Final Project Report

Open-Source Prototyping of 5G Wireless Systems for Smart Ag, Autonomous Vehicles and Beyond

sdmay19-04

Team Members

- Hye-Sung Moon
  Email: mhss5458@iastate.edu
- Jaime Zetina
  Email: jzetina@iastate.edu
- Anthony Benson
  Email: anthonyb@iastate.edu
- Khanh Luu
  Email: kdluu@iastate.edu
- Jared Gorton
  Email: jmgorton@iastate.edu
- Theodore Miller
  Email: tgm@iastae.edu

Team Website: http://sdmay19-04.sd.ece.iastate.edu/team.html

Adviser & Client: Prof. Hongwei Zhang

Table of Contents

List of Acronyms

AV (Autonomous Vehicle)

BS (Base Station)

CPS (Cyber Physical Scheduling)

ER (Exclusion Region)

ETG (Electronic Technology Group of Department of Electrical and Computer Engineering at Iowa State University)

eNB (eNodeB)

GCS (Geometric Cellular Scheduling)

HW (Hardware)

IoT (Internet of Things)

OAI (Open Air Interface)

PRK (Physical-Ratio-K)

PRKS (Physical-Ratio-K Scheduling)

SNR (Sound to Noise Ratio)

SUMO (Simulation of Urban Mobility)

SW (Software)

TRaCI (Traffic Control Interface)

UCS (Unified Cellular Scheduling)

UE (User Equipment)

USRP (Universal Software Radio Peripheral)

V2V (Vehicle to Vehicle)

VM (Virtual Machine)

0. Executive Summary

Reliability of communications is one of the biggest hurdles in current wireless communications, and is a major issue that needs to be solved in order to facilitate many new technologies such as autonomous vehicles. This project was an attempt to implement a scheduling algorithm, which we have dubbed GCS, for 5G V2X communication that would increase reliability by using geographic position of vehicles to identify potentially conflicting user equipments and schedule transmission times accordingly. In order to simulate a realistic scenario we used SUMO to simulate vehicles and traffic conditions in and around the ISU campus. The GCS scheduling algorithm is being implemented using OAI which is an open source software and hardware development platform for the core network (EPC), access network and user equipment (EUTRAN) of 3GPP cellular networks. Our GCS algorithm is based on the previous CPS algorithm developed in part by our client and advisor Professor Hongwei Zhang. Though we were unable to complete the project in full we were able to complete several key aspects of the algorithm and provide a base for future work.

1. Requirements Specification

1.1 Functional Requirements

1. **Interference Identification**: is characterized by leveraging Cyber Physical Scheduling (CPS) and Unified Cellular Scheduling (UCS) into the designed Geometric Cyber Scheduling (GCS) algorithm. It will allow Vehicles (nodes in our case) identify or "mark" other nearby nodes with the potential to cause interference in the Vehicles communication attempts both in transmitting and receiving. The algorithm will allow real time location analysis and future location prediction. By identifying potential interference in present and future locations the vehicle will be able to correctly schedule communication between itself and other nodes with respect to time and frequency. The idea behind GCS is maximizing reliability, reducing latency, reducing interference between concurrent transmissions, and maximizing throughput.

1.2 Non-Functional Requirements

1. **Reliability**: The reliability is the essential part of V2V communication. Conventional network system scheduling protocols are unable to overcome the interferences between vehicles. Theoretically, we can obtain high reliability with the GCS algorithm, and we will test through simulation with OAI and SUMO.
2. **Delay**: Delay is another factor that we need to consider for simulations. In fast moving vehicles, a tiny amount of time delay can be a tremendous difference. The latency for 5G is conceptually around 1 ms. However, we will not be able to obtain this goal. The delay level will be as same as 4G network.
3. **Concurrency**: The number of simultaneous non-interfering transmissions in the same time slot is one part we should consider.
4. **Throughput**: The rate of successful packet transmission with respect to time should be maximized, but is a low priority than concurrency.

2. System Design & Development

2.1 Design Plan

Our project is divided into two major components: virtual simulation and physical implementation. First, we will run the simulation of the GCS algorithm over OAI and SUMO. In the software simulation, we will be able to run multiple UEs and check the reliability when the system has at least 75 UEs. However, when it comes to the hardware implementation, we are limited by how many units are available for use, so we are going to use 5 SDRs during the physical testing process to verify the GCS algorithm for correctness.

Figure 1: V2V Communication

2.2 Software

Listed softwares are the most important parts of our project. GCS algorithm will be built over OAI network simulator. To provide the vehicle dynamic, we will use SUMO and integrate with OAI. Both softwares are operating over Linux OS, we will use an ETG server to run these softwares.

- OAI - OAI's built-in simulator will be used for software testing and simulations. OAI can implement the full protocol stack to run on a real execution environment respecting frame timing constraints which make the algorithm built over OAI compatible with the SDRs.
- SUMO - SUMO will be used for simulation of traffic and to provide vehicle dynamics to UEs.

Our proposed software simulation will be based on OAI. For V2V wireless channels, we will implement in OAI a channel model based on the real-world.

The GCS algorithm is scheduling using a UE's physical location along with its communication data to be able to find interference. Unlike conventional communication systems, the advantage of the GCS algorithm is that it does not need the base station, because each UE can communicate with the others to overcome the interference and maximize the reliability of overall communications. The GCS algorithm we will be implementing is based on the PRKS algorithm. The number of UE nodes that we will be using is dependent on our processing power, but should be at least 75.

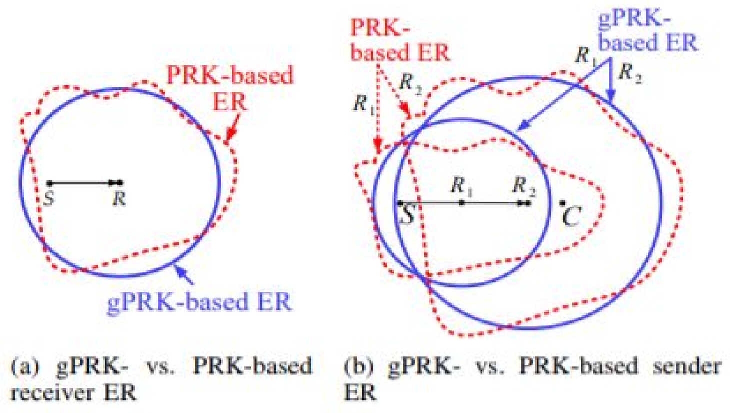


Figure 2: GCS Algorithm Cyber-Physical Scheduling for Predictable Reliability of Inter-Vehicle Communications Hongwei Zhang et. al.

The AV data packet contains the information of the vehicle's status for the surrounding vehicles. However, in our simulation, the required information that is needed to build the PRKS will be included, so that the GCS algorithm can schedule the transmission and avoid co-channel interference.

For vehicle mobility dynamics, we use the SUMO simulator that simulates vehicle traffic flow dynamics at high-fidelity based on real-world road and traffic conditions of Ames, Iowa, USA. To integrate these two systems, we will use the TraCI (Traffic Control Interface) of SUMO.

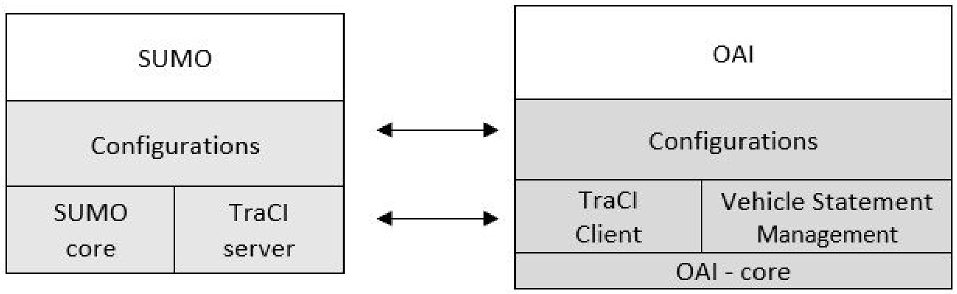


Figure 3: Integration of SUMO with OAI

After integrating OAI with SUMO, we will be able to check the reliability of the communication when considering the dynamics of a system with 75 mobile UEs. This environment can reflect the real-world road condition.

2.3 Hardware Implementation

Due to the limited number of SDRs that we have available for use, we will be unable to run physical tests of the same magnitude as the virtual simulation. However, the fundamental mechanics of the scheduling algorithm should be the same regardless of the size of the network. The proposed hardware implementation will be performed in Iowa State's campus with five vehicles mounted with SDRs. The SDR B210s will be mounted on each vehicle, OAI's UE with our CPS addition will be uploaded into SDR. With the VERT 2450 antennas, each vehicle will communicate and check the transmission and receiving signals from the surrounding UEs.

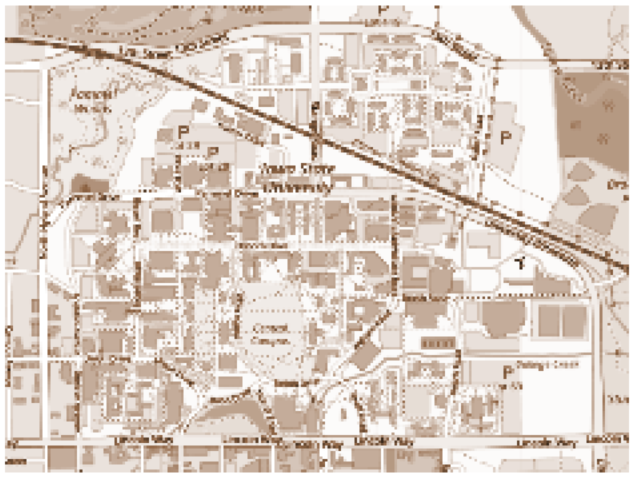


Figure 4: Map of Iowa State Campus. [12]

Figure 5: VERT 2450 Antenna

Figure 6:SDR B210 (Board Only)

The proposed design will present a solution to interference between V2V communication. Integrating with SUMO, the simulation can provide a more realistic V2V communication environment. The algorithm built over OAI is highly compatible with SDRs, and as a result, this project can provide a more realistic platform.

2.4 Design Objectives, System Constraints, Design Trade-offs

The GCS algorithm over the OAI framework will be heavily based on the software simulation, therefore the following strengths and weaknesses are expected:

2.4.1 Design Strengths

1. The advantage of using OAI is that OAI can implement the full protocol stack to run on a real execution environment respecting frame timing constraints. As a result, it is a more realistic platform (even in simulation) and has high compatibility with a SDR.
2. Our design is integrated with SUMO, so we can provide the mobility dynamics of a moving vehicle to UEs. As a result, it can reflect the real-world communication environment.
3. Our implementation will achieve reliability figures well beyond what other implementations have been able to achieve.

2.4.2 Design Weaknesses

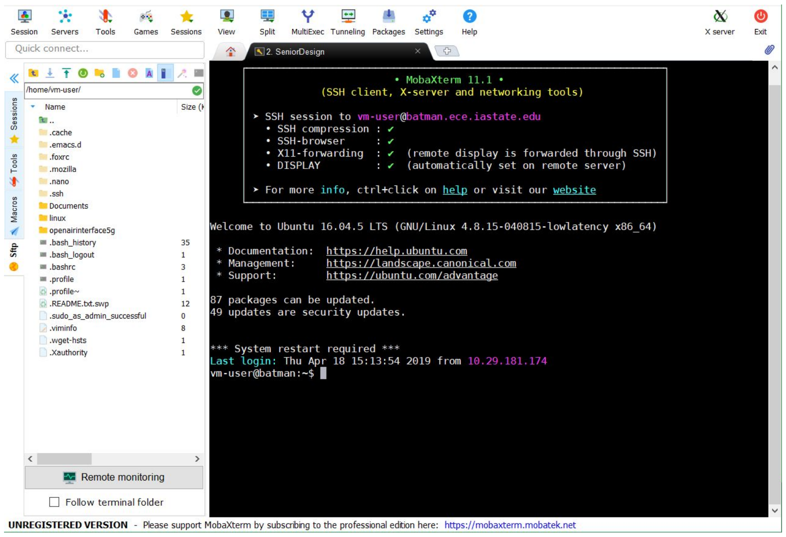
1. The algorithm, at this stage, will not consider obstacles such as buildings, terrain, weather, or other possible physical obstacles that exist in the real world. The parameter-K can take these factors into consideration, but that is outside the scope of our implementation.
2. Because the algorithm is mostly focused on reliability, latency is a secondary priority in this project. In more advanced implementations, this issue will need to be resolved in order for V2V communication to be safe.

3. Our project is using a platform (OAI) that is currently being modified regularly, and as a direct result, our product depends heavily on updates to OAI. When OAI receives an update, our project may possibly need to be updated as well, depending on the backwards-compatibility of the update.

3. Implementation

3.1 OAI

We have installed OAI on ETG server on Linux Operating System. The installation for OAI is required Ubuntu version 16.04.02 with the 4.8 low latency kernel. Along with the successful installation of it we also created a list that was necessary for anyone trying to install it in the future. The reason for this is that no installation step by step existed so we created one for the project.



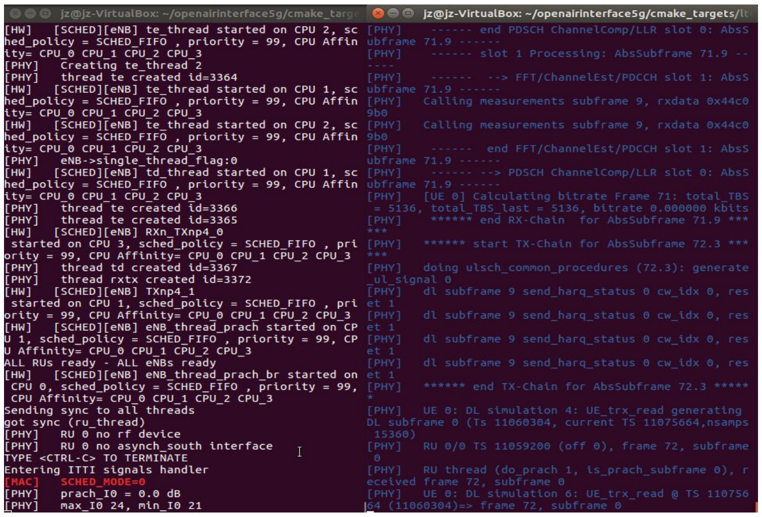
```
/home/vm-user/
Name                          Size (K
..
.cache
.emacs.d
.foxrc
.mozilla
.nano
.ssh
Documents
linux
openairinterface5g
.bash_history                 35
.bash_logout                  1
.bashrc                       3
.profile                      1
.profile~                     1
.README.txt.swp               12
.sudo_as_admin_successful     0
.viminfo                      8
.wget-hsts                    1
.Xauthority                   1
```

```
                    • MobaXterm 11.1 •
          (SSH client, X-server and networking tools)

 ➤ SSH session to vm-user@batman.ece.iastate.edu
   • SSH compression : ✔
   • SSH-browser     : ✔
   • X11-forwarding  : ✔  (remote display is forwarded through SSH)
   • DISPLAY         : ✔  (automatically set on remote server)

 ➤ For more info, ctrl+click on help or visit our website

Welcome to Ubuntu 16.04.5 LTS (GNU/Linux 4.8.15-040815-lowlatency x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

87 packages can be updated.
49 updates are security updates.


*** System restart required ***
Last login: Thu Apr 18 15:13:54 2019 from 10.29.181.174
vm-user@batman:~$
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

Figure 7: Server

One of the main requirements of the project was to modify the OAI code. Which proved to be more complicated than originally predicted. The reason for this is that the code had very little comments and a short readme. Our goal was to modify the UE code to be able to store appropriate vehicle data. On top of that we also had to modify the eNB code to contact SUMO during the UE update phase, and to use the GCS algorithm to predict future interference. We were given some code from our client to get started in OAI. It was helpful but, it was from a much older version of the code with many changes needed to make the code function.
In the end we managed to get OAI installed and running on a personal virtual machine. Running multiple UE's with a single eNB. While we did make modifications to the OAI code we weren't far enough along for anything to compile. We hope to be able to pass the project to another group. During the process of all this we documented all of our steps and thoughts, so that should make it easier on a future group.


jz@jz-VirtualBox: ~/openairinterface5g/cmake_targ
jz@jz-VirtualBox: ~/openairinterface5g/cmake_targets/lt

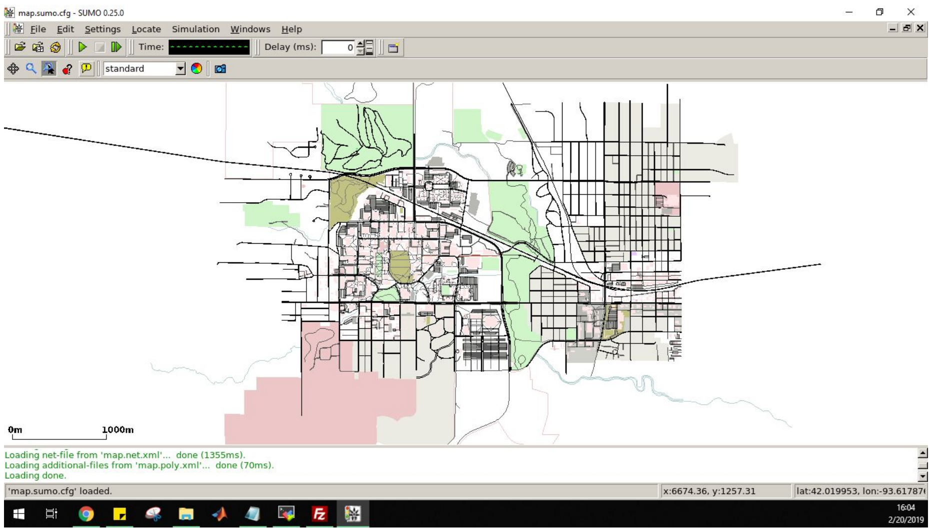
```
[HW]   [SCHED][eNB] te_thread started on CPU 2, sc
hed_policy = SCHED_FIFO , priority = 99, CPU Affin
ity= CPU_0 CPU_1 CPU_2 CPU_3
[PHY]   Creating te_thread 2
[PHY]   thread te created id=3364
[HW]   [SCHED][eNB] te_thread started on CPU 1, sc
hed_policy = SCHED_FIFO , priority = 99, CPU Affin
ity= CPU_0 CPU_1 CPU_2 CPU_3
[HW]   [SCHED][eNB] te_thread started on CPU 2, sc
hed_policy = SCHED_FIFO , priority = 99, CPU Affin
ity= CPU_0 CPU_1 CPU_2 CPU_3
[PHY]   eNB->single_thread_flag:0
[HW]   [SCHED][eNB] td_thread started on CPU 1, sc
hed_policy = SCHED_FIFO , priority = 99, CPU Affin
ity= CPU_0 CPU_1 CPU_2 CPU_3
[PHY]   thread te created id=3366
[PHY]   thread te created id=3365
[HW]   [SCHED][eNB] RXn_TXnp4_0
 started on CPU 3, sched_policy = SCHED_FIFO , pri
ority = 99, CPU Affinity= CPU_0 CPU_1 CPU_2 CPU_3
[PHY]   thread td created id=3367
[PHY]   thread rxtx created id=3372
[HW]   [SCHED][eNB] TXnp4_1
 started on CPU 1, sched_policy = SCHED_FIFO , pri
ority = 99, CPU Affinity= CPU_0 CPU_1 CPU_2 CPU_3
[HW]   [SCHED][eNB] eNB_thread_prach started on CP
U 1, sched_policy = SCHED_FIFO , priority = 99, CP
U Affinity= CPU_0 CPU_1 CPU_2 CPU_3
ALL RUs ready - ALL eNBs ready
[HW]   [SCHED][eNB] eNB_thread_prach_br started on
 CPU 0, sched_policy = SCHED_FIFO , priority = 99,
 CPU Affinity= CPU_0 CPU_1 CPU_2 CPU_3
Sending sync to all threads
got sync (ru_thread)
[PHY]   RU 0 no rf device
[PHY]   RU 0 no asynch_south interface
TYPE <CTRL-C> TO TERMINATE
Entering ITTI signals handler
[MAC]   SCHED_MODE=0
[PHY]   prach_I0 = 0.0 dB
[PHY]   max_I0 24, min_I0 21
```

```
[PHY]    ------ end PDSCH ChannelComp/LLR slot 0: AbsS
ubframe 71.9 ------
[PHY]    ------ slot 1 Processing: AbsSubframe 71.9 --
----
[PHY]    ------  --> FFT/ChannelEst/PDCCH slot 1: AbsS
ubframe 71.9 ------
[PHY]   Calling measurements subframe 9, rxdata 0x44c0
9b0
[PHY]   Calling measurements subframe 9, rxdata 0x44c0
9b0
[PHY]    ------  end FFT/ChannelEst/PDCCH slot 1: AbsS
ubframe 71.9 ------
[PHY]    ------ --> PDSCH ChannelComp/LLR slot 0: AbsS
ubframe 71.9 ------
[PHY]   [UE 0] Calculating bitrate Frame 71: total_TBS
 = 5136, total_TBS_last = 5136, bitrate 0.000000 kbits
[PHY]    ****** end RX-Chain  for AbsSubframe 71.9 ***
***
[PHY]   ****** start TX-Chain for AbsSubframe 72.3 ***
***
[PHY]   doing ulsch_common_procedures (72.3): generate
_ul_signal 0
[PHY]   dl subframe 9 send_harq_status 0 cw_idx 0, res
et 1
[PHY]   dl subframe 9 send_harq_status 0 cw_idx 0, res
et 1
[PHY]   dl subframe 9 send_harq_status 0 cw_idx 0, res
et 1
[PHY]   dl subframe 9 send_harq_status 0 cw_idx 0, res
et 1
[PHY]   ****** end TX-Chain for AbsSubframe 72.3 *****
*
[PHY]   UE 0: DL simulation 4: UE_trx_read generating
DL subframe 0 (Ts 11060304, current TS 11075664,nsamps
 15360)
[PHY]   RU 0/0 TS 11059200 (off 0), frame 72, subframe
 0
[PHY]   RU thread (do_prach 1, is_prach_subframe 0), r
eceived frame 72, subframe 0
[PHY]   UE 0: DL simulation 6: UE_trx_read @ TS 110756
64 (11060304)=> frame 72, subframe 0
```

Figure 8: Communication Between UE and eNB

3.2 SUMO

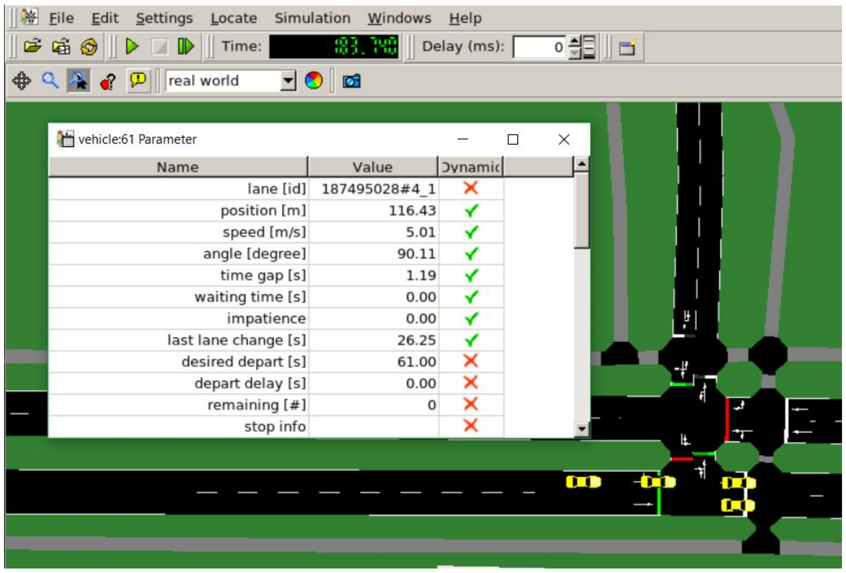
SUMO is an open source simulator from the Institute of Transportation Systems at the German Aerospace Center. Generating real-road conditions with vehicle's data including location, speed.
 is installed on the server. We imported Iowa State Campus map successfully on the simulation to provide the real road vehicle data to OAI.



```
Loading net-file from 'map.net.xml'... done (1355ms).
Loading additional-files from 'map.poly.xml'... done (70ms).
Loading done.
'map.sumo.cfg' loaded.
```

Figure 9: Imported Map of Iowa State Campus

We generated the 3600 vehicles on SUMO to generate realistic environment of the real road in campus. We can extract the each vehicle's data in given timestep. We extracted the the data of position, speed and acceleration of the vehicle in every 10ms. We can either extracted in xml format or sending it to the server so that it is compatible with OAI.



Name	Value	Dynamic
lane [id]	187495028#4_1	✗
position [m]	116.43	✓
speed [m/s]	5.01	✓
angle [degree]	90.11	✓
time gap [s]	1.19	✓
waiting time [s]	0.00	✓
impatience	0.00	✓
last lane change [s]	26.25	✓
desired depart [s]	61.00	✗
depart delay [s]	0.00	✗
remaining [#]	0	✗
stop info		✗

Figure 10: Data Extraction of Vehicles

3.3 Integration Between OAI and SUMO

Previously in the OAI production lifetime, this feature was implemented as a function within OAI and would require no extra coding on the part of users. However, shortly before the start of our project, this section was removed without our advisor's knowledge. Therefore, this presented an extra challenge for us that we did not anticipate.
The simulated vehicle data generated by SUMO, including the vehicle location, speed, acceleration, and other information, are exported to either an XML file or accessible via a server that can be created within SUMO. We wrote a C client to interact with the SUMO server, with the goal of extracting the vehicle data and relaying it directly to the GCS algorithm in the OAI platform. The GCS algorithm would then be able to use this information to determine whether any specific vehicles were likely to interfere with one another. The client to integrate these two platforms was partially successful but incomplete. The client can connect to the SUMO server, as well as successfully format and send the command to close the connection and kill the server. However, the format of the other commands does not follow the outline listed on the OAI website. This has made it difficult to manually format and send correct messages. Due to this setback, we were unable to correctly send the majority of the vehicle data queries.

4. Testing, Validation, and Evaluation

A4.1 Testing Plan, Testing Environment & Methods

Type	OS	Kernel	OAI Version	SUMO Version
Linux Virtual Machine/ETG Server	Ubuntu 16.04.02	Low-Latency 4.8	1.0.0	0.25.0

Table 1. Testing Environment

OAI Stress Testing	Number of UEs	2
	Number of eNBs	1
SUMO Simulation	Number of Vehicles	3600
	Testing Area	Iowa State Campus
GCS a_ACC	Tested without Kalman Filter. Outputs expected results as defined by the algorithm formulas	
GCS Approximation	Outputs similar latitude and longitude to future SUMO data	

Table 2. Testing Strategy

Our major goal of this project is to identify the interferences between vehicles' communication. GCS algorithm allows to minimize the interference along with maximizing the reliability and throughput.

a. **Test Method**: We count the number of the successful communication between each nodes in given area and timestpes. We compare the number of successful communication with conventional V2V communication.
b. **Expected Results**: We expect higher number of successful communication with GCS algorithm.

We are also testing four non-functional requirements of the project.

1. **Reliability:** The reliability is the essential part of V2V communication. Conventional network system scheduling protocols are unable to overcome the interferences between vehicles. Theoretically, we can obtain high reliability with the CPS algorithm, and we will test through simulation with OAI and SUMO.

   a. **Test Method:** Tests will be implemented by sending the certain number of data packets (Rx) and checking the successfully received data packets (Tx). The reliability will be measured as, Tx/Rx * 100.
   b. **Expected Results:** We expect that we can obtain more than 90% reliability over simulation ((Tx/Rx * 100) > 90) or packets lost less than 10%.

2. **Delay:** Delay is the other factor that we need to consider for simulations. In fast moving vehicles, a tiny amount of time delay can be a tremendous difference. The latency for 5G is conceptually around 1 ms. However, we will not be able to obtain this goal. The delay level will be as same as 4G network.
   a. **Test Method:** We will timestamp events and then calculate the time difference between events.
   b. **Expected Results:** Current 4G network has a delay of 50 ms. Although it is not good enough for AVs, due to practical limitations, we expect that delay will be similar to the 4G network. In a crowded region, we expect more than 50 ms.

3. **Concurrency:** The number of simultaneous non-interfering transmissions in the same time slot is one part we should consider.
   a. **Test method:** In each time slot, we will count the number of active transmissions between nodes.
   b. **Expected results:** We expect to have high number of non-interfering concurrency with the GCS algorithm to ensure high reliability, high throughput, and low latency.

4. **Throughput:** The rate of successful packet transmission with respect to time should be maximized, but is a low priority than concurrency.
   a. **Test method:** We will measure the number of the packets per unit time for each communication. Each packet is 1kB.
   b. **Expected results:** Using the GCS algorithm, we expect to retain high throughput representing a high successful packet delivery ratio.

4.2 Unit Testing (includes types of completed work)

4.2.1 Sumo Testing

In order to verify that the SUMO simulation would meet our needs it was tested running a simulation using a map of the Iowa State University campus. These tests where verified by running the simulation and observing the generated traffic conditions. We first tried testing using only 100 vehicles but found that it did not provide adequate traffic density to test our algorithm.

In order to solve this issue we increased the number of vehicles to 3600 which provided the necessary density to test our algorithm.

4.2.2 OAI Testing

To be able to test our GCS algorithm we needed to incorporate a simulation of communication to mimic that of cellular devices communicating in a congested area.OAI is able to provide this scenario simulating communication between multiple UEs in an unscheduled manner allowing us to validate our GCS algorithm. We were able to successfully test 1 eNB and 2 UEs communicating with the simulated network but we were unable to do more large scale testing due to issues in implementing our algorithm with OAI.

4.2.3 GCS Testing

Most of the testing for the GCS algorithm involved testing our equations for determining the future positions of vehicles. These equations are a key part of the algorithm and provided us the ability to determine possible future conflicting vehicles. These equations were tested in a basic form using simulation data gathered from SUMO as well as edge cases to make sure the equations provided correct information. We ran into issues with further testing due to difficulties in implementing an Unscented Kalman filter which would have provided more accurate estimations. Due to these issues we were unable to test a fully implemented version of our algorithm.

4.3 Validation and Verification

We were able to make the software validation and verification for SUMO and OAI.

1. SUMO
   We were able to import the map of Iowa State Campus and run the desired number of vehicles. We generated 3600 vehicles for one hour and extracted the vehicle's data with the timestep of 10ms.
2. OAI
   We were able to make one UE and one eNB communication which is the base step of generating multiple nodes communication. Link below is the simulation of OAI testing.
   OAI Testing
   https://www.youtube.com/watch?v=OpZYZI4kq1Y

However, we were not able to make the functional and nonfunctional testing due to uncompleted algorithm and software integration.

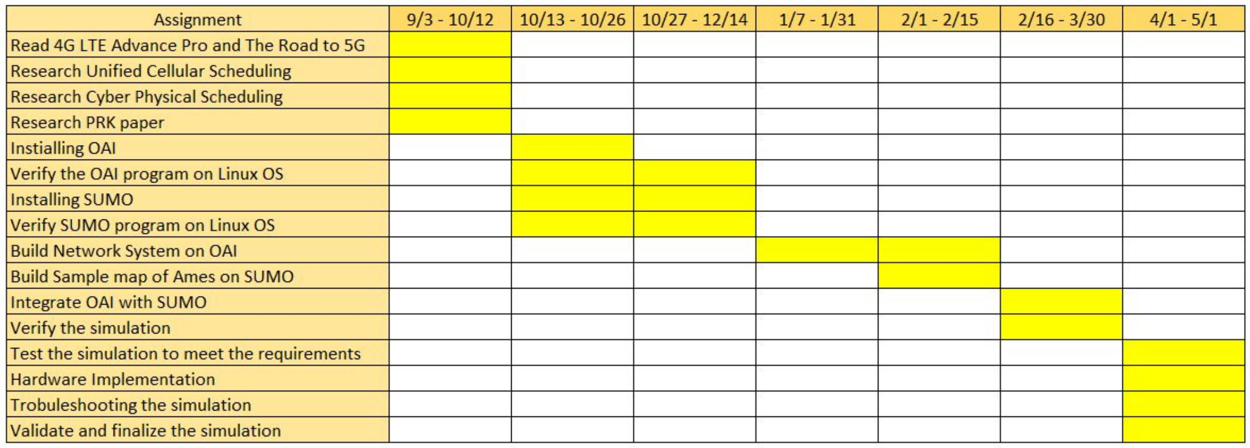
5. Project and Risk Management

5.1 Task Decomposition & Roles and Responsibilities

1. Ted Miller
    a. Ted had multiple roles in the project. His role from the start was to do the meeting minutes and keep the groups trello up to date. During the course of the project he worked on most of the documents in some way and on the code section. The main section of the code he worked on was OAI and the GCS algorithm.
2. Jaime Zetina
    a. Jaime worked on many different aspects of the project. He kept track of the weekly project reports for the group. Worked on the documentation for the project, OAI, and SUMO. Assisted Ted to understand certain math issues.
3. Anthony Benson
    a. Anthony made many aspects of the project easier. He worked on the project documents. On top of that he was one of the lead figures on the programming side of the project helping the keep people motivated and moving. He was responsible for editing the code to OAI and just figuring out the code in general.
4. Jared Gorton
    a. Jared had his hands on many parts of the project. Working on the project documents was one of the roles that he had. Along with that he was part of the coding group for the project. His main focus for that was getting SUMO and OAI to communicate with each other.
5. Khanh Luu
    a. Just like everyone else in the project Khanh worked on the project documents. On top of that he was also part of the programming group. He was worked on the server for the project and getting OAI up and running. On top of that he was wandering help for the programming group helping where needed.
6. Hye-Sung Moon
    a. Project documents were a key thing for Hye-Sung. He was the main person who kept the group moving towards it destination and on track. Along with that he served as the main point of contact for the advisor/client to the group. He also worked on getting Ames map in SUMO and getting OAI running.

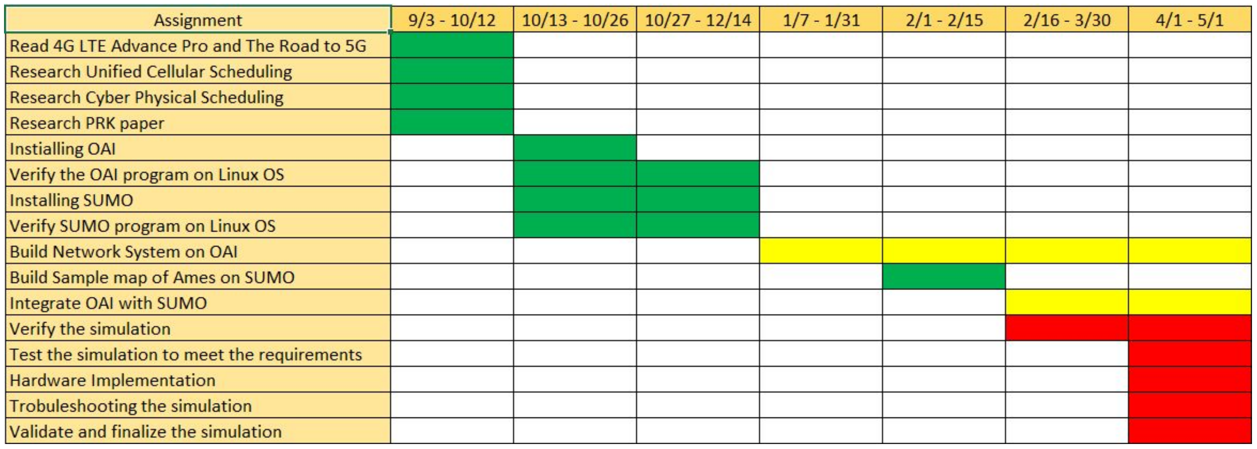
5.2 Project Schedule

5.2.1 Predicted Schedule



Assignment	9/3 - 10/12	10/13 - 10/26	10/27 - 12/14	1/7 - 1/31	2/1 - 2/15	2/16 - 3/30	4/1 - 5/1
Read 4G LTE Advance Pro and The Road to 5G	🟨						
Research Unified Cellular Scheduling	🟨						
Research Cyber Physical Scheduling	🟨						
Research PRK paper	🟨						
Instialling OAI		🟨					
Verify the OAI program on Linux OS		🟨	🟨				
Installing SUMO		🟨	🟨				
Verify SUMO program on Linux OS		🟨	🟨				
Build Network System on OAI				🟨	🟨		
Build Sample map of Ames on SUMO					🟨		
Integrate OAI with SUMO						🟨	
Verify the simulation						🟨	
Test the simulation to meet the requirements							🟨
Hardware Implementation							🟨
Trobuleshooting the simulation							🟨
Validate and finalize the simulation							🟨

5.2.2 Actual Schedule



Assignment	9/3 - 10/12	10/13 - 10/26	10/27 - 12/14	1/7 - 1/31	2/1 - 2/15	2/16 - 3/30	4/1 - 5/1
Read 4G LTE Advance Pro and The Road to 5G	🟩						
Research Unified Cellular Scheduling	🟩						
Research Cyber Physical Scheduling	🟩						
Research PRK paper	🟩						
Instialling OAI		🟩					
Verify the OAI program on Linux OS		🟩	🟩				
Installing SUMO		🟩	🟩				
Verify SUMO program on Linux OS		🟩	🟩				
Build Network System on OAI				🟨	🟨	🟨	🟨
Build Sample map of Ames on SUMO					🟩		
Integrate OAI with SUMO						🟨	🟨
Verify the simulation						🟥	🟥
Test the simulation to meet the requirements							🟥
Hardware Implementation							🟥
Trobuleshooting the simulation							🟥
Validate and finalize the simulation							🟥

Legend: Green is things that got done on time, Yellow is things that are currently being worked, Red is things we never got to.

5.3 Risks and Mitigation

5.3.1 Potential Risks

1. One of the Risks that we had originally thought of was that we would be at the mercy of OAI and SUMO software updates. We mitigated that issue by choosing just not to update the software.
2. Hardware issues were another risk that we had thought of. To be more specific that implementing our code into the hardware would be harder than expected. In an attempt to mitigate that issue we gave it more time in the schedule.

3. Lack of computing power was a big worry for us as a group. Both OAI and SUMO take a great deal of ram and processing power to run. To fix this risk we got a server from the ETG which had a great deal more power than our laptops.
4. Another possible risk we had thought of was how hard it was to set up OAI. The way we thought of to mitigate this was making step by step instructions, on how to install OAI and the features needed.

5.3.2 Actual Risks

All of the risks that we had prepared for did become and issue. Sadly the hardware risk never entered the picture at all. We never got far enough in the project to get to the hardware testing

1. Parts of OAI being out of date was a risk that we ran into. More specifically the fact that OAI's built in integration with SUMO no longer existed. In an attempt to mitigate this risk we tried to create a new integration system for OAI and SUMO.
2. Failure to understand different topics that came up during the project. Over the course of the project many different new topics came up. We took several steps in-order to mitigate these issues. The first was reading research papers on the subject. Secondly we talked to our advisor/client who is an expert in the subject. Finally when all else failed we tried to outsource help from other people which included grad student friends, other professors, and the internet.
3. The project at the end became very software heavy. This became in issue because our group has two electrical engineers in it. So at the end the project for them became mostly research and writing different documents for the project.
4. Another risk we ran into was the tasks taking longer time then we had thought. In the end our mitigation tactic was to just cut different parts from the project. Which is not the best solution to anything like this, but it proved to be necessary.

5.4 Lessons learned

Don't trust projects that sound fun.

1. Lacking proper background in the proposed subject matter is a serious hurdle to overcome and becomes a hindrance in project goals.This project was mostly about Network Systems and Dr. Zhang's previous work. This lack in background lead to an unexpected amount of time dedicated to understanding and researching relevant materials. In our opinion much of the relative material provided was complex in nature, requiring proper background and knowledge to comprehend and leverage for our purpose.

2. Proper time management is vital to completion of a project. The amount of research needed to be done before beginning our actual design became more intensive throughout the project consuming more of our time and hindering our progress. Needing to learn various software, various new concepts covering wireless communication, and previous work by Dr. Zhang. Further research needed to be conducted when questions arose

regarding the material being read, because our adviser had become unfamiliarized with some of the specifics of these topics. He would recommend additional reading materials for research.The amount of time needed for the research became overwhelming for the group due to our responsibilities as full time students. There is a list of research material we were recommended to read during the project is included in Appendix.

3. We have been using OAI and SUMO which is only used specific in the wireless communication industry and academia requiring in-depth level knowledge. There is not an abundance of user friendly material that we can refer to during our installation process, in addition running these software is very challenging due to their extremely specific operating environments. OAI is both an open source software and in a constant state of development, therefore it does not have much supporting documentation or guidelines so we had to complete more researching time to develop a strategy on how to install, and how to run the OAI to simulate the UEs and eNBs. Using open source software comes with many risks for the progress of a project.

4. Use all available resources as extensively as possible, while it is still possible and a good use of time. At the first moment that difficulties are encountered, it is extremely beneficial to use every resource available and identify the resources that are most effective and helpful. These resources will be the wisest use of time, and therefore should be used extensively.

5. More interaction between students and advisers may be required with actual progress check and deadlines. Although we met with our advisor at least once a week, it would have been better if we had been able to make our issues more clear to the advisor. Our advisor seemed not to realize the problems that we were encountering, which hindered our progress in this project. In effect, there were points in the project at which we thought we were making progress but in reality we were working in the wrong direction.

6. Documentation is incredibly important for projects that will be switching hands. When a project is expected to change between teams, the incoming team will likely need to spend great amounts of time understanding the existing code base. If the code is poorly commented, this process can take vastly longer than if the comments are frequent and well-placed.

6. Conclusions

It is unfortunate that we could not completely finish the project. There were several issues that developed in the course of the project.

1. Lack of background information is not good.

2. Time Constraint

3. Software Restriction

We, as a group, have set up many resources for a future group to take over for us:

1. Our google drive has many of the research papers that we read or links to them. It also has different documents listing information on OAI's code. We also have some virtual machine images to give the group a good starting point. If they want to set things up themselves we have multiple step by step set up instructions for different parts of the project.
2. For the Linux server we should request the permission on port 5500 in order to retrieve the OAI communication between the UEs and eNBs, or look into the code of OAI and modify the input/output port within the actual OAI code.
3. All of our code that we've worked on is on a gitlab that can be shared with anybody working on the project in the future.
4. OAI and SUMO need to be installed in a specific way to work and to be integrated with each other. We made step-by-step instructions on how to install and run these programs for the group who takes over our work.

7. References

[1] E. Dahlman, S. Parkvall, and J. Sköld, *4G, LTE-Advanced Pro and the Road to 5G*, Amsterdam: Elsevier, 2016.

[2] Open Air Interface, “Open Air Interface”, *Open Air Interface*, 2018. [Online]. Available: http://www.openairinterface.org/. [Accessed Oct. 24, 2018].

[3] Sumo. SUMO - Simulation of Urban Mobility, *German Aerospace Center, Institute of Transportation Systems*, 2018. [Online] Available at: http://sumo.dlr.de/index.html [Accessed Oct. 24, 2018].

[5] A. Woo, “Evaluation of Efficient Link Reliability Estimators for Low-Power Wireless Networks”, 2018. [online] Available at: http://digitalassets.lib.berkeley.edu/techreports/ucb/text/CSD-03-1270.pdf [Accessed 24 Oct. 2018].

[6] Ubuntu, “Ubuntu” *Canonical*, 2018. [Online]. Available: https://www.ubuntu.com/. [Accessed Nov. 12, 2018]

[7] H. Zhang, “Scheduling with Predictable Link Reliability for Wireless Networked Control”, 2018 [Online] Available at: https://www.ece.iastate.edu/~hongwei/group/publications/PRKS-TWC.pdf [Accessed 24 Oct. 2018].

[8] H. Zhang, “Cyber-Physical Scheduling for Predictable Reliability of Inter-Vehicle Communications”, 2018. [Online] Available at: http://www.ece.iastate.edu/~hongwei/group/publications/CPS-IOTDI18.pdf [Accessed 24 Oct. 2018]

[9] H. Zhang, “Link Estimation and Routing in Sensor Network Backbones: Beacon-based or Data-driven?”, 2018. [Online] Available at: http://www.cs.wayne.edu/%7Ehzhang/group/publications/LOF-TMC.pdf [Accessed 24 Oct. 2018].

[10] L. Carrillo, “vehicle-to-vehicle communications”, 2016. [Online]. Available at: https://www.a1autotransport.com/how-v2v-technology-will-make-driving-more-safe/. [Accessed Nov. 12, 2018].

[11] “URSP B210(board only)”, 2018. [Online]. Available at: https://www.ettus.com/product/details/UB210-KIT. [Accessed 12, Nov. 2018].

[12] OpenStreetMap, “OpenStreetMap”, *OpenStreetMap*, 2018. [Online] Available: https://openstreetmap.org [Accessed 12 Nov. 2018].

[13] “Vert2450 Antenna”, 2018. [Online]. Available: https://www.ettus.com/product/details/VERT2450. [Accessed 2 Dec. 2018].

[14] E. Wan and R Merwe, “The Unscented Kalman Filter for Nonlinear Estimation”, Oregon Graduate Institute of Science & Technology,[Online Document],2000. Available: https://www.seas.harvard.edu/courses/cs281/papers/unscented.pdf [Accessed 3/27/19]