2X.

Нарешті, ми проведемо комплексну симуляцію V2X, поєднуючи сценарій трафіку і мережеві умови, щоб зробити висновки і наслідки для реальних комунікацій V2X. Аналіз цих результатів допоможе спрямувати майбутні вдосконалення систем зв'язку V2X.

3.1. Підготовка оточення для моделювання

Для проведення моделювання було використане таке програмне забезпечення:

- Windows Subsystem for Linux (WSL) - це прошарок сумісності, розроблений Microsoft, який дозволяє запускати двійкові файли. Ця функція дозволяє розробникам та адміністраторам використовувати утиліти та програми Linux безпосередньо у Windows, без необхідності встановлювати окрему операційну систему Linux або

запускати віртуальну машину. В якості дистрибутиву Linux використано Ubuntu 22.04

- "Simulation of Urban MObility" (SUMO) - це комплексний програмний пакет для моделювання транспортних потоків з відкритим вихідним кодом, описаний у попередньому розділі. Використано останню доступну версію – SUMO 1.17.0.
- ns-3 (network simulator 3) - мережевий симулятор дискретних подій з відкритим вихідним кодом. Також використано останню версію – ns-3.38.
- Для аналізу трафіку між транспортними засобами, використаємо програму Wireshark

3.2 Створення карти

Для генерування карти, яка буде використовуватись для подальшої роботи, використаємо програму OSMWebWizard з пакету SUMO. Ми її використовуємо для генерування трафіку в реальному часі на обраному ділянці реальної поверхні.

На імпорт інфраструктури з OSM до симуляції SUMO впливають різні значення пункту Options. Активуємо наступні два:

- *Add Polygon* - генерується симуляція дорожнього руху, але також будуть імпортовані всі типи доріг і рейок (велодоріжки, пішохідні доріжки, залізничні колії і т.д.)
- *Car-only Network* - до мережі будуть включені тільки дороги, на яких дозволено рух легкових автомобілів. Це використано для зменшення розміру мережі та зменшення складності перехресть

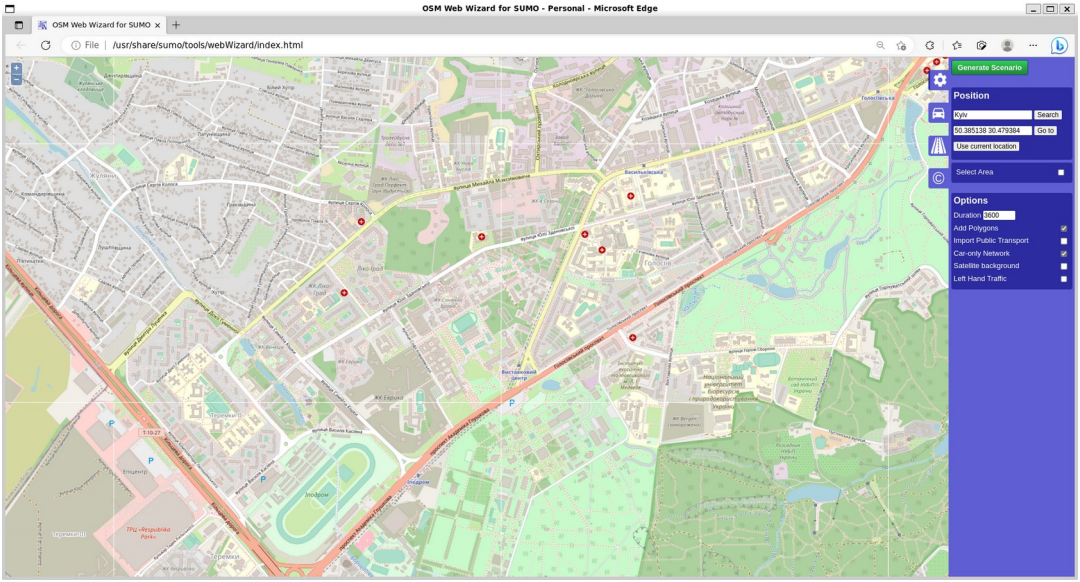


Рис. 3.1: Створення карти за допомогою інструмента OSMWebWizard

SUMO полегшує моделювання різних видів транспорту. В інтерфейсі генерації попиту кожен вид транспорту можна увімкнути або вимкнути, встановивши відповідні прапорці. Веб-майстер OpenStreetMap (OSM) генерує рандомізований попит для кожного виду транспорту на основі розподілу ймовірностей, визначеного двома ключовими параметрами::

- Веб-майстер OSM призначає межі відправлення та прибуття для кожного новоствореного транспортного засобу, використовуючи процес випадкового вибору. **Through Traffic Factor**- це параметр, який вказує на ймовірність вибору ребра на кордоні області моделювання, порівняно з ребрами, що повністю знаходяться в межах області моделювання. Високе значення коефіцієнта наскрізного руху вказує на високу ймовірність того, що транспортні засоби виїжджатимуть і прибуватимуть на межі моделювання, що є репрезентативним для сценарію зі значним наскрізним рухом..
- Параметр **Count** визначає кількість транспортних засобів, що генеруються на кожний смуго-кілометр на годину. Наприклад, якщо мережа складається з 3 ділянок загальною довжиною 5 км, кожна з яких має 2 смуги, сумісні з поточним режимом руху, і значення count

дорівнює 90, загальна кількість транспортних засобів, що генеруються за годину, становитиме 5 * 2 * 90 = 900. Це відповідає параметру randomTrips p=4, що означає, що новий транспортний засіб додається до мережі з інтервалом кожні 4 секунди[21].

Після низки випробувань встановлено, що найкращими параметрами для нашого випадку є значення 10 та 30 відповідно.

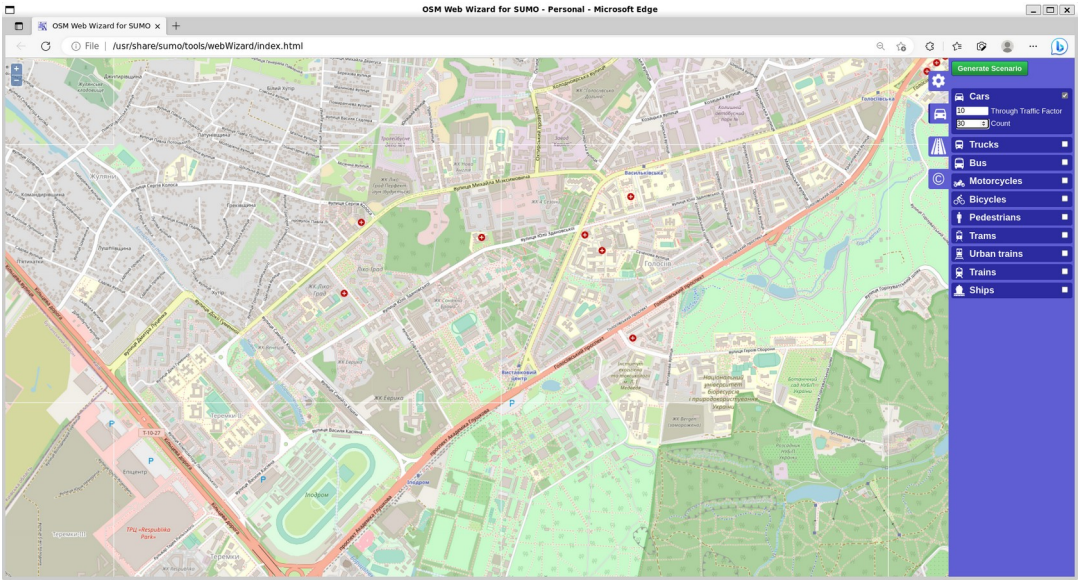


Рис. 3.2: Задання параметрів симулювання сценарію

В результаті ми отримали папку з наступною структурою:

Табл. 3.1. Структура папки зі згенерованим сценарієм

Назва файлу	Призначення
build.bat	Пакетний файл для автоматизації процесу побудови симуляції
osm.net.xml.gz	Основний мережевий файл, який SUMO використовує для визначення транспортної мережі. Він стиснутий у форматі GZIP для ефективності. Він містить інформацію про дорожню мережу, витягнуту з даних OpenStreetMap.
osm.netccfg	Конфігураційний файл для NETCONVERT, інструменту в пакеті SUMO, який створює SUMO-мережі з різноманітних вхідних даних. Файл конфігурації містить параметри для процесу перетворення.
osm.passenger.trips.xml	Цей файл містить визначення пасажирських поїздок у симуляції.

	Кожна поїздка визначається від початкового до кінцевого краю мережі, і алгоритми маршрутизації SUMO визначають конкретний маршрут.
osm.poly.xml.gz	Цей стислий файл містить полігональні об'єкти карти, такі як будівлі та природні об'єкти, витягнуті з даних OpenStreetMap.
osm.polycfg	Це конфігураційний файл для POLYCONVERT, інструменту в пакеті SUMO для перетворення полігональних даних.
osm.sumocfg	основний конфігураційний файл, який SUMO використовує для запуску моделювання. Він визначає мережу, маршрут і додаткові файли, які має використовувати SUMO, а також будь-які параметри моделювання.
osm.view.xml	Цей файл містить налаштування для графічного інтерфейсу SUMO, такі як початковий масштаб і положення, а також які шари потрібно відображати.
osm_bbox.osm.xml.gz	Цей стислий файл містить необроблені дані OpenStreetMap для регіону, який ми моделюємо. Bbox у назві означає "обмежувальну рамку", тобто географічну область, яку охоплюють дані.
run.bat	Подібно до build.bat, це ще один пакетний файл, який, ймовірно, використовується для автоматизації процесу запуску симуляції.

Додатково нам знадобиться отримати число вузлів. Для його обрахунку за допомогою команди sumo -c osm.sumocfg --fcd-output trace.xml. За допомогою цієї команди ми створюємо файл з інформацією про місцезнаходження і швидкість, а також іншу інформацію для кожного автомобіля в мережі на кожному часовому кроці. Вихідні дані поводяться дещо подібно до надточного високочастотного GPS-пристрою для кожного транспортного засобу. Вихідні дані можна обробляти далі за допомогою інструменту TraceExporter для адаптації частоти, швидкості обладнання, точності та формату даних. Наприклад, можна візуалізувати на графіку траєкторії руху транспортних засобів.

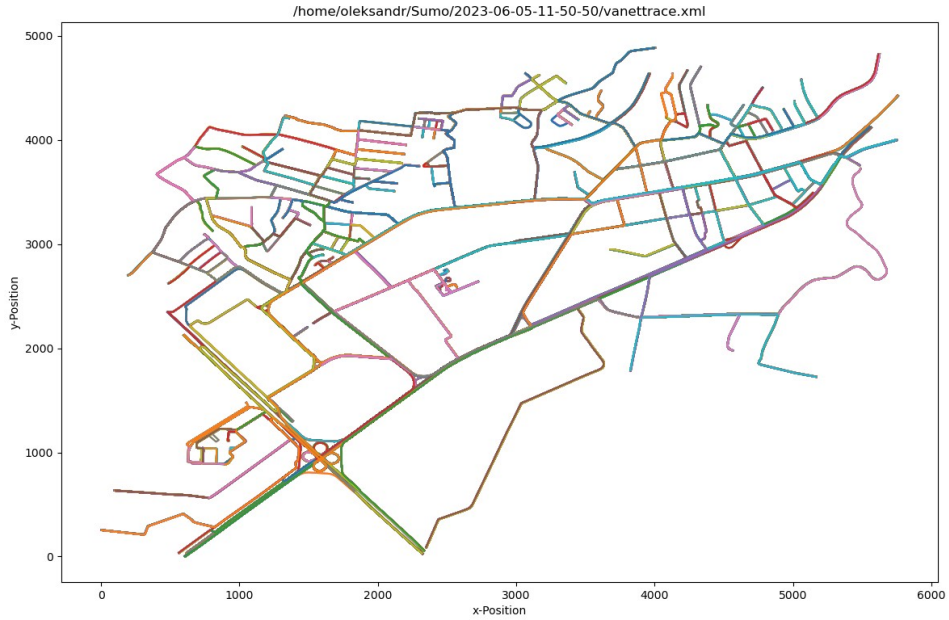


Рис. 3.3: Візуалізація траєкторії руху транспортних засобів у сценарії

Сконвертуємо створений на попередньому кроці файл формату .xml у файл формату .tcl. Для цього скористаємось командою «traceExporter.py -i trace.xml --ns2mobility-output=trace.tcl». Через великий розмір файлу та малу кількість даних, які потрібно отримувати, ми вручну знайдемо максимальне значення вузлів у згенерованому сценарії, яке у нашому випадку становить 7912. Тепер ми маємо усі параметри, щоб продовжувати симуляцію V2X в ns-3

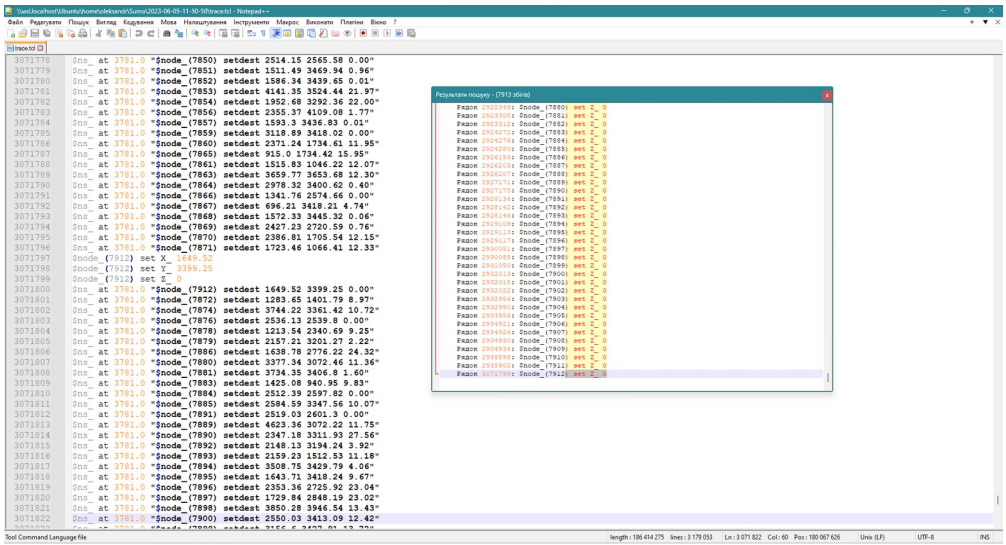


Рис. 3.4: Пошук кількості вузлів у сценарії

3.3 Налаштування скрипту для емуляції V2X

Для моделювання отриманого сценарію в ns-3 за основу було взято скрипт v2x-emulator[23]. Цей скрипт дозволяє надсилати CAM і DENM, згенеровані транспортними засобами, що (віртуально) рухаються по карті SUMO, через реальну мережу, покладаючись на фізичний інтерфейс. В процесі розробки його було значно модифіковано, зокрема додано інтеграцією з реальною картою.

Розберемо детальніше принцип роботи:

1. *Налаштування симуляції*: Сценарій створює контейнер для OBU (транспортних засобів) та налаштовує їх інтернет-стек, якщо увімкнено режим UDP. Потім він налаштовує модель мобільності дл OBU, яка визначатиме їх переміщення в межах симуляції.

2. *Інтеграція SUMO*: Сценарій налаштовує клієнт TraCI для взаємодії з SUMO, встановлюються різні параметри клієнта, такі як файл конфігурації, час початку, значення сіду, та інші

3. *Налаштування OBU*: сценарій відповідає за створення та налаштування програми-емулятора V2X, яка, ймовірно, є основним об'єктом зв'язку V2X в кожному бортовому пристрої (On-Board Unit; OBU) або транспортному засобі в мережі.

4. *Налаштування мережевого пристрою*: сценарій налаштовує мережевий пристрій для кожного транспортного засобу (On-Board Unit, OBU) у симуляції, щоб дозволити їм надсилати та отримувати пакети. Тип мережевого пристрою, що використовується, - емульований мережевий пристрій (Emulated Net Device, або EmuFdNetDevice)..

5. *Створення та конфігурація вузлів*: Для створення нових вузлів (транспортних засобів) визначена функція зворотного виклику. Для кожного нового вузла функція:

- Отримує вузол з пулу.

- Встановлює мережевий пристрій та встановлює MAC-адресу.

- Налаштовує IP-стек, якщо увімкнено режим UDP.

- Встановлює додаток v2xEmulator та встановлює час запуску та зупинки для нього.

6. *Вимикання вузла*: Ще одна функція зворотного виклику визначена для вимкнення вузлів. Коли вузол вимикається, він зупиняє всі додатки та встановлює позицію за межами діапазону зв'язку.

7. Імітаційний запуск: Нарешті, клієнт TraCI налаштовується з функціями створення та вимкнення вузлів. Потім симуляція запускається на задану тривалість, а ресурси очищаються після завершення симуляції.

Даний скрипт має значну кількість параметрів, доступних до редагування. Розглянемо їх:

Табл. 3.2: Параметри налаштування скрипту v2x-emulator

Параметр	Значення	Опис
deviceName	eth0	ім'я мережевого пристрою (зазвичай мережевої інтерфейсної карти), який емуляція використовуватиме для надсилання та отримання пакетів.
encapMode	Dix	Встановлює режим інкапсуляції для мережевого пристрою. "Dix" означає тип кадру Ethernet II, який є найпоширенішим типом кадру Ethernet, що використовується сьогодні.
udpIp		Цей рядок використовується для встановлення IP-адреси для режиму UDP. Порожній рядок вказує на те, що режим UDP не увімкнено.
gwstr	192.168.1.1	Це IP-адреса шлюзу в мережевому налаштуванні.
subnet	192.168.1.0	Це IP-адреса підмережі у мережевому налаштуванні.
netmask	255.255.255.0	Це мережева маска для IP-адреси, яка використовується для ідентифікації мережевої та

		хост-частини IP-адреси.
destPort	0	Номер порту призначення для UDP-з'єднання.
sumo_config	map.sumo.cfg	Це шлях до конфігураційного файлу SUMO
sumo_netns		Простір мережевих імен для симуляції SUMO.
sumo_startTime	0.0	Встановлює час запуску симуляції SUMO.
sumo_port	3400	Встановлює номер порту для симуляції SUMO.
sumo_penetrationRate	1.0	Встановлює швидкість проникнення для симуляції SUMO
sumo_seed	10	Встановлює початкове значення для генератора випадкових чисел, що використовується в симуляції SUMO
sumo_WaitForSocket	1.0	Встановлює час очікування з'єднання з сокетом у симуляції SUMO.
sendCam;	true	Це булеве значення визначає, чи надсилати в симуляції повідомлення про співпрацю (Cooperative Awareness Messages, CAM), чи ні.
sendDenm	true	Це логічне значення визначає, чи надсилати децентралізовані повідомлення про стан оточення (DENM) під час симуляції.
verbose	true	Це булеве значення вмикає або вимикає розгорнуте ведення журналу.
sumo_gui	true	Це булеве значення визначає, чи використовувати графічний інтерфейс SUMO під час симуляції.
sumo_updates	1	Встановлює інтервал оновлення для симуляції SUMO.
vehicle_vis	true	Це булеве значення вмикає або вимикає візуалізацію транспортних засобів у симуляції.
m_baseline_prr	150.0	Встановлює базову швидкість прийому пакетів.
m_prr_sup	false	Це булеве значення вмикає або вимикає супервізор швидкості прийому пакетів.
emuTime	5000	Задає загальний час емуляції у секундах.
numberOfNodes	7912	Задає кількість вузлів

Зі заданими параметрами проведемо симуляцію у двох режимах: у стандартному та режимі UDP

3.3.1. Симуляція в стандартному режимі

Стандартний режим емуляції, коли режим UDP вимкнено, передбачає моделювання декількох транспортних засобів, кожен з яких оснащений базовими послугами Cooperative Awareness Message (CAM) та Decentralized Environmental Notification Message (DENM). Ці транспортні засоби генерують і передають CAM-повідомлення та DENM-повідомлення, які відповідають стандарту ASN.1. Ці повідомлення передаються через фізичний інтерфейс, що відкриває можливості для майбутнього тестування та оцінки апаратного забезпечення в циклі.

Коли режим UDP вимкнено, код в основному імітує зв'язок V2X без використання адресації IP та транспортування за допомогою протоколу UDP, а повідомлення надсилатимуться через вказаний інтерфейс у вигляді широкомовних пакетів, інкапсульованих у BTP та GeoNetworking.

У цьому режимі при налаштуванні OBU, встановлюється лише об'єкт TraciClient для доступу до SUMO у додатку та два логічних параметри для надсилання CAM та DENM. Окрім цього, при створені вузла пропускається частина конфігурації IP-стеку. Варто зазначити, що без увімкненого режиму UDP транспортні засоби в симуляції не будуть взаємодіяти через UDP/IP, що може обмежити типи взаємодій та комунікації, які можуть бути змодельовані.

При запуску сценарію у цьому режимі, ми бачимо детальніше, як транспортні засоби створюються і починають свою роботу в моделюванні, і як обмінюються між собою повідомленнями

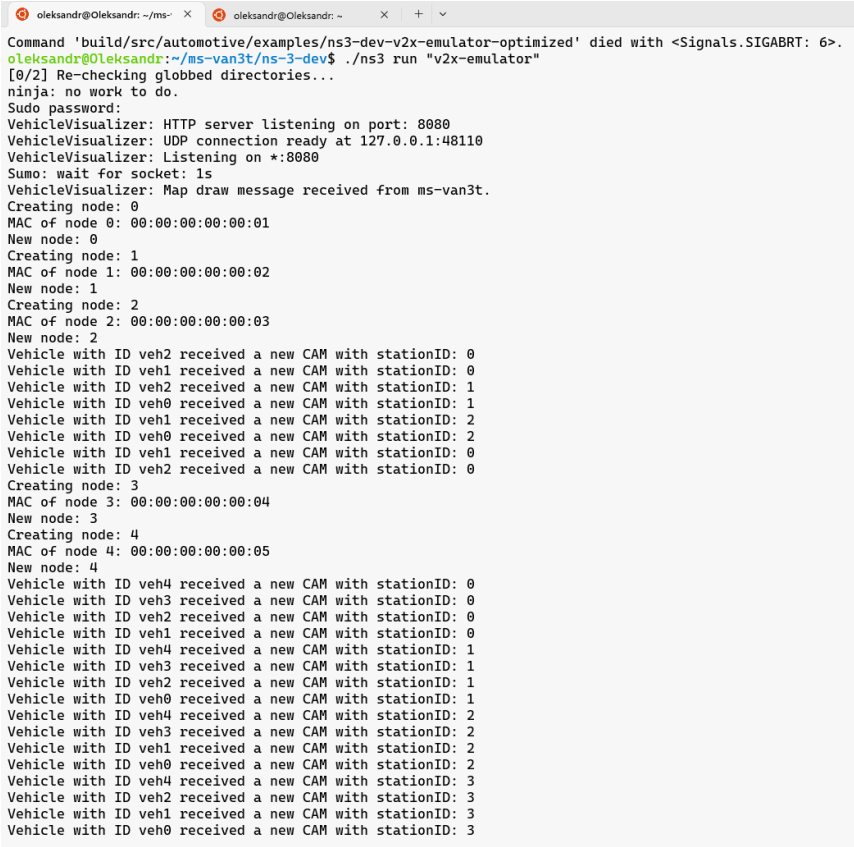


Рис. 3.5: Робота програми у стандартному режимі

Оскільки обмін повідомленнями йде через фізичний інтерфейс, а не використовуючи будь-яку імітаційну модель ns-3, можемо проаналізувати трафік за допомогою Wireshark і дослідити кожен пакет

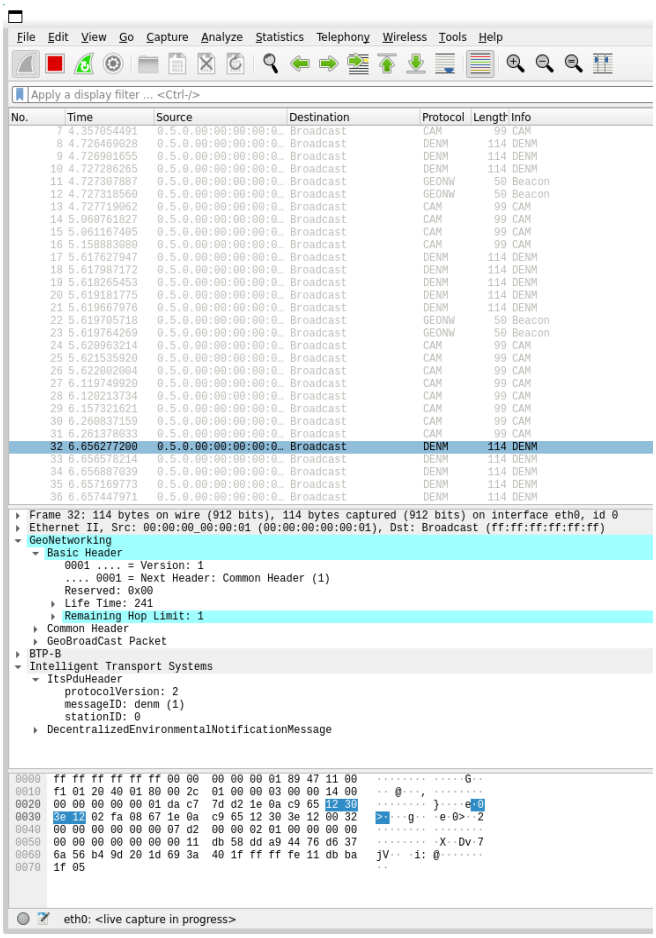


Рис. 3.6: Захоплення програмою Wireshark повідомлень, надісланих транспортними засобами при роботі у звичайному режимі

3.3.2. Симуляція в режимі UDP

Якщо режим UDP увімкнено, робота коду змінюється кількома способами:

- Відбувається налаштування мережі UDP. Це включає налаштування інтернет-стеку для вузлів (OBU), встановлення базової IPv4-адреси та маски підмережі, а також "випалювання" першої IP-адреси, яка може бути призначена маршрутизатору в локальній мережі.
- Для кожного створеного вузла, якщо увімкнено режим UDP, код налаштовує IP-стек вузлів/транспортних засобів. Це включає додавання інтерфейсу до стеку протоколу IPv4 вузла, призначення IP-адреси

інтерфейсу, налаштування метрики для інтерфейсу та встановлення маршруту за замовчуванням для статичної маршрутизації.

Ці кроки дозволяють OBU спілкуватися один з одним за допомогою UDP, який є протоколом без з'єднання. Це означає, що пакети даних можуть надсилатися з одного вузла на інший без необхідності встановлення з'єднання між вузлами. Це особливо корисно в автомобільних мережах, де вузли (транспортні засоби) постійно рухаються, а топологія мережі швидко змінюється.

При запуску сценарію у цьому режимі, ми бачимо детальніше, як транспортні засоби створюються, і на відмінну роботи у звичайному режимі, кожному транспортному засобу задається власна IP адреса

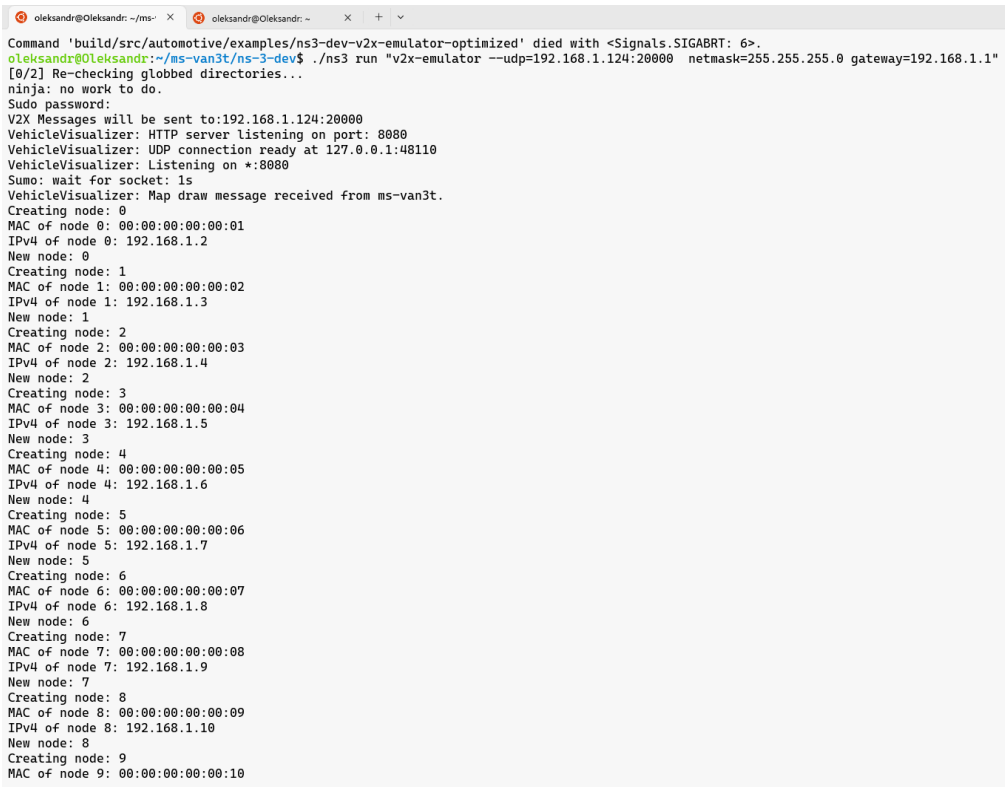


Рис. 3.7: Робота програми у режимі UDP

При аналізі трафіку, ми бачимо як кожен транспортний засіб робить запит до UDP серверу, а не спілкується напряму з іншими транспортними засобами

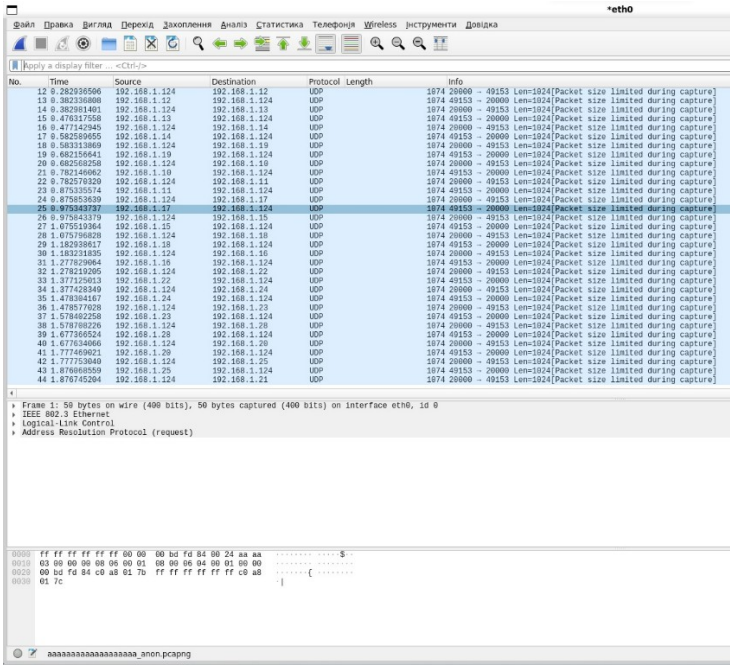


Рис. 3.8: Захоплення програмою Wireshark повідомлень, надісланих транспортними засобами при роботі у режимі UDP

3.4. Симуляція сценарію в SUMO

У ході виконання скрипту також відкривається графічний інтерфейс SUMO, оскільки параметр sumo_gui має значення true.

Для правильної роботи додаток-емулятор повинен завжди працювати в режимі реального часу, а пристрій, на якому запущено ns-3, повинен бути здатним обробляти вказану кількість транспортних засобів без затримок і без уповільнення. І через те, що ми маємо досить велику карту, то загалом моделювання триває 4998 секунд, тобто 1 годину 23 хвилини.

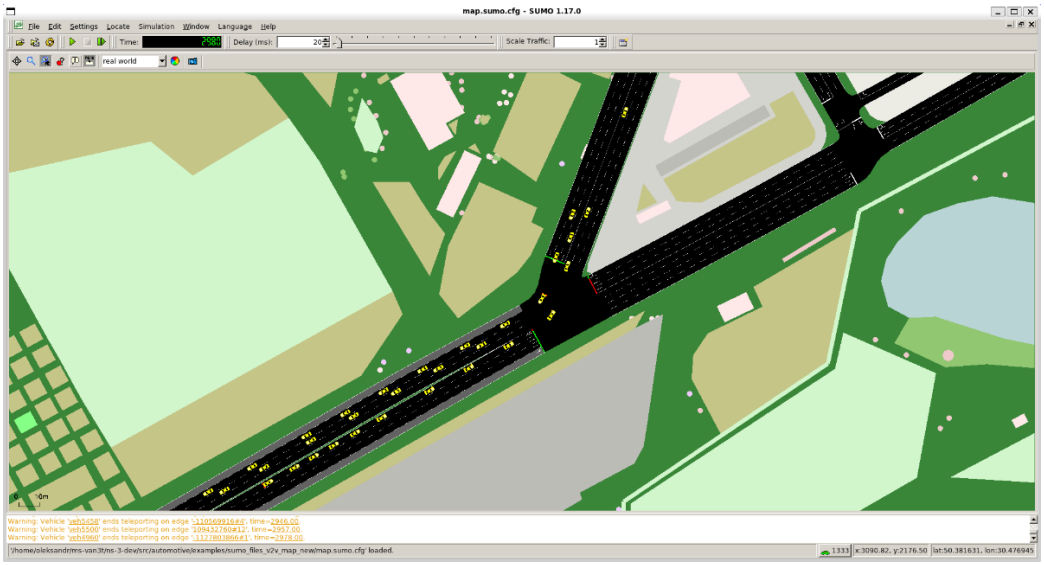


Рис. 3.9: Рух транспортних засобів під час моделювання сценарію

У ході симуляції можемо переглянути велику кількість параметрів.

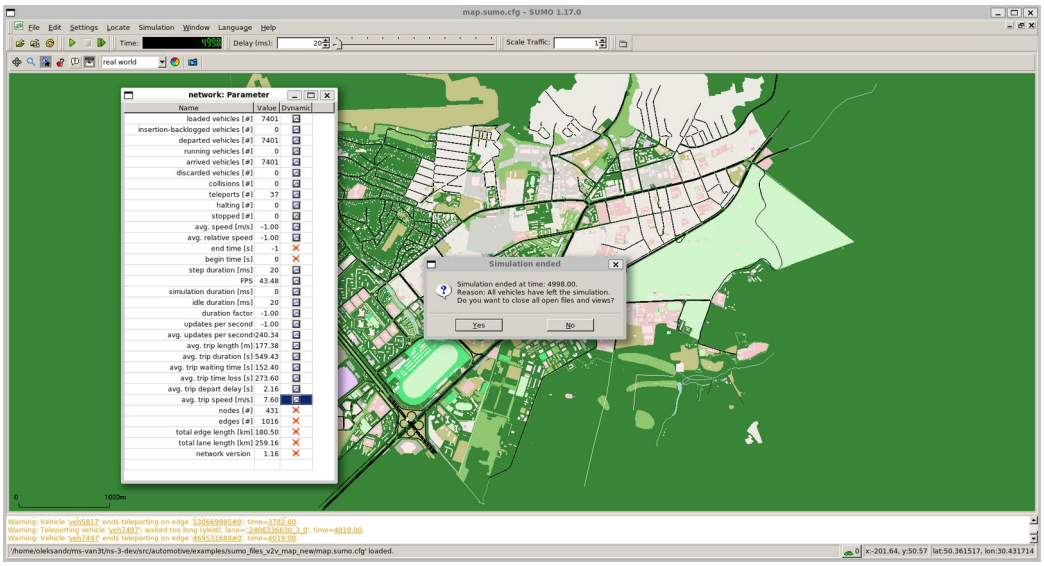


Рис. 3.10: Значення параметрів після завершення симуляції сценарію

Їх досить багато, тому розглянемо лише основні параметри детальніше:

Табл. 3.3: Параметри сценарію

Назва параметру	Значення	Опис
loaded vehicles [#]	7401	Це загальна кількість транспортних засобів,

		які були завантажені в симуляції. Ці транспортні засоби визначені у вхідних даних, але не всі з них могли розпочати або завершити свою поїздку.
departed vehicles [#]	7401	Кількість транспортних засобів, які розпочали свої поїздки в симуляції.
running vehicles [#]	0	Кількість транспортних засобів, які наразі знаходяться в мережі або в дорозі.
arrived vehicles [#]	7401	Кількість транспортних засобів, які успішно завершили свої поїздки.
discarded vehicles [#]	0	Кількість транспортних засобів, які були видалені з симуляції з інших причин, ніж успішне завершення поїздки.
collisions [#]	0	Кількість зіткнень, що сталися під час симуляції.
teleports [#]	37	Кількість разів, коли транспортний засіб було телепортовано. Телепортація відбувається, коли транспортний засіб переміщується симуляцією з одного місця в інше без проходження проміжного простору, часто через перевантаження мережі або інші проблеми.
halting [#]	0	Кількість транспортних засобів, які на даний момент зупинилися в мережі, не рухаючись.
stopped [#]	0	Кількість транспортних засобів, які повністю зупинилися в симуляції.
avg. speed [m/s]	-1.00	Середня швидкість усіх транспортних засобів у симуляції, виміряна в метрах за секунду.
avg. relative speed	-1.00	Середня швидкість усіх транспортних засобів відносно максимально дозволеної швидкості на поточному відрізку дороги.
end time [s]	-1	Час завершення симуляції, виміряний у секундах.
begin time [s]	0	Час початку симуляції, виміряний у секундах.

step duration [ms]	20	Тривалість кожного кроку симуляції, що вимірюється в мілісекундах.
FPS	35.71	Кадри в секунду, показник того, скільки разів за секунду оновлюється візуальне представлення симуляції.
simulation duration [ms]	0	Загальний час виконання симуляції в мілісекундах.
idle duration [ms]	20	Загальний час, проведений в симуляції, коли транспортні засоби не рухалися.
duration factor	-1.00	Коефіцієнт, який використовується для прискорення або сповільнення симуляції відносно реального часу.
updates per second	-1.00	Швидкість, з якою оновлюється стан симуляції.
avg. updates per second	110240.34	Середня швидкість оновлення симуляції в секунду протягом тривалості симуляції
avg. trip length [m]	4177.38	Середня довжина всіх завершених поїздок у симуляції, виміряна в метрах.
avg. trip duration [s]	549.43	Середня тривалість усіх завершених поїздок у симуляції, виміряна в секундах.
avg. trip waiting time [s]	152.40	Середній час, проведений транспортними засобами в очікуванні (без руху) під час поїздок.
avg. trip time loss [s]	273.60	Середній час, втрачений через дорожні умови або інші затримки під час поїздки.
avg. trip depart delay [s]	2.16	Середня затримка між запланованим та фактичним часом відправлення транспортних засобів.
avg. trip speed [m/s]	7.60	Середня швидкість транспортних засобів за час їхніх завершених поїздок, виміряна в метрах за секунду.
nodes [#]	431	Кількість вузлів (перехресть або розв'язок) у дорожній мережі, що використовується в

		симуляції.
edges [#]	1016	Кількість ребер (відрізків доріг) в мережі доріг, що використовуються в моделі.
total edge length [km]	180.50	Загальна довжина всіх ребер (відрізків доріг) у мережі, що моделюється, виміряна в кілометрах.
total lane length [km]	259.16	Загальна довжина всіх смуг руху в модельній мережі, виміряна в кілометрах

Оскільки для частини параметрів ми маємо можливість побачити їх зміну у часі, наведемо графіки зміни деяких з них

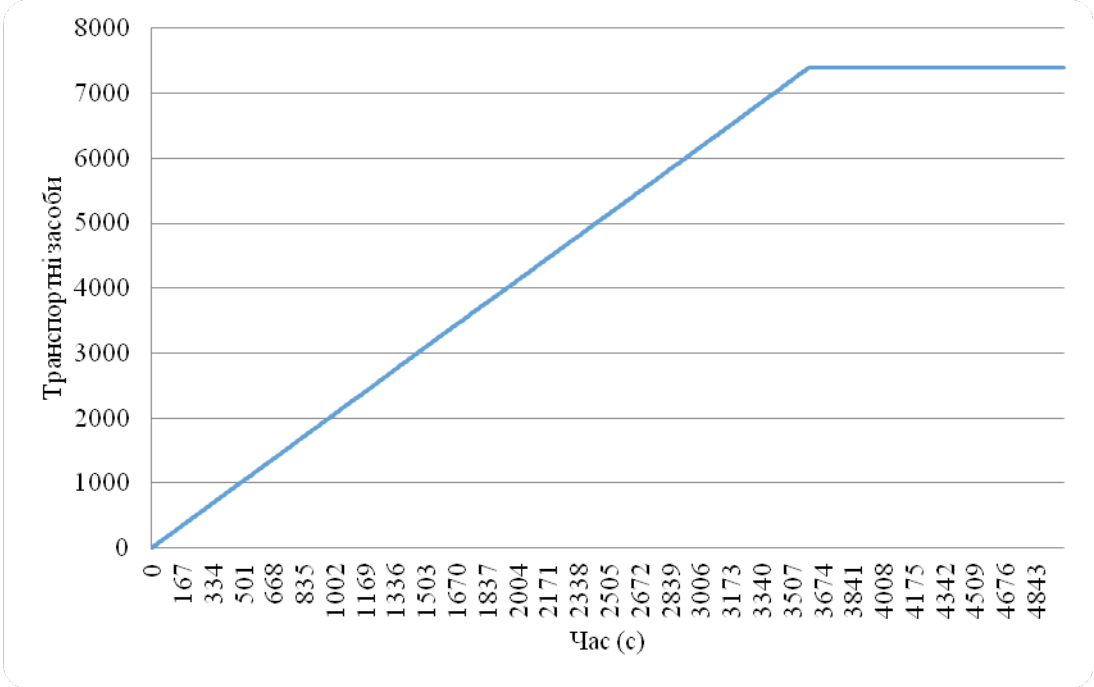


Рис. 3.11: Графік зміни кількості транспортних засобів, які розпочали свої поїздки в симуляції

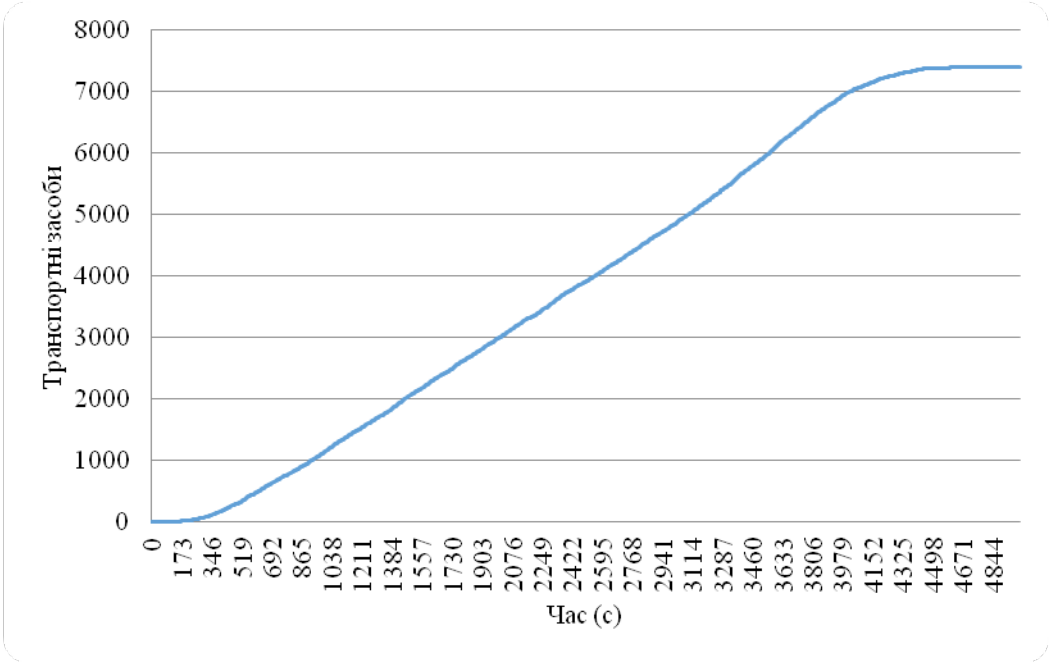


Рис. 3.12: Графік зміни кількості транспортних засобів, які успішно завершили свої поїздки

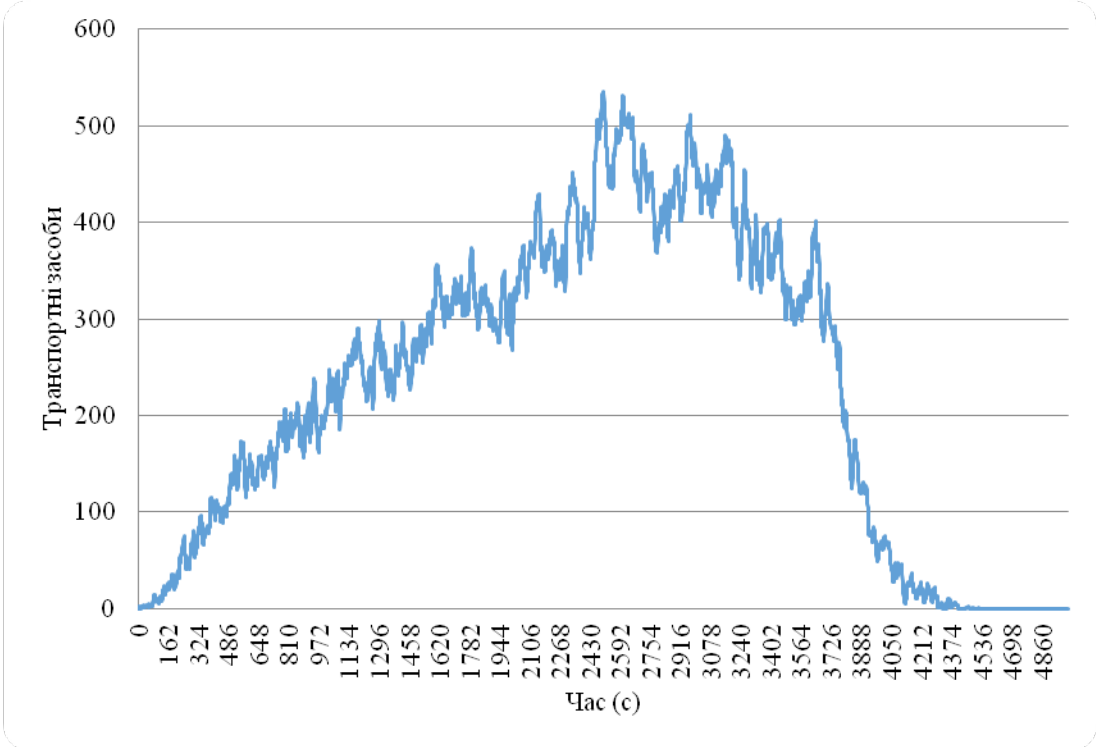


Рис. 3.13: Графік зміни кількості транспортних засобів, які на даний момент зупинилися в мережі, не рухаючись.

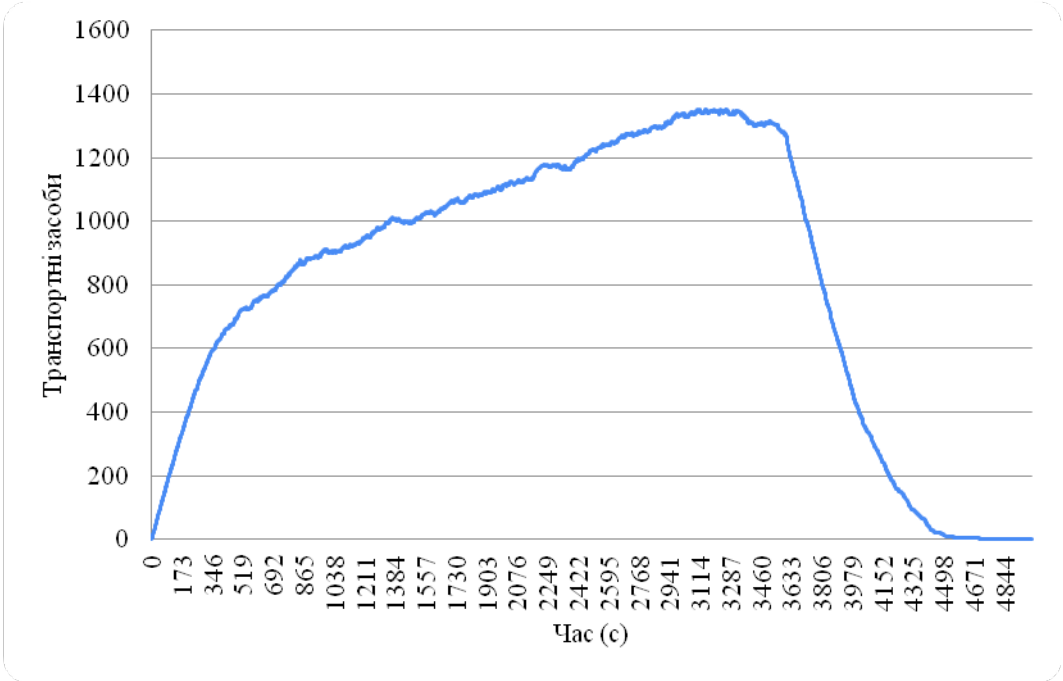


Рис. 3.14: Графік зміни кількості транспортних засобів, які рухаються

Додатково накладемо графік зміни кількості транспортних засобів, які почали поїздки, які завершили поїздку, та які рухаються у симуляції

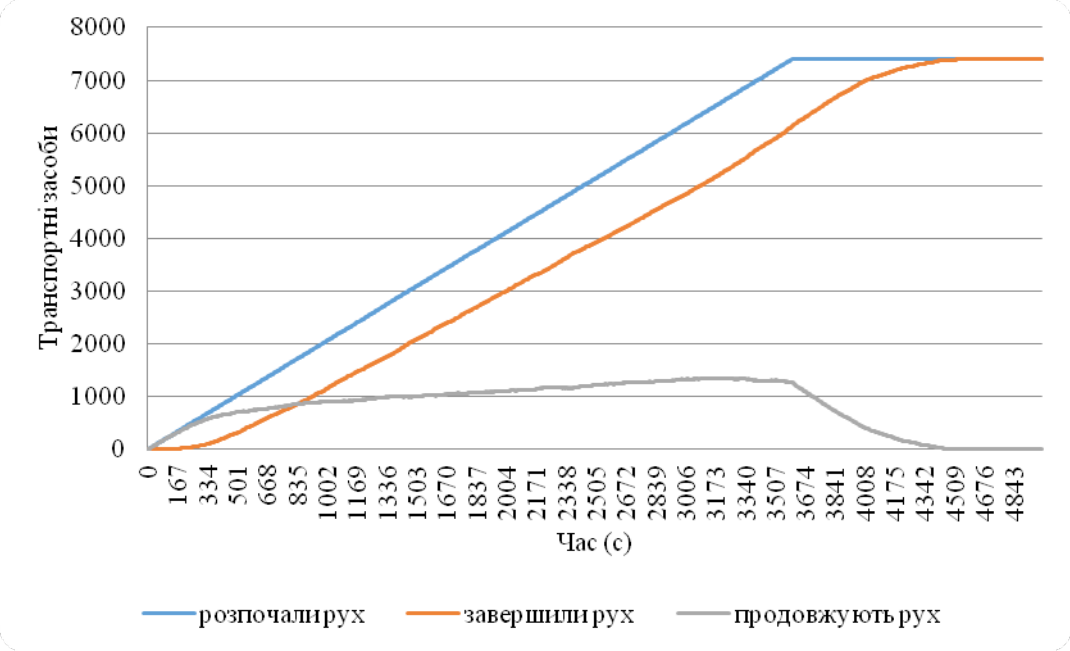


Рис. 3.15: Об’єднаний графік

На цьому графіку ми можемо побачити, як кількість транспортних засобів, що розпочали рух, лінійно зростала впродовж години, а далі залишалась майже незмінною. Кількість же транспортних засобів, які закінчили рух, зростала з меншою швидкість і рівномірно, але не лінійно, що ми можемо побачити по графіку зміни кількості автомобілів, які продовжують рух

3.5. Обговорення результатів та висновок

У цьому розділі ми успішно виконали V2X моделювання, використовуючи комбінацію SUMO та ns-3.

За допомогою інструментів пакету SUMO було створено реалістичний сценарій дорожнього руху частини Голосіївського району Києва. Створена цифрова карта відображає міське розмаїття та заходи з управління дорожнім рухом, притаманні обраній локації.

Звернувшись до нашого мережевого симулятора ns-3, ми провели моделювання у двох різних режимах: звичайному та режимі UDP. Вмикаючи та вимикаючи UDP, ми змогли отримати цінну інформацію про його вплив на доставку даних, затримку та втрату пакетів. Зокрема, ми дійшли висновку, що звичайному режимі повідомлення надсилаються через вказаний фізичний інтерфейс і поширюються на всі пристрої в мережі. Цей режим зазвичай використовується, коли потрібно емулювати прямий бездротовий зв'язок між транспортними засобами, який є основою для більшості сценаріїв V2X. З іншого боку, режим UDP підходить, коли ми хочемо, щоб симуляція включала передачу повідомлень на зовнішній UDP-сервер. Цей режим буде особливо корисним, коли ми хочемо розширити симуляцію для взаємодії із зовнішніми серверами, системами або службами. Наприклад, ми можемо використовувати цей режим, коли хочемо з'єднати симуляцію з центром управління трафіком або хмарним сервісом.

Далі ми об'єднали сценарій трафіку від SUMO зі сценарієм мережі від ns-3 і виконали комплексне моделювання трафіку. Спостереження, отримані в результаті цієї симуляції, надали багато даних і розуміння поведінки зв'язку V2X в реальних умовах.

На закінчення, поєднання SUMO і ns-3 ефективно продемонструвало потужність і потенціал зв'язку V2X. Завдяки ретельному та реалістичному моделюванню Голосіївського району Києва ми розкрили нюанси ролі UDP у мережевому середовищі V2X. Крім того, наш підхід дав практичне розуміння того, як різні фактори можуть впливати на ефективність та надійність V2X-зв'язку. Це слугує безцінним підґрунтям для нашого подальшого дослідження потенційних покращень та оптимізацій для систем V2X.

ВИСНОВКИ

У ході виконання дипломної роботи ми дослідили дедалі важливішу роль технології V2X в інтелектуальних транспортних системах, зосередившись на моделюванні дорожнього руху та мереж. Ми глибоко занурилися у використання двох різних симуляторів, а саме SUMO для моделювання дорожнього руху та ns-3 для моделювання мереж, і на реальному прикладі продемонстрували спосіб їх застосування у сфері сучасних транспортних систем..

У першому розділі ми дослідили ключові аспекти інтелектуальних транспортних систем та зв'язку V2X. Ми розглянули, як ITS змінюють сучасний транспорт і роль даних про дорожній рух у цих системах. Ми обговорили мережеву архітектуру, необхідну для зв'язку V2X, та її важливість для безпеки та ефективності транспортних засобів. Нарешті, ми заглибилися в розподіл ресурсів, особливо в контексті мереж 5G, які мають вирішальне значення для майбутнього ITS.

У другому розділі ми заглибилися у сферу моделювання трафіку та мереж у V2X. Ретельне вивчення та порівняння різних симуляторів привело нас до вибору SUMO та ns-3 як ідеальних інструментів для нашого дослідження завдяки їх комплексним можливостям, гнучкості та широкому використанню в галузі.

У практичній частині ми успішно створили реалістичний сценарій дорожнього руху для Голосіївського району Києва за допомогою SUMO. Схеми руху та точки заторів, що спостерігалися в симуляції, точно відповідали реальним умовам району, демонструючи точність та надійність обраного нами симулятора. Унікальністю цієї практичної частини є те, що з реальною картою міста Києва така симуляція була проведена вперше

Запуск симуляції в ns-3, як в стандартному режимі, так і в режимі UDP, дозволив нам проаналізувати продуктивність V2X-зв'язку в різних умовах мережі.

На закінчення, наша робота представляє глибокий погляд на можливості і потенціал V2X-зв'язку в інтелектуальних транспортних системах, демонструючи його важливість у забезпеченні більш безпечного та ефективного дорожнього руху. Успішне моделювання та аналіз ще раз підтверджують життєздатність використання передових інструментів, таких як SUMO і ns-3, для вивчення і вдосконалення систем V2X. Подальша робота може розширити наші висновки, досліджуючи інші мережеві архітектури або методи моделювання, а також застосовуючи наші методи до різних географічних місць і сценаріїв руху.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. “Global status report on road safety 2021”, Всесвітня організація охорони здоров’я, грудень 2021.
2. Francisco J. Martinez et al. “Emergency Services in Future Intelligent Transportation Systems Based on Vehicular Communication Networks”, IEEE Intelligent Transportation Systems Magazine, October 2020, vol. 2, issue. 2, pp. 6-20.
3. Susan A. Shaheen, Rachel Finson, в Encyclopedia of Energy, 2019;
4. Li Feng et al., “V2X White Paper by NGMN Alliance”, project. NGMN V2X Task Force, confidentiality class. P -Public, approved by / date. NGMN Board / 10th July 2021.
5. Andreas Festag, “Standards for Vehicular Communication—from IEEE 802.11p to 5G”, e & i Elektrotechnik und Informationstechnik, November 2022, vol. 132, issue. 7, pp. 409-416
6. Release 14 V2X Sidelink PSCCH and PSSCH Throughput [Електронний ресурс] // Режим доступу до ресурсу: https://www.mathworks.com/help/lte/ug/release-14-v2x-sidelink-pssch-throughput.html
7. 3GPP TS 36.213 "Процедури фізичного рівня"
8. 3GPP TS 36.101 "Обладнання користувача (UE) радіопередача та прийом"
9. Schuenemann, B. (2011), V2x Simulation Runtime Infrastructure Vsimrti: An Assessment Tool to Design Smart Traffic Management Systems, Computer Networks, 55(14), 3189-3198.
10. https://youtu.be/cRyZ_yni2i8
11. F. Karnadi, Z. Mo & K. Lan “Rapid Generation of Realistic Mobility Models for VANET,” WCNC’07
12. M. Šrotýř, Optimalizace bezdrátové komunikace v kooperativních systémech (Doctoral dissertation), CTU, Prague, 2021.

13. Riley, G.F.; Henderson, T.R. The ns-3 Network Simulator. In Modeling and Tools for Network Simulation; Wehrle, K., Güne¸s, M., Gross, J., Eds.; Springer: Berlin/Heidelberg, Germany, 2020; pp. 15–34.
14. L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, and H. Yu, "Advances in Network Simulation," IEEE Computer, no. May, pp. 59-67, 2020.
15. S. Bhatt, R. Fujimoto, A. Ogielski, and K. Perumalla, "Parallel simulation techniques for large-scale networks," IEEE Communications Magazine, August, pp. 42-46, 2019.
16. Tonguz, Ozan K. (25 Sep 2022). "How Vehicle-to-Vehicle Communication Could Replace Traffic Lights and Shorten Commutes". IEEE Spectrum.
17. DLR and contributors, SUMO Homepage [Online], URL: http://sumo.sourceforge.net/
18. D. Krajzewicz. “Traffic Simulation with SUMO - Simulation of Urban Mobility”. In: Fundamentals of Traffic Simulation International Series in Operations Research and Management Science. Springer. Seiten 269-294. ISBN 978-1-4419-6141-9. ISSN 0884-8289, 2010.
19. Documentation - SUMO Documentation (dlr.de) URL: https://sumo.dlr.de/docs/index.html
20. ns-3 Manual Release 3.38. URL: https://www.nsnam.org/docs/release/3.38/manual/singlehtml/index.html
21. OSMWebWizard - SUMO Documentation (dlr.de) URL: https://sumo.dlr.de/docs/Tutorials/OSMWebWizard.html
22. TraceExporter - SUMO Documentation URL: https://sumo.dlr.de/docs/Tools/TraceExporter.html
23. ms-van3t-devs/ms-van3t: A multi-stack, ETSI compliant, V2X framework for ns-3. URL: https://github.com/ms-van3t-devs/ms-van3t/tree/master

ДОДАТКИ

Код скрипта v2x-emulator.cc:

```
/* -*- Mode:C++; c-file-style:"gnu"; indent-tabs-mode:nil; -*- */
/*
 * This program is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License version 2 as
 * published by the Free Software Foundation;
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA

*/

#include <fstream>
#include "ns3/core-module.h"
#include "ns3/internet-module.h"
#include "ns3/applications-module.h"
#include "ns3/fd-net-device-module.h"
#include "ns3/mobility-module.h"
#include "ns3/traci-module.h"
#include "ns3/v2xEmulator-helper.h"
#include "ns3/emu-fd-net-device-helper.h"
#include "ns3/v2xEmulator.h"
#include "ns3/packet-socket-helper.h"

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("v2x-emulator");

int
main (int argc, char *argv[])
{
  std::string deviceName ("eth0");
  std::string encapMode ("Dix");

  std::string udpIp = "192.168.1.124:20000";
  std::string gwstr = "192.168.1.1";
  std::string subnet = "192.168.1.0";
  std::string netmask = "255.255.255.0";
  int destPort = 0;
  Ipv4Address destAddr;
  Ipv4AddressHelper ipv4helper;

  std::string sumo_config = "src/automotive/examples/sumo_files_v2v_map_new/map.sumo.cfg";
  std::string sumo_netns = "";

  double sumo_startTime = 0.0;
  int sumo_port = 3400;
```

```
53.  double sumo_penetrationRate = 1.0;
54.  int sumo_seed = 10;
55.  double sumo_WaitForSocket = 1.0;
56.
57.  bool sendCam = true;
58.  bool sendDenm = true;
59.
60.  bool verbose = true;
61.  bool sumo_gui = true;
62.  double sumo_updates = 1;
63.  bool print_summary = false;
64.
65.  bool vehicle_vis = true;
66.  double m_baseline_prr = 150.0;
67.  bool m_prr_sup = false;
68.
69.  double emuTime = 5000;
70.
71.  int numberOfNodes = 7912;
72.  uint32_t nodeCounter = 0;
73.
74.  CommandLine cmd;
75.
76.  cmd.AddValue ("sumo-gui", "Use SUMO gui or not", sumo_gui);
77.  cmd.AddValue ("sumo-updates", "SUMO granularity", sumo_updates);
78.  cmd.AddValue ("sumo-config", "Location and name of SUMO configuration file", sumo_config);
79.  cmd.AddValue ("sumo-netns",
80.               "[Advanced users] Name of the network namespace SUMO shall be launched in. If not "
81.               "specified, SUMO is launched normally.",
82.               sumo_netns);
83.  cmd.AddValue ("summary", "Print a summary for each vehicle at the end of the simulation",
84.               print_summary);
85.  cmd.AddValue ("verbose", "Enable verbose printing on stdout", verbose);
86.  cmd.AddValue ("interface", "Name of the physical interface to send V2X messages to", deviceName);
87.  cmd.AddValue ("sim-time", "Total duration of the emulation [s]", emuTime);
88.  cmd.AddValue ("send-cam", "To trigger the CAM dissemination", sendCam);
89.  cmd.AddValue ("send-denm", "To trigger the DENM dissemination", sendDenm);
90.  cmd.AddValue ("udp",
91.               "[UDP mode] To enable UDP mode and specify UDP port and IP address where the V2X "
92.               "messages are redirected (format: <IP>:<port>)",
93.               udpIp);
94.  cmd.AddValue ("gateway",
95.               "[UDP mode] To specify the gateway at which the UDP/IP packets will be sent",
96.               gwstr);
97.  cmd.AddValue ("subnet",
98.               "[UDP mode] To specify the subnet which will be used to assign the IP addresses of "
99.               "emulated nodes (the .1 address is automatically excluded)",
100.              subnet);
101. cmd.AddValue ("netmask", "[UDP mode] To specify the netmask of the network", netmask);
102. cmd.AddValue ("baseline", "Baseline for PRR calculation", m_baseline_prr);
103. cmd.AddValue ("vehicle-visualizer", "Activate the web-based vehicle visualizer for ms-van3t", vehicle_vis);
104. cmd.AddValue ("prr-sup","Use the PRR supervisor or not",m_prr_sup);
105.
106.
107. cmd.Parse (argc, argv);
108.
109. if (verbose)
110.   {
111.     LogComponentEnable ("v2x-emulator", LOG_LEVEL_INFO);
112.     LogComponentEnable ("CABasicService", LOG_LEVEL_INFO);
```

```
    LogComponentEnable ("DENBasicService", LOG_LEVEL_INFO);
  }

GlobalValue::Bind ("SimulatorImplementationType", StringValue ("ns3::RealtimeSimulatorImpl"));

GlobalValue::Bind ("ChecksumEnabled", BooleanValue (true));

if (udpIp != "")
  {
    std::istringstream udpIpStream (udpIp);
    std::string curr_str;
    // IP address
    std::getline (udpIpStream, curr_str, ':');

    destAddr = Ipv4Address (curr_str.c_str ());

    // Port
    std::getline (udpIpStream, curr_str, ':');
    destPort = std::stoi (curr_str);

    if (destPort <= 0 || destPort > 65535)
      {
        NS_FATAL_ERROR ("Error: " << destPort << " is not a valid port for UDP operations.");
      }

    std::cout << "V2X Messages will be sent to:" << destAddr << ":" << destPort << std::endl;
  }

NS_LOG_INFO ("Simulation will last " << emuTime << " seconds");
ns3::Time simulationTime (ns3::Seconds (emuTime));

NodeContainer nodes;
nodes.Create (numberOfNodes);

if (udpIp != "")
  {
    InternetStackHelper internet;
    internet.SetIpv4StackInstall (true);
    internet.Install (nodes);
    ipv4helper.SetBase (subnet.c_str (), netmask.c_str ());
    ipv4helper
        .NewAddress ();
  }

MobilityHelper mobility;
mobility.Install (nodes);

Ptr<TraciClient> sumoClient = CreateObject<TraciClient> ();
sumoClient->SetAttribute ("SumoConfigPath", StringValue (sumo_config));
sumoClient->SetAttribute ("SumoBinaryPath", StringValue ("")); // use system installation of sumo
sumoClient->SetAttribute ("SynchInterval", TimeValue (Seconds (sumo_updates)));
sumoClient->SetAttribute ("StartTime", TimeValue (Seconds (sumo_startTime)));
sumoClient->SetAttribute ("SumoGUI", (BooleanValue) sumo_gui);
sumoClient->SetAttribute ("SumoPort", UintegerValue (sumo_port));
sumoClient->SetAttribute ("PenetrationRate", DoubleValue (sumo_penetrationRate));
sumoClient->SetAttribute ("SumoLogFile", BooleanValue (false));
sumoClient->SetAttribute ("SumoStepLog", BooleanValue (false));
sumoClient->SetAttribute ("SumoSeed", IntegerValue (sumo_seed));
sumoClient->SetAttribute ("SumoAdditionalCmdOptions", StringValue ("--verbose true"));
sumoClient->SetAttribute ("SumoWaitForSocket", TimeValue (Seconds (sumo_WaitForSocket)));
```

```
173.  sumoClient->SetAttribute ("SumoAdditionalCmdOptions",
174.                           StringValue ("--collision.action warn --collision.check-junctions "
175.                                        "--error-log=sumo-errors-or-collisions.xml"));
176.  sumoClient->SetAttribute ("UseNetworkNamespace", StringValue (sumo_netns));
177.
178.  vehicleVisualizer vehicleVisObj;
179.  Ptr<vehicleVisualizer> vehicleVis = &vehicleVisObj;
180.  if (vehicle_vis)
181.    {
182.      vehicleVis->startServer ();
183.      vehicleVis->connectToServer ();
184.      sumoClient->SetAttribute ("VehicleVisualizer", PointerValue (vehicleVis));
185.    }
186.
187.  Ptr<PRRSupervisor> prrSup = NULL;
188.  PRRSupervisor prrSupObj (m_baseline_prr);
189.  if (m_prr_sup)
190.    {
191.      prrSup = &prrSupObj;
192.      prrSup->setTraCIClient (sumoClient);
193.    }
194.
195.  v2xEmulatorHelper emuHelper;
196.  emuHelper.SetAttribute (
197.      "Client",
198.      (PointerValue) sumoClient);
199.  emuHelper.SetAttribute ("SendCAM", (BooleanValue) sendCam);
200.  emuHelper.SetAttribute ("SendDENM", (BooleanValue) sendDenm);
201.  if (udpIp != "")
202.    {
203.      emuHelper.SetAttribute ("DestinationIPv4", Ipv4AddressValue (destAddr));
204.      emuHelper.SetAttribute ("DestinationPort", IntegerValue (destPort));
205.      emuHelper.SetAttribute ("UDPmode", (BooleanValue) true);
206.    }
207.
208.  EmuFdNetDeviceHelper emuDev;
209.  emuDev.SetDeviceName (deviceName);
210.  emuDev.SetAttribute ("EncapsulationMode", StringValue (encapMode));
211.
212.  if (udpIp == "")
213.    {
214.      PacketSocketHelper packetSocket;
215.      packetSocket.Install (nodes);
216.    }
217.
218.  NetDeviceContainer fdnetContainer;
219.  Ipv4InterfaceContainer ipv4ic;
220.  Ipv4Address gateway (gwstr.c_str ());
221.  Ipv4StaticRoutingHelper ipv4RoutingHelper;
222.
223.  STARTUP_FCN setupNewEmuNode = [&] (std::string vehicleID) -> Ptr<Node> {
224.    if (nodeCounter >= nodes.GetN ())
225.      NS_FATAL_ERROR ("Node Pool empty!: " << nodeCounter << " nodes created.");
226.
227.    std::cout << "Creating node: " << nodeCounter << std::endl;
228.
229.    Ptr<Node> includedNode = nodes.Get (nodeCounter);
230.    ++nodeCounter;
231.
232.    std::ostringstream veh_mac;
```

```
233.    veh_mac << "00:00:00:00:00:" << std::setfill ('0') << std::setw (2) << nodeCounter;
234.    fdnetContainer = emuDev.Install (includedNode);
235.    Ptr<FdNetDevice> dev = fdnetContainer.Get (0)->GetObject<FdNetDevice> ();
236.    dev->SetAddress (Mac48Address (veh_mac.str ().c_str ()));
237.
238.    std::cout << "MAC of node " << nodeCounter - 1 << ": " << veh_mac.str () << std::endl;
239.
240.    if (udpIp != "")
241.      {
242.        Ptr<Ipv4> ipv4 = includedNode->GetObject<Ipv4> ();
243.        uint32_t interface = ipv4->AddInterface (dev);
244.        ipv4ic = ipv4helper.Assign (fdnetContainer);
245.        Ipv4InterfaceAddress address =
246.            Ipv4InterfaceAddress (ipv4ic.GetAddress (0, 0), netmask.c_str ());
247.        ipv4->AddAddress (interface, address);
248.        ipv4->SetMetric (interface, 1);
249.        ipv4->SetUp (interface);
250.
251.        Ptr<Ipv4StaticRouting> staticRouting = ipv4RoutingHelper.GetStaticRouting (ipv4);
252.        staticRouting->SetDefaultRoute (gateway, interface);
253.        std::cout << "IPv4 of node " << nodeCounter - 1 << ": " << ipv4ic.GetAddress (0, 0)
254.                  << std::endl;
255.      }
256.
257.    ApplicationContainer v2xEmulatorApp = emuHelper.Install (includedNode);
258.    v2xEmulatorApp.Start (Seconds (0.0));
259.    v2xEmulatorApp.Stop (simulationTime - Simulator::Now () - Seconds (0.1));
260.
261.    std::cout << "New node: " << nodeCounter - 1 << std::endl;
262.
263.    return includedNode;
264.  };
265.
266.  SHUTDOWN_FCN shutdownEmuNode = [] (Ptr<Node> exNode, std::string vehicleID) {
267.    Ptr<v2xEmulator> v2xEmulatorApp_ = exNode->GetApplication (0)->GetObject<v2xEmulator> ();
268.    if (v2xEmulatorApp_)
269.      v2xEmulatorApp_->StopApplicationNow ();
270.    Ptr<ConstantPositionMobilityModel> mob = exNode->GetObject<ConstantPositionMobilityModel> ();
271.    mob->SetPosition (Vector (-1000.0 + (rand () % 25), 320.0 + (rand () % 25),
272.                              250.0));
273.  };
274.
275.  sumoClient->SumoSetup (setupNewEmuNode, shutdownEmuNode);
276.
277.  Simulator::Stop (simulationTime);
278.  Simulator::Run ();
279.  Simulator::Destroy ();
280.}
```